INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI®

Bell & Howell Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

Numerical Evaluation of Wind-Induced Dispersion of Pollutants around Buildings

Ye Li

A Thesis

in

the Department of Building, Civil and Environmental Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy at

Concordia University
Montreal, Quebec, Canada

October 1998

ABSTRACT

Numerical Evaluation of Wind-Induced Dispersion of Pollutants around Buildings

Ye Li, Ph.D.
Concordia University, 1998

A detailed and comprehensive numerical study of wind-induced pollutant dispersion around buildings has been carried out. Research in this area was reviewed and analyzed. The results from various studies were compared. Based on the review and the comparison with various studies, a more effective methodology for the numerical evaluation of dispersion of pollutants around buildings has been proposed.

Computed results of a two-layer approach show various degrees of improvement for the concentration field in the wake region and on the back wall of the building. Results from two-layer approach also show improvement on the building roof for along-wind direction. However, the two-layer approach does not seem to work well for roof concentration along the cross-wind direction; this is probably due to the drawback of the isotropic turbulence model used.

The previous numerical studies on pollutant dispersion around rectangular buildings only investigated the concentration profiles for along-wind direction or vertical direction, and these were only for limited situations. A systematic examination of

pollutant concentration in the wake of a rectangular building has been carried out for two different release sources. The concentration profiles have been investigated for along-wind, across-wind and vertical directions for various locations. The numerical simulation results have been compared with those from the experiments and from the Modified Gaussian Model. Results from numerical simulation generally agree well with those from the experiments. It was found that the numerical simulation predicts ground level pollutant concentrations better than the Modified Gaussian Model in the near building wake area.

The influence of numerical errors on the computed results and the effects of thresholds in the discretization and convergence process have been investigated for the first time in the area of numerical simulation of pollutant dispersion around buildings.

Acknowledgement

I wish to express my profound gratitude to Dr. Theodore Stathopoulos for his constant support and encouragement in the course of this work, and for many stimulating suggestions and discussions.

A special note of appreciation and thanks are due to Dr. Hanqing Wu for numerous valuable helps during the course of this study. Thanks are also due to Dr. Patrick Saathoff for many useful discussions during this study. Many thanks to Mr. Sylvain Belanger for assistance in the utilization of the computer facilities.

My sincere thanks to CERCA (Center for Research on Computation and its Applications) for offering me an opportunity to use their computer facilities for some computer runs. Suggestions from Dr. Anne Bourlioux during my staying at CERCA are also deeply appreciated.

A very special thank you is given to my parents; whose unselfish love has been without bounds.

TABLE OF CONTENTS

LIST OF FIGURES xi

LIST OF TABLES xvi

NOMENCLATURE xvii

Chapter 1

INTRODUCTION 1

1.1 Wind Engineering 1

1.2 Air Pollution around Buildings 2

1.3 Computational Wind Engineering 8

Chapter 2

LITERATURE REVIEW 10

2.1 Introduction 10

2.2 Literature Survey 10

2.2.1 Theoretical and experimental study of air pollutant dispersion 10

2.2.2 Numerical simulation of flowfield 20

2.2.3 Numerical simulation of air pollutant concentration field 23

2.2.4 Error estimation of computational results 24

2.3 Justification of the Present Study 27

Chapter 3

NUMERICAL PROCEDURES 29

3.1 The Governing Equations 29

3.1.1 Reynolds-averaged Navier-Stokes equations 30

3.1.2 Concentration equation 31

3.1.3 Equations for the two-layer method 32

3.2 The Discretization Method 33

3.2.1 Flow field 34

3.2.2 Concentration equation --- explicit method 34

3.2.3 Concentration equation --- implicit method 39

3.3 Grid Generation 42

3.4 Boundary Conditions 43

3.4.1 Inlet condition of velocity 44

3.4.2 Inlet condition of turbulence properties 44

3.4.3 Outlet boundary conditions 45

3.4.4 Side and top of the computation domain 45

3.4.5 Solid boundaries 46

3.4.6 Zonal treatment of k and ε 47

3.5 Convergence Criteria 47

3.5.1 Difference of iterates 48

3.5.2 Relative residue 49

3.5.3 Ferziger's convergence criterion 49

3.6 Computational Codes 50

Chapter 4

FLOW FIELD AND CONCENTRATION FIELD AROUND A CUBIC BUILDING: A COMPARATIVE STUDY 51

4.1 Introduction 51

4.2 Numerical Simulation Set-Up 53

4.3 Concentration Field in the Building Wake 57

4.4 A Study of the Effect of the Stack Height 59

4.5 Summary 62

Chapter 5

FLOW FIELD AND CONCENTRATION FIELD AROUND A RECTANGULAR BUILDING 63

5.1 Introduction 63

5.2 Numerical Simulation Set-Up 65

5.2.1 Pollutant dispersion from a ground source 65

5.2.2 Pollutant dispersion from an elevated source 68

5.3 Flow Field around the Rectangular Building 72

5.4 Concentration Field around the Rectangular Building 75

5.4.1 Concentration field around the rectangular building for ground source 75

5.4.2 Concentration field around the rectangular building for elevated source 81

5.5 Error Analysis 83

5.5.1 Introduction 83

5.5.1.1 Discretization error 84

5.5.1.2 Iteration convergence error 85

5.5.2 The case investigated 86

5.5.3 Results and discussion 86

5.6 Summary 88

Chapter 6

FLOW FIELD AND CONCENTRATION FIELD AROUND A CUBIC BUILDING: A TWO-LAYER APPROACH 91

6.1 Introduction 91

6.2 Numerical Simulation Set-Up 91

6.3 Flow Field around the Cubic Building 93

6.4 Mean Pressure Field on the Surface of a Cubic Building 98

6.5 Concentration Field around the Cubic Building 99

6.5.1 Concentration field on the building surface 100

6.5.2 Concentration field in near-wake region behind building 104

6.6 Summary 111

Chapter 7 CLOSURE 113

7.1 Concluding Remarks 113

7.2 Research Contributions 114

7.3 Recommendations for Further Work 115

REFERENCES 117

APPENDICES

A. Fortran source code for WIPDB_I --- Wind-Induced Pollutant Distribution around Buildings by Implicit method 126

B. Fortran source code for WIPDB_E --- Wind-Induced Pollutant Distribution around Buildings by Explicit method 148

C. Fortran source code for TWIST --- Turbulent Wind Simulation Technique --- standard k-ε model approach 158

D. Fortran source code for TWIST --- Turbulent Wind Simulation Technique --- two-layer approach 192

LIST OF FIGURES

Chapter 1

1.1 Flow around a rectangular building (From: ASHRAE Handbook-1997 Fundamentals 15.2) 5

1.2 Typical pollutant dispersion around a rectangular building (From: ASHRAE Handbook-1997 Fundamentals 15.2) 6

1.3 Effect of the location and height of a stack on dispersion around a building (From: Wark and Warner (1998)) 7

1.4 Typical pollutant dispersion around a flat-roof building (From: ASHRAE Handbook-1997 Fundamentals 15.2) 8

Chapter 2

2.1 Coordinate system showing Gaussian distributions in the horizontal and vertical direction 14

2.2 Vertical concentration profiles taken through the plume centerline for an isolated stack (From: Huber and Snyder (1982)) 15

2.3 Lateral concentration profiles taken through the plume centerline for an isolated stack (From: Huber and Snyder (1982)) 16

2.4 Suggested use for adjusted dispersion equations (From: Huber and Snyder (1982)). 17

Chapter 3

3.1 Nodes and interfaces in x-y plane 40

3.2 Nodes and interfaces in x-z plane 40

Chapter 4

4.1 Model of flow near a sharp-edged building in a deep boundary layer (From: Hosker (1979)) 52

4.2 Geometry and parameters for the case of Brzoska et al (1996) 54

4.3 Geometry and parameters for the case of Zhang (1993) 54

4.4 Geometry and parameters for the present simulation 55

4.5 Grid system for the present simulation 56

4.6 Comparison of concentrations for roof top release from a cube 58

4.7 Comparison of vertical concentration field for different stack height releases 60

4.8 Comparison of horizontal concentration field for different stack height releases .. 61

Chapter 5

5.1 Experimental set-up for ground source, after Huber and Snyder (1982) 67

5.2 Grid system for the ground source 67

5.3 Experimental set-up for elevated source, after Selvam and Huber (1995) 69

5.4 Coarse grid system for elevated source 70

5.5 Fine grid system for elevated source 71

5.6 Vertical wake cavity and envelope measurements (From: Huber and Snyder (1982)) 73

5.7 Vertical velocity vector field (present simulation) 73

5.8 Lateral wake cavity and envelope measurements (From: Huber and Snyder (1982)) 74

5.9 Horizontal velocity vector field (present simulation) 74

5.10 Longitudinal ground-level concentration for ground source 76

5.11 Vertical concentration for ground source at $x/H_b = 5$ 77

5.12 Vertical concentration for ground source at $x/H_b = 10$ 78

5.13 Vertical concentration for ground source at $x/H_b = 15$ 79

5.14 Lateral ground-level concentration for ground source at $x/H_b = 5$ 80

5.15 Lateral ground-level concentration for ground source at $x/H_b = 15$ 80

5.16 Longitudinal ground-level concentration along the centerline for elevated source 81

5.17 Vertical profiles of concentrations along the centerline at $x/H_b = 3$ for elevated source 82

5.18 Relative residue of discretized equations vs. iteration steps in solving the flow field 86

5.19 Difference of iterates vs. iteration steps in solving the concentration field 87

5.20 The effects of discretization and convergence criteria on longitudinal ground-level concentrations 89

Chapter 6

6.1 The x-z plane grid outline 92

6.2 Velocity vector distributions on the vertical section of the cube, after Tsuchiya et al.(1996) 94

6.3 The velocity field at the longitudinal center plane around the front corner of a cubic building 95

6.4 Vertical velocity profile above the roof 96

6.5 Velocity vector field in the wake of a cube 97

6.6 Surface pressure distribution on the vertical plane 98

6.7 Surface pressure distribution on the horizontal plane 99

6.8 Sketch of the vent locations and the coordinate system 100

6.9 The concentration field on the roof of a cubic building (upward vent) 101

6.10 The concentration field on the rear wall of a cubic building (upward vent)..... 103

6.11 Concentration coefficient contour in near-wake region for central roof vent release at x/H = 2.0 105

6.12 Concentration coefficient contour in near-wake region for central roof vent release at x/H = 2.5 106

6.13 Concentration coefficient contour in near-wake region for central roof vent release at x/H = 3.5 107

6.14 Vertical concentration profiles at y/H = 0.0, x/H = 0.5 for downwind roof vent release 108

6.15 Vertical concentration profiles at y/H = 0.0, x/H = 2.5 for downwind roof vent release 109

6.16 Vertical concentration profiles at y/H = 0.0, x/H = 3.5 for downwind roof vent release 110

6.17 Vertical concentration profiles at y/H = 0.0, x/H = 4.5 for downwind roof vent release 111

LIST OF TABLES

Chapter 1

1.1 Health Effects of Carbon Monoxide (From: Wark and Warner (1998)) 2

1.2 Health Effects of Sulfur Dioxide (From: Wark and Warner (1998)).....3

NOMENCLATURE

a	coefficient of discretized equations (Chapter 3)
A	area of the largest side of the building
b	source term in the discretized equation (section 3.2.3)
c'	fluctuating concentration
C	mean concentration
$\tilde{C} = C + c'$	instantaneous concentration
C_1, C_2, C_3	constants used in the empirical formula of Huber and Snyder
$C_{1\varepsilon}$, $C_{2\varepsilon}$, C_μ	constants in k-ε turbulence model
C_D, C_ε	constants in Norris-Reynolds two-layer approach
D	difference of iterates divided by time step
	dilution ratio (chapter 2)
	molecular diffusion coefficient (section 3.1.2)
	diffusion conductance (section 3.2.3)
d	the distance from the point to the solid
f_μ	damping function in Norris-Reynolds two-layer approach
F	flow rate through a control volume face
h	stack height
H	building height
H_b	building height

k	turbulent kinetic energy
K	concentration coefficient
L	length-scale in Norris-Reynolds two-layer approach
n	iteration steps
P	augmented pressure
$\widetilde{P}$	instantaneous pressure
Q	pollutant release rate
r_n	relative residue
R_d	turbulence Reynolds number from the point at distance d from solid wall
t	time
U,V,W	mean velocity
U_H	reference velocity at building height
U_r	mean wind speed affecting the plume
$\widetilde{U}_i$	instantaneous velocity
x,y,z	space coordinate
XFLUX, YFLUX, ZFLUX	pollutants flux due to convection and diffusion across the control volume surfaces in x, y and z directions respectively

Greek Symbols

α	exponent of the power-law wind speed profile

δ	boundary layer depth
ε	turbulent kinetic energy dissipation
	discretization error (section 5.5.1)
κ	von-Karman constant
ν	kinematic viscosity
ν_t	turbulent eddy viscosity
ρ	fluid density
σ_c	constant in concentration equation
σ_k, σ_ε	constants in k-ε turbulence model
σ_y, σ_z	the standard deviation of the concentration distribution in the crosswind and vertical direction at the downwind distance x
σ'	enhanced dispersion parameter in Huber and Snyder's model
ϕ	represents C in the discretized equations (section 3.2.3)
	solution of the general engineering problem (section 5.5.1)
χ	mean concentration
Γ	ν_t/σ_c

Math Symbols

$\|\|..\|\|$	max(..) in section 3.2
$\|\|..\|\|$	The norm of (..) in section 3.5

Subscripts

e, w, n, s, t, b	the east, west, north, south, top and bottom interfaces of the reference point
E, W, N, S, T, B	the east, west, north, south, top and bottom neighboring points of the reference point
h, 2h	grid-point spacing
i, j, k	the x, y and z direction

Chapter 1

INTRODUCTION

1.1 Wind Engineering

Emerged in the 1960s', wind engineering is an interdisciplinary research area dealing among other things, with wind effects on buildings. Basically, there are three kinds of such effects: structural, environmental and energy-related.

Most of the early research activities in wind engineering concerned with the structural aspects. Experiments were carried out in aeronautical wind tunnels to evaluate wind pressure distribution on buildings so that local wind loads and total wind loads could be assessed for the design of building panels such as a window glass and the main frame of the building. However, the aeronautical wind tunnels cannot simulate the wind effects on buildings adequately; therefore, atmospheric boundary layer wind tunnels were built for wind engineering research purposes.

The environmental aspects of wind engineering research include air pollution and pedestrian level wind comfort around buildings.

Wind flow over buildings can increase the heat losses through the building envelope. The wind-induced pressure and air velocity around the inlet and exit of building ventilation systems can influence the efficiency of the system.

1.2 Air Pollution around Buildings

Airborne pollutants, besides other drawbacks, constitute a very serious health hazard. Table 1.1 and Table 1.2 show the health effects of Carbon Monoxide (CO) and Sulfur Dioxide (SO_2) respectively.

Table 1.1 Health Effects of Carbon Monoxide

(From: Wark and Warner (1998))

ENVIRONMENTAL CONDITION	EFFECTS
9 ppm 8-hr exposure	Ambient air quality standard
50 ppm 6-wk exposure	Structural changes in heart and brain of animals
50 ppm 50-min exposure	Changes in relative brightness threshold and visual acuity
50 ppm 8 to 12-hr exposure non-smokers	Impaired performance on psychomotor tests

Table 1.2 Environmental and Health Effects of Sulfur Dioxide

(From: Wark and Warner (1998))

CONCENTRATION	EFFECT
0.03 ppm, annual average	1974 air quality standard, chronic plant injury
0.037-0.092 ppm, annual mean	Accompanied by smoke at a concentration of 185 $\mu g/m^3$, increased frequency of respiratory symptoms and lung disease may occur
0.11-0.19 ppm, 24-hr mean	With low particulate level, increased hospital admission of older persons for respiratory diseases may occur. Increased metal corrosion rate
0.19 ppm, 24-hr mean	With low particulate level, increased mortality may occur
0.25 ppm, 24-hr mean	Accompanied by smoke at a concentration of 750 $\mu g/m^3$, increased daily death rate may occur; a sharp rise in illness rates
0.3 ppm, 8 hr	Some trees show injury
0.52 ppm, 24-hr average	Accompanied by particulate, increased mortality may occur

In the tables, the quantity of a gaseous pollutant present in the air is expressed by parts per million (ppm) defined as:

$$\frac{1\ volume\ of\ gaseous\ pollutant}{10^6\ volume\ (pollutant + air)} = 1\ ppm$$

It can also be expressed by micrograms of pollutant per cubic meter of air ($\mu g/m^3$). The basic relation between $\mu g/m^3$ and ppm at 1atm and 25°C is:

$$\mu g / m^3 = \frac{ppm \times molecular\ weight}{24.5} \times 10^3$$

where 24.5 is a constant which is equal to $P/(R_u T)$ at P=1 atm, T = 25°C = 298°K and universal gas constant R_u = 0.08208 atm·m³/Kg·mol·°K. There are also other pollutants such as Hydrocarbons, Nitrogen Dioxide etc., which can also cause health problems for people. For each type of pollutant, there is an allowable concentration (Threshold Limit Value (TLV)) defined as: "the concentration level under which it is believed that nearly all individuals may be repeatedly exposed day after day without adverse effects." (American Conference of Governmental Industrial Hygienists, 1988). Whenever the Threshold Limit Value is exceeded in a certain area, there are possible adverse health effects to the people around that area.

Under the influence of the wind, the pollutants could contaminate the area around the building as shown in Figure 1.1. If the pollutants are reinjected either via the open windows or through the intake of ventilation system, the health of people inside the building may be severely influenced. Sometimes, this may require the evacuation of the building. Pollutants around a building can also influence the people nearby.

The entrainment of plumes emitted from short stacks into the wakes of buildings can result in maximum ground-level concentration that are significantly greater than those found for similar sources in the absence of buildings. This is often potentially

hazardous. Dilution of these emissions with surrounding air along with the application of control devices is necessary in order to meet ambient air quality standards. The effectiveness of such a control device depends upon the designer's estimate of the probable external dilution. Thus it is important to evaluate the ground level concentration in the building wake.

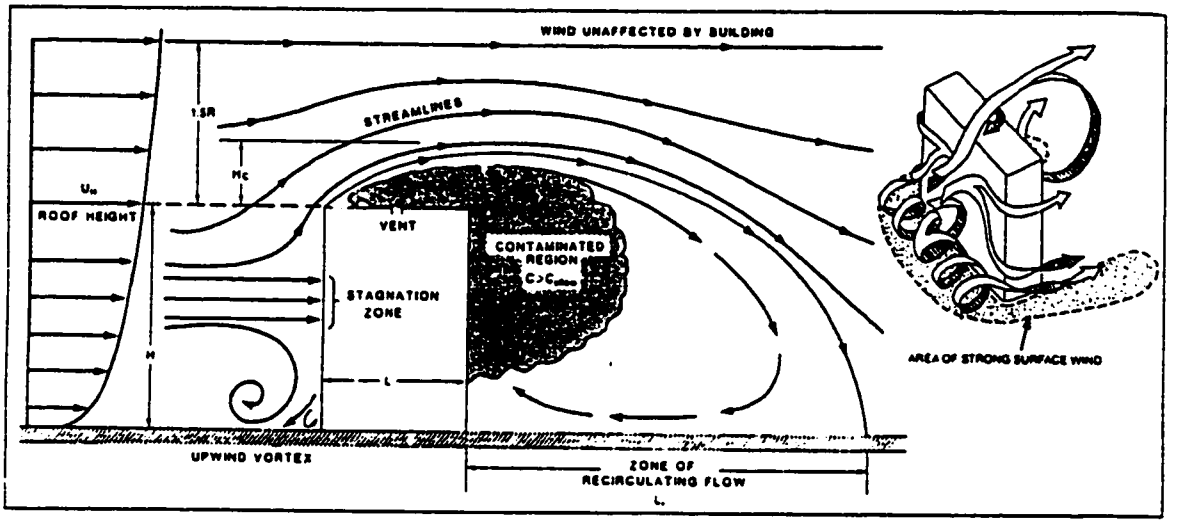


Figure 1.1 Flow around a rectangular building

(From: ASHRAE Handbook-1997 Fundamentals 15.2)

Very low concentrations of some fumes, like the sulfur compounds, can cause rapid deterioration of the building and air-conditioning equipment. So there are additional cost considerations that should be added to the toxic problems. Evaporative coolers and system cooling coils are particularly vulnerable to corrosion because they are wet. Re-entry from cooling towers can also be a problem. Cooling tower fog could enter air-conditioning inlets and cause either heavy icing in the winter, or greatly increased load on the refrigeration

equipment in the summer. Thus air pollution around buildings is of great concern for architects, HVAC engineers and health physicists.

Unfortunately, pollutant dispersion around buildings is a very complex phenomenon, which includes oncoming atmospheric boundary layer flow, stagnation zone and upwind vortex in front of the building and building wake recirculating zone as shown in Figure 1.1. On the roof, the flow can be divided into the roof recirculation region, high turbulence region and roof wake region as shown in Figure 1.2.

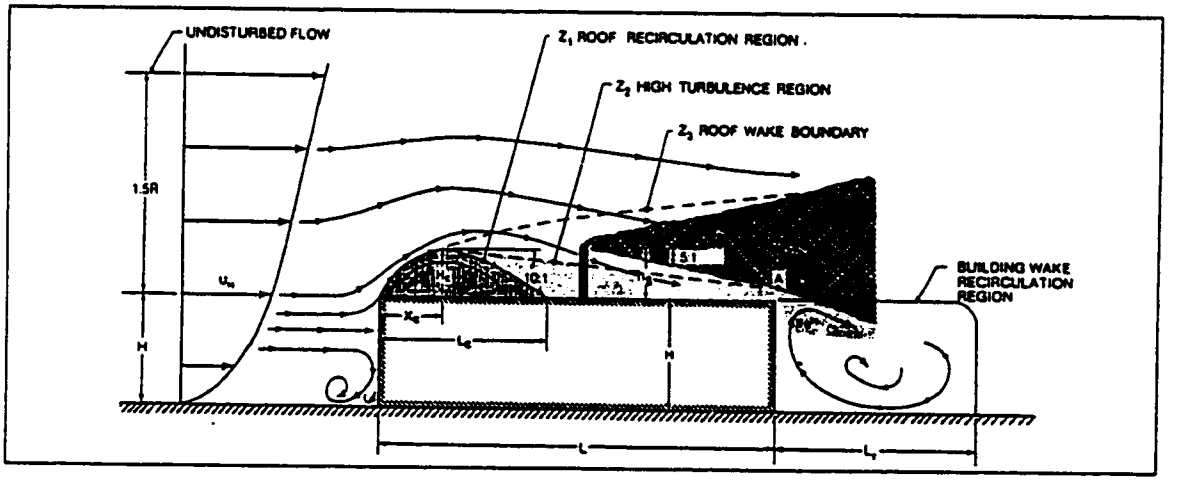


Figure 1.2 Typical pollutant dispersion around a rectangular building

(From: ASHRAE Handbook-1997 Fundamentals 15.2)

The location and height of the exhaust stack can also play an important role for the pollutants around the building, as shown in Figure 1.3.

The problem becomes much more difficult when the complex geometry of the building is considered since more separation and recirculation areas will appear, as shown in Figure 1.4.

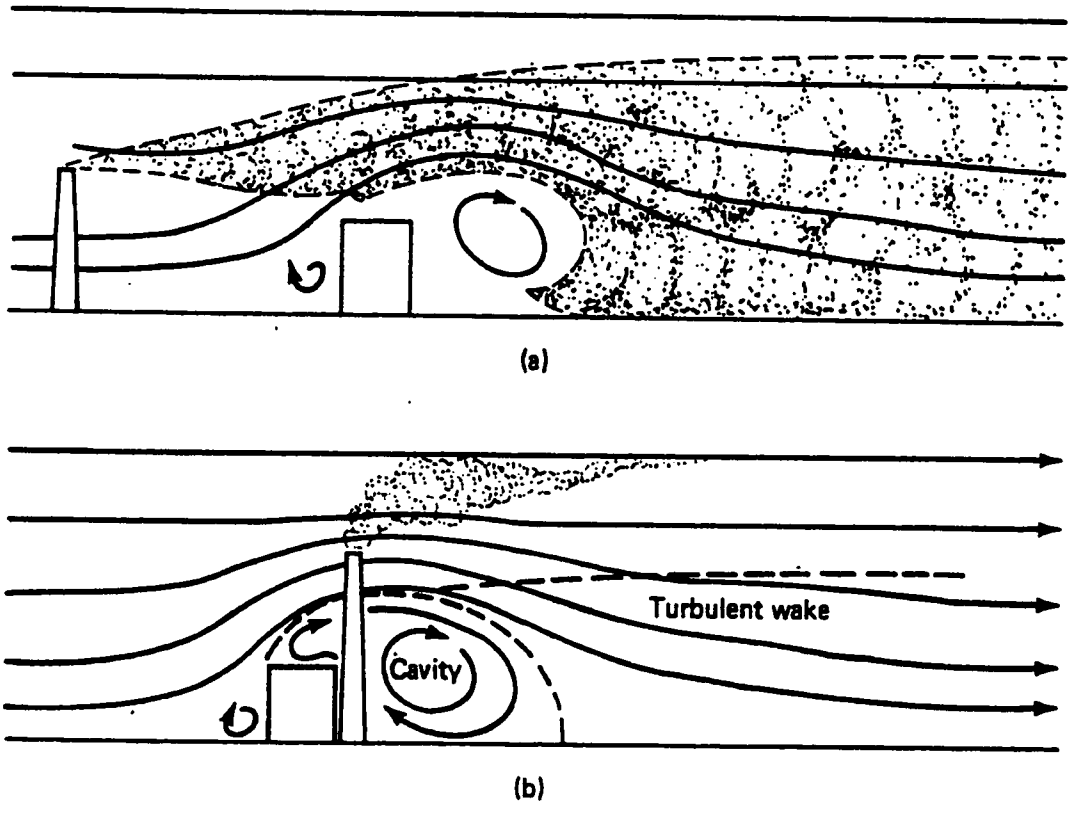


Figure 1.3 Effect of the location and height of a stack on dispersion around a building (From: Wark and Warner (1998))

Several other parameters can also influence the dispersion of pollutants around a building. These include wind directionality, wind turbulence intensity, exhaust momentum, surface roughness of the ground, buoyant effect of the pollutants and the character of the atmosphere (neutral, superadiabatic or subadiabatic).

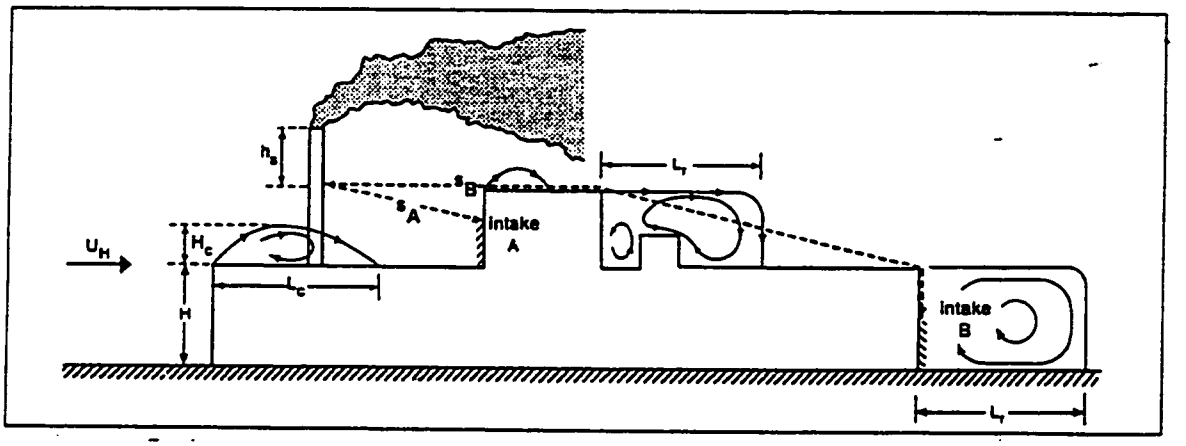


Figure 1.4 Typical pollutant dispersion around a flat-roof building

(From: ASHRAE Handbook-1997 Fundamentals 15.2)

1.3 Computational Wind Engineering

Traditionally the wind environmental conditions around buildings and wind-induced pressures on buildings have been studied using models of buildings in atmospheric boundary layer wind tunnels. The task of constructing a series of scale-model experiments to explore systematically the general 'model space' of a collection of buildings is laborious because of the multiplicity of configurations to be investigated. Furthermore, wind-tunnel experiments require resources of time and expertise which are often not directly available to architects and engineers.

A reliable computer simulation of wind flow around buildings can make a contribution to this problem by facilitating a less time-consuming exploration of the model space. In principle, a computer wind-flow simulation can make wind-related design information accessible to an architect at every stage of the design process. In contrast to the

increasing cost of performing experiments, this alternative becomes more and more encouraging considering that the relative cost of computation for a given algorithm and flow decreases by a factor of 10 every 8 years (Chapman, 1979). Computational Wind Engineering has thus been recently established as a new branch of Wind Engineering applying Computational Fluid Dynamics (CFD) to all classical wind engineering problems such as wind-induced pressures on buildings, dispersion of pollutants around buildings, etc.

The objective of the research described in this thesis is to attempt the evaluation of pollutant dispersion around buildings by CFD.

Chapter 2

LITERATURE REVIEW

2.1 Introduction

In this chapter previous studies in the area of dispersion of pollutants around buildings are reviewed. The conventional studies of air pollutant dispersion around buildings by theoretical or experimental methodology will be surveyed first. In order to simulate the dispersion field around buildings, the flow field must be simulated. Hence the simulation of flow field will also be investigated, followed by the review of numerical simulation of dispersion field. Particular attention will be paid to the estimation of numerical errors involved in such simulations.

2.2 Literature Survey

2.2.1 Theoretical and experimental study of air pollutant dispersion

The first attempt to predict the pollutant concentration by Gaussian diffusion theory was made by Halitsky (1963). This analytical method is easy to follow but it has serious deficiencies due to the foundation of the theory, which is based on the assumption that the pollutants are disposed from an isolated stack. Since the theory does not consider

the effect of the presence of buildings, the air pollution around a building cannot be predicted effectively.

Physical simulation in wind tunnels has also been attempted on buildings of certain geometry, followed by the derivation of empirical equations for the evaluation of pollutant concentration. Halitsky (1963) did several wind tunnel experiments involving various configurations including different building types, vent locations, velocities of the effluent at the aperture, wind speeds and directions, building surface temperature as well as effluent temperatures. Results were given by isopleths of concentration coefficient K, defined as

$$K = \frac{\chi A U}{Q} \qquad \text{.....(2-1)}$$

where χ is the concentration, Q is the gas release rate, U is the reference mean wind velocity and A is the area of the largest side of the building. The concentration coefficient can be applied to full scale configurations directly. His paper also gave a better curve fitting formula for dilution ratio D from all the experimental data.

Wilson and Britter (1982) made a contribution by considering different positions of pollutant sources such as: (1) upwind sources, (2) surface sources, (3) downwind sources in the near wake recirculation region and (4) short roof-mounted stacks. A design procedure for predicting building surface concentrations was developed, but the uncertainty was high for complex site or building configurations. Another study suggested that the dilution ratio method can be used only in relatively simple building

configurations. If the building configuration is such that the plume trajectory cannot be predicted reliably, a wind tunnel test is necessary. (Halitsky, 1982)

Li and Meroney (1983) investigated the dispersion of effluent plumes emitted from a cubical model building in the near-wake region ($x/H \leq 5.0$). The model study was performed in a wind tunnel with simulated neutrally stratified shear layers. A plexiglas model (5cm x 5cm x 5cm) was constructed to simulate a cubical building in the wind tunnel. Mean concentration measurements were made on the model building from three different roof vent locations and three different building orientations. They observed that concentration isopleths on a building surface would appear as closed continuous curves with their centers at the vent location unless intercepted by the presence of the ground. For equal vent exhaust to vent intake distance the mean concentrations decrease as the intake directions deviate from the wind direction. Orientation of the building at an angle of 45° results in a secondary peak in the crosswind ground-level concentration distribution. Orientation of the building at angles other than normal to the wind tends to increase the concentration level as a result of an enhanced downwash effect in the near-wake region.

Several studies by Huber and Snyder (1982), and Huber (1984, 1988 and 1991) tried to predict the pollutant dispersion around buildings based on the following Gaussian plume equation for estimating normalized concentration:

$$\frac{\chi U_r H^2}{Q} = \frac{\left[\exp(-\frac{1}{2}\frac{y^2}{\sigma_y^2})\right]\left\{\exp\left[-\frac{1}{2}\left(\frac{z-H_s}{\sigma_z}\right)^2\right]+\exp\left[-\frac{1}{2}\left(\frac{z+H_s}{\sigma_z}\right)^2\right]\right\}}{2\pi\sigma_z\sigma_y / H^2} \quad \ldots\ldots$$

.....(2-2)

Where,

χ= pollutant concentration (g m^{-3})

U_r= mean wind speed affecting the plume (m s^{-1})

Q= emission rate (g s^{-1})

H= building height (m)

H_s= plume centreline height (m)

σ_y = the standard deviation of the concentration distribution in the crosswind direction at the downwind distance x.

σ_z = the standard deviation of the concentration distribution in the vertical direction at the downwind distance x.

Figure 2.1 is a coordinate system showing Gaussian distributions in the horizontal and vertical direction.

Vertical and lateral concentration profiles for an isolated stack were measured at distances equal to 5, 10, 15 times building height downwind by Huber and Snyder (1982), and were used in characterizing dispersion in the simulated atmospheric boundary. Figure 2.2 and Figure 2.3 present the concentration profiles and their estimated Gaussian

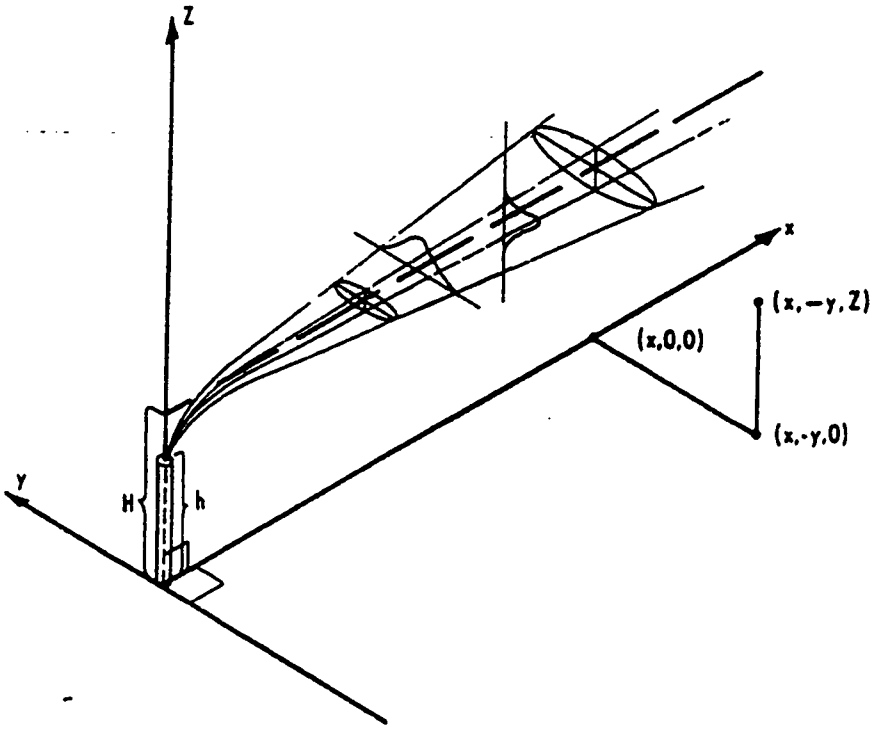


Figure 2.1 Coordinate system showing Gaussian distributions in the horizontal and vertical direction

distributions, expressed as equation (2-2). The standard deviations σ_y and σ_z were computed from the concentration distributions plotted on normal-probability paper and then slightly adjusted to provide a better fit to the lower half of the vertical profiles, since this was the region of greatest interest. The dispersion parameters in the wind tunnel were best described by:

$$\sigma_z / H = \sigma_y / H = 0.115(x/H)^{0.8} \quad \text{.....(2-3)}$$

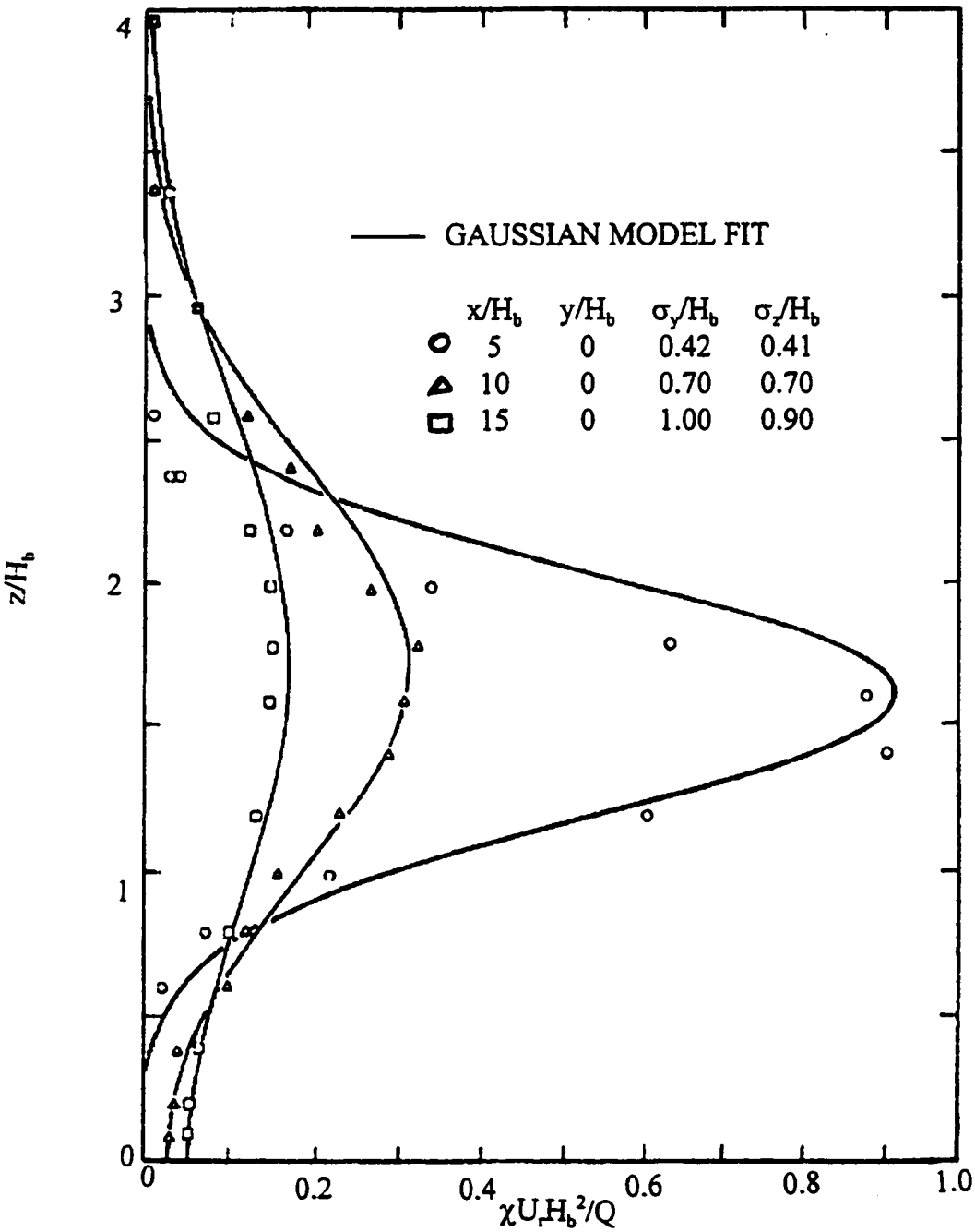


Figure 2.2 Vertical concentration profiles taken through the plume centerline for an isolated stack (From: Huber and Snyder (1982))

A rectangular-shaped building with its length equal to twice its height and width was oriented with the long side perpendicular to the approaching wind. The plume from a ground-level source was found to spread rapidly both in the vertical and lateral directions

in the wake of the building. Plumes released at or above 1.2 times the building height were similarly enhanced in the vertical but not so much in the lateral direction. The building influence was found to be reduced with increases in the effective source height.

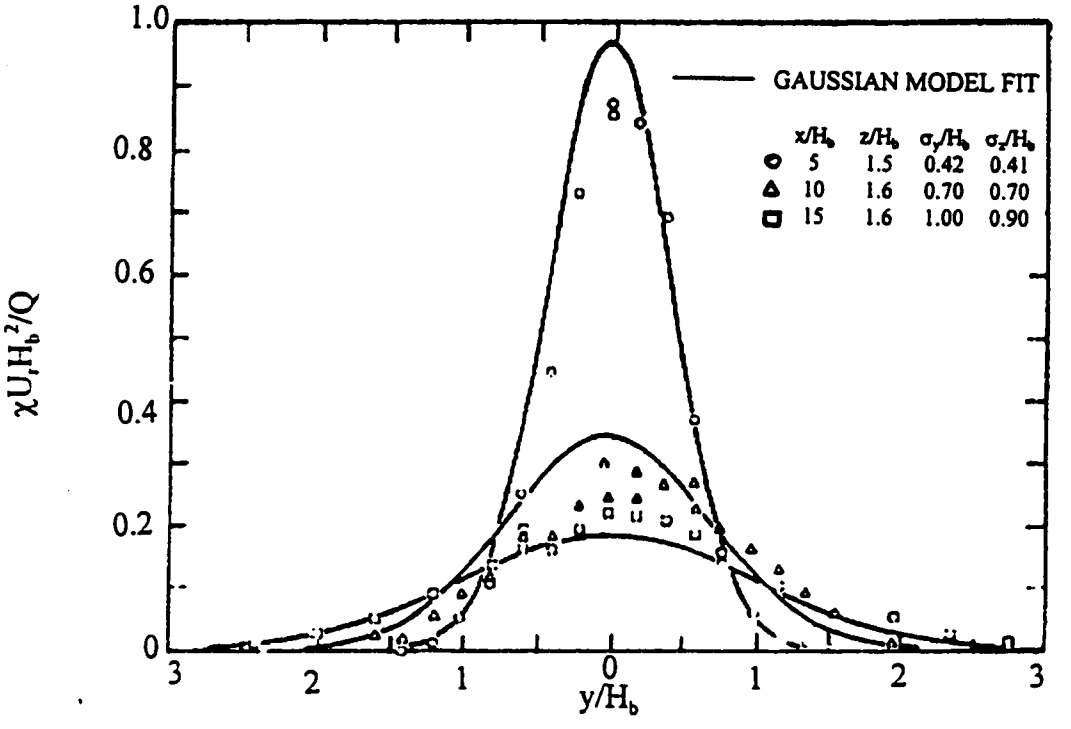


Figure 2.3 Lateral concentration profiles taken through the plume centerline for an isolated stack (From: Huber and Snyder (1982))

A simple mathematical model was formulated:

$$\frac{\sigma'}{H} = \left[C_1 + C_2 (x/H)^{C_3}\right]^{0.5} \frac{\sigma}{H}, for\, x/H \leq 10 \quad \text{.....(2-4)}$$

The constants C_1, C_2, C_3 were determined from the ground-level concentration measurements for the ground source. The resulting modified Gaussian plume model has the enhanced dispersion parameters, σ', (vertical and lateral),

$$\frac{\sigma'}{H} = \left[2 + 35\left(\frac{x}{H}\right)^{-1.8}\right]^{0.5}\left[0.115\left(\frac{x}{H}\right)\right]^{0.8} \quad \text{.....(2-5)}$$

$$\cong 0.7 + 0.067(x/H - 3), for 3 \le x/H \le 10 \quad \text{.....(2-6)}$$

For the concentrations in the far wake of the building, the dispersion parameter σ' could be determined by:

$$\frac{\sigma'}{H}(x/H) = \frac{\sigma}{H}(x/H + S) = 0.115(x/H + S)^{0.8} \quad \text{.....(2-7)}$$

Where S is a virtual source distance in building heights, a function of the background atmospheric conditions, which could also be determined by experiment.

This simple modified Gaussian plume model provided good estimates of concentration in the building wake. However, the model could not predict the dispersion near the building, (x/H<3 and z/H<2.5), as shown by Figure 2.4. Furthermore, it is believed to be applicable only to buildings with similar shapes to that used in the experiment.

Further study in the near wake region (i.e. 3-10 building heights) by Huber (1984) suggested that the effect of building wake enhanced dispersion on maximum ground-level concentrations is to decrease their values for releases near ground-level and to significantly increase their values for elevated releases for which enhanced horizontal dispersion dilutes the plume more rapidly. Enhanced vertical dispersion also rapidly dilutes the plume but it can also bring it to ground-level while in-plume concentrations are still high. The Gaussian plume equation has been modified to incorporate building

wake enhanced dispersion parameters in this near wake region, and the resulting ground-level plume centreline concentration estimates have been compared to the data of 10 sets of field measurements. The results indicate that this method can provide a good correction for the overall effect of adjacent buildings.

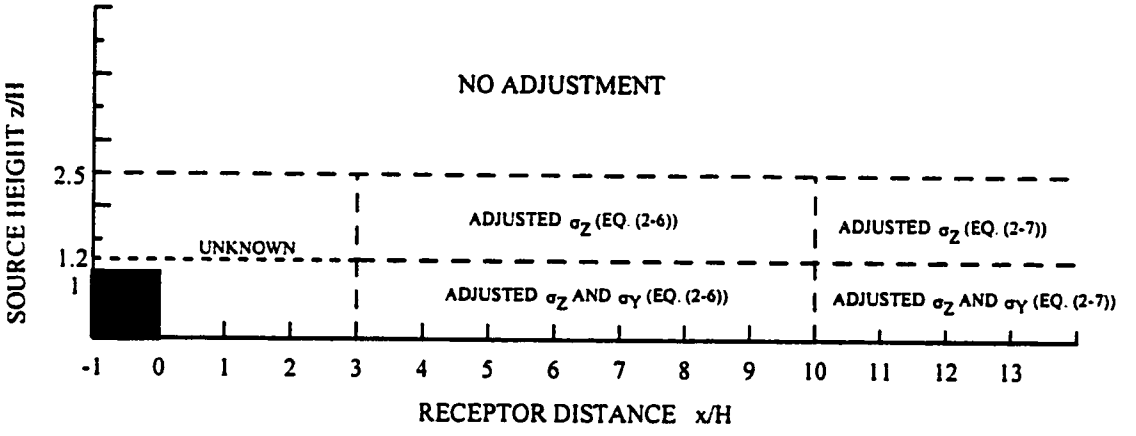


Figure 2.4 Suggested use for adjusted dispersion equations

(From: Huber and Snyder (1982))

Point comparisons between observed concentrations from field measurements and predicted concentrations from the modified Gaussian plume model by scatterplots were carried out by Huber (1988). The proposed model predicted from a factor of 3.5 lower to a factor of 2 higher than the observations.

Huber (1991) investigated the effect of boundary layer turbulence and building size on pollution concentrations near and downwind of an isolated rectangular building by conducting a series of wind tunnel measurements covering a range of four flow speeds

and four different sized buildings. Differences in observed velocity and concentrations between the results for the low-turbulence and simulated atmospheric boundary layer flow were found to be significant very near the building, but increased at downstream distances greater that 10 times the building height.

A wind tunnel study was conducted by Thompson (1991) to determine concentrations on building surfaces and at ground level from sources located on and above the roofs of four different rectangular building shapes. All of the buildings were 0.15 m high. Building 1 was a cube 0.15-m on a side. Building 2 was 0.3 m wide in the dimension perpendicular to the wind and 0.15 m long in the dimension aligned with the wind. Building 3 was 0.6 m wide and 0.15 m long, while building 4 was 0.15 m wide and 0.30 m long. Excessive concentrations were found for the wider buildings for stacks taller than the two-and-one-half-times rule suggests. The largest amplifications were found for releases just above the cavity of recirculating flow where the plume followed a streamline to the ground just downwind of the building.

A study addressing the validity of wind tunnel experiments for concentration measurements has been conducted by Saathoff et al (1994). This study, which examined the effects of mismatching model and boundary layer scales, found that the largest differences in concentration coefficient k were generally less than 20%. The study concluded that a four to one difference between model and boundary layer scales may not affect estimates of pollutant concentration on building surfaces significantly.

Previous experimental studies have shown the mechanism of the pollutant dispersion around buildings. A simple formula based on Gaussian plume has been derived and results have been compared with analytical predictions; the latter agree with wind tunnel or field measurement data generally within a factor of 2. However, the pollutant dispersion very near the building still remains unpredictable and the empirical formula can only be applied to buildings with similar shape.

2.2.2 Numerical simulation of flowfield

Based on the rapid development of computer facilities, it has become apparent that computer simulated fluid flow has a number of advantages compared to the wind tunnel tests. The computer-simulated results do not have the limitations of the physical simulation in the wind tunnel. Model scale and wind tunnel blockage effects are avoided. Changing parameters such as oncoming wind velocity, turbulence intensity, direction etc. becomes easier. In fact, several researchers have already predicted successfully the wind pressure on buildings and the flow field around buildings by using computer simulation and different turbulence models, at least for some particular cases.

At present, a commonly-used approach in computational wind engineering is to solve Reynolds-Averaged Navier-Stokes Equations (RANSE) with the k-ε model. This approach was followed for 3D rectangular buildings by Paterson & Apelt (1986), Mathews & Meyer (1987), Murakami & Mochida (1989) and Zhang (1994). The

prediction of surface mean pressures on cubes was generally good but the separation at corner and reverse flows on roof were poorly predicted. Baskaran & Stathopoulos (1989) used a modified k-ε model and zonal treatment near the boundary in order to minimize the discrepancy in corner separation areas. Zhou & Stathopoulos (1995) using a two-layer approach successfully predicted the flow separation at the roof corner with improved prediction of pressures.

The application of k-ε model on the wind flow around low-rise buildings has been attempted by Haggkvist (1989), Selvam & Paterson (1991), Selvam & Konduru (1992), Paterson & Holmes (1992), Richards & Hoxey (1992) etc. Generally, the predictions of mean pressures are good; however, strong suctions near the front corner are poorly predicted. The estimation of r.m.s. pressures was unsatisfactory; and mean pressure results were not in good agreement with the full scale data for some oblique wind directions. Stathopoulos & Zhou (1992), who applied the k-ε model on L-shaped buildings have achieved good agreement for normal wind directions but also found differences for oblique wind cases. Haggkvist (1989), Murakami & Mochida (1989), Baskaran & Stathopoulos (1989), Gadilhe et al. (1992), Takakura & Suyama (1992) and Yamamura & Kondo (1992) tried to predict the flow past building complexes with the k-ε model but only got qualitatively good results.

Recently, Tsuchiya et al. (1996) compared different modified versions of the standard k-ε model. They found that the Launder and Kato model (Launder and

Kato (1993), Kato and Launder (1993)) and Murakami-Mochida-Kondo model (Kondo et al. (1994)) provide better results than the standard k-ε model.

Lakehal and Rodi (1996) compared the simulation results of turbulent flow past a surface-mounted cubical obstacle placed in developed channel flow from standard k-ε model, the Kato-Launder modified k-ε model (Kato and Launder (1993)), the Norris-Reynolds two-layer model, the velocity based two-layer model and the Kato-Launder modified two-layer approach. Despite the simple geometry of the obstacle, the flow developing in its vicinity is very complex with multiple, unsteady separation regions, vortices of various kinds, strong curvature and adverse as well as favorable pressure gradients. Using the two-layer approach, the details of the complex flow structure near the ground wall including the converging-diverging behavior of the horse-shoe vortex can be resolved much better, the separation location in front of the cube is predicted correctly and also the prediction of the separation region on the roof is improved - when combined with the Kato-Launder modification the size of the roof separation bubble is predicted fairly well.

Bui and Oppenheim (1987) presented some simulated results for the air flow over a two-dimensional square model in a wind tunnel with a uniform approaching velocity using the Random Vortex Method (RVM). The main advantage of the RVM is to avoid the effect of numerical diffusion due to finite differentiation because it is a grid-free method. It is particularly attractive for the simulation of flows at high Reynolds numbers,

which occur in most of the building engineering applications. However the expansion of RVM for turbulent three-dimensional wind flow is not a trivial one.

Murakami et al. (1987), Murakami (1990) and Murakami et al. (1992) used Large Eddy Simulation (LES) to predict flows around a cubic building. The results are better than those with the k-ε model are and reverse flow at the front corner has been predicted. The vortex shedding behind a square cylinder can also be predicted very well with LES. However, a typical calculation of flow around a cube takes 100 CPU hours on a Fujitsu VP2600 (peak performance 2GFLOPS) machine.

He & Song (1992) applied the Weakly Compressible Flow concept with Large Eddy Simulation on a cubic building. The results are encouraging; however, about 3 hours on a Cray-2 (peak performance 100MFLOPS) were required for a typical calculation of flow around a cube.

2.2.3 Numerical simulation of air pollutant concentration field

Very few results of numerical simulation on air pollution around buildings have appeared in the literature. Dawson et al. (1991) reported that their computations for the dispersion from a building rooftop release show good agreement with wind tunnel measurements, except for locations very close to the ground. They also reported that their

numerical simulation of transport and dispersion of a plume over a 300-m conical hill was compared with near ground-level field measurement results.

Perhaps the most important study in this area was conducted by Zhang (1994). Her research investigated the effects of approach flow shear, turbulence and atmospheric stability on the flow and pollutant diffusion around buildings, especially in the wake of a building, using an existing computer code (TEMPEST). The study found that mean advection plays a more important role compared with turbulent diffusion in dispersing the effluent from a source located in the recirculating cavity region behind the building.

The numerical simulation of diffusion field around a cubic building by LES has been carried out by Murakami et al. (1991). They found that the result of the numerical simulation corresponds well to that of the wind-tunnel experiment for the pollutant discharged from a ground source downwind of the building.

2.2.4 Error estimation of computational results

Computational fluid dynamics has established itself as a viable technique for performing research and solving engineering problems. This technique has the potential to give accurate results for many fairly complex flow situations. Unfortunately, very little information is provided about the numerical uncertainty and the experimental data are often treated as if they were 100 percent accurate.

Meta (1991) indicated that uncertainties are inherent in computational fluid dynamics. He identified the sources of computational uncertainties as equivalence and numerical accuracy. The computational model needs to describe the "reality" contained in the theoretical (mathematical and/or empirical) model. A departure from equivalence of the two realities introduces errors. He also identified the three sources of uncertainty, which can influence the numerical accuracy, as follows: discretization, algorithm and presentation of computed results.

Celik (1993) advocated the need for implementing a policy regarding numerical uncertainty analysis. He indicated that the major topics of interest are: (i) separation of numerical errors from modelling errors (ii) identification, estimation, and reduction of numerical errors, (iii) assessment of computer codes and computational schemes with respect to numerical uncertainty-bench marking.

Celik and Zhang (1993) applied the Richardson extrapolation method to numerical simulations of turbulent, developing pipe flow, and turbulent recirculating flow over a backward facing-step. A commercial CFD code was used for this purpose. The application of the method cannot be used near singular points such as separation and reattachment points, neither can it be used in the immediate vicinity of the wall.

Demuren and Wilson (1993) estimated the numerical uncertainty in computations of two-dimensional backward facing step flows. They estimated the truncation error by comparing solutions from low and high-order schemes. The effect of outflow boundary

conditions was estimated by varying systematically the location of the outflow boundary without changing the grid distribution or the numerical scheme. The discretization error was estimated by making computations on related grids with varying degrees of density and using Richardson extrapolation method. The authors suggested that the uncertainty in computed results due to incomplete convergence of the iterative scheme can be removed by computing an estimate of the convergence error and using this as a stopping criterion rather than the more widely used change in computed results between iterates.

Ferziger (1993) thoroughly analysed and proposed the estimation method of convergence, discretization and grid generation errors. Methods converging after few iterations at low cost per iteration and adaptive grid methods are recommended to reduce the numerical error.

Roach (1993, 1994) proposed the use of a Grid Convergence Index (GCI) for the uniform reporting of grid refinement studies in Computational Fluid Dynamics.

Sbaibi and Manno (1993) tested different Total Variation Diminishing (TVD) schemes using a novel set of performance measures. These measures include absolute error integrated over the domain, the first through fourth moments of the computed and benchmark solutions, and solution "total variation". It was shown that this set of measures provides a comprehensive picture of algorithm performance and solution uncertainty.

Selvam and Huber (1995) provide a general review of the current status of computer modelling of pollutant dispersion around buildings. They found that the error involved in simulating the flow field and pollutant concentrations has not been investigated completely for any of the available methods. Some of the results compare well with wind-tunnel measurements but others are significantly higher than the wind tunnel data.

2.3 Justification of the Present Study

From the review it is clear that pollutant dispersion around buildings has acquired significant attention from industry. However, among all the previous research efforts, very few have used the methodology of numerical simulation. With the fast growth of computer technology, numerical simulation shows more and more promise as a faster and more economic tool in solving problems of pollutant dispersion around building. However, further research in this area is needed.

All the previously mentioned studies considered the standard k-ε turbulence model for the simulation of the flow field. Wall functions are necessary to relate surface boundary conditions to points in the fluid away from the boundaries and thereby avoid the problem of modeling the direct influence of viscosity. The validity of this procedure is restricted to situations in which the Reynolds number is sufficiently high for the viscous effects to be unimportant, which is not the case for most of the wind engineering applications where

separated flows exist. One drawback of this aberration from reality is the failure to predict the seperation region near the front corner of the building. Zhou and Stathopoulos (1995) found improved results in predicting the wind-induced pressure on building surface by using the Norris-Reynolds two-layer approach. The simulation of concentration field based on this two-layer approach may produce a better result than that from the standard k-ε model.

Most of the previous studies compared their results with experimental data to justify the numerical simulation method used without any consideration of numerical error. The experimental results are thought to be 100% correct and the numerical simulation error is thought as the difference between the experimental and the numerical simulation results, which usually is attributed to the physical model used in the simulation. However, Cowan et al. (1996) have shown that typical numerical solutions obtainable in an industrial context are likely to be strongly dependent not only on the turbulence model, but also, and often more importantly, on mesh design and the numerical method used. Stathopoulos (1996) provided an additional perspective by considering the broad scatter of results and the consequent difficulties associated with the general practice of validating a code by comparing numerical results with experimental data. Nevertheless, the analysis of numerical errors in computational wind engineering applications is an important aspect in the process of code validation. Though some of the previous studies carried out a sensitivity analysis to identify the influence of numerical treatment on the output of the system, no research seems to have been done to quantify the numerical error.

Chapter 3

NUMERICAL PROCEDURE

3.1 The Governing Equations

Fluid flow is governed by the law of conservation, i.e., the law of mass and momentum conservation, which results in the general governing equation of fluid flow, the Navier-Stokes equation:

$$\frac{\partial \tilde{U}_i}{\partial t} + \frac{\partial \tilde{U}_i \tilde{U}_j}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \tilde{P}}{\partial x_i} + \nu \frac{\partial^2 \tilde{U}_i}{\partial x_j x_j} \quad \text{.....(3-1)}$$

$$\frac{\partial \tilde{U}_i}{\partial x_i} = 0 \quad \text{.....(3-2)}$$

where, i,j=1,2,3 and repeated indices in any term imply summation.

It is customary to classify fluid flows in terms of the so-called Reynolds number, which is a nondimensional measure of the nonlinearity in equation (3-1) and is defined as Re=UL/ν, where U is a typical velocity and L is a typical length scale. If the Reynolds number is not too large, the flow will be laminar, in the sense that it will display regular and predictable variations in both space and time. As the Reynolds number increases, flows typically undergo a sequence of instabilities until, at some large enough value of the Reynolds number, they become fully turbulent. For turbulent flow, a description of

the flow at all points in time and space is very difficult. Instead of providing such a description, equations governing mean quantities, such as the mean velocity can be developed, following Reynolds (Rodi, 1980).

3.1.1 Reynolds-Averaged Navier-Stokes equations

In the present study, the steady state Reynolds-Averaged Navier-Stokes (RANS) equations and the k-ε turbulence model are adopted as governing equations of the turbulent air flow around buildings. These equations are as follows:

The continuity equation:

$$\frac{\partial U_j}{\partial x_j} = 0 \qquad \text{.....(3-3)}$$

The momentum equations:

$$U_j \frac{\partial U_i}{\partial x_j} = -\frac{\partial P}{\partial x_i} + \frac{\partial}{\partial x_j}\left[(\nu + \nu_t)\frac{\partial U_i}{\partial x_j}\right] \qquad \text{.....(3-4)}$$

The k-equation:

$$U_j \frac{\partial k}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\left(\nu + \frac{\nu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] + \nu_t\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right)\frac{\partial U_i}{\partial x_j} - \varepsilon \qquad \text{.....(3-5)}$$

The ε-equation:

$$U_j \frac{\partial \varepsilon}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\left(\nu + \frac{\nu_t}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right] + C_1 \frac{\varepsilon}{k}\nu_t\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right)\frac{\partial U_i}{\partial x_j} - C_2 \frac{\varepsilon^2}{k} \qquad \text{.....(3-6)}$$

where P is the augmented pressure defined by:

$$P = \frac{p}{\rho} + \frac{2}{3}k \quad \text{.....(3-7)}$$

and ν_t is the turbulence eddy viscosity:

$$\nu_t = C_\mu \frac{k^2}{\varepsilon} \quad \text{.....(3-8)}$$

The model constants are as follows:

$C_{1\varepsilon}$	$C_{2\varepsilon}$	C_μ	σ_k	σ_ε
1.44	1.92	0.09	1.0	1.3

3.1.2 Concentration equation

From the conservation of contaminant species, we can derive the following equation:

$$\frac{\partial \tilde{C}}{\partial t} + \frac{\partial \tilde{C}\tilde{U}_i}{\partial x_i} = \frac{\partial}{\partial x_i}\left(D\frac{\partial \tilde{C}}{\partial x_i}\right) + S \quad \text{.....(3-9)}$$

Similar to the derivation of equation (3-4), the concentration can be divided into a mean part and a fluctuating part. Equation (3-9) is averaged to produce the equation for mean concentration. The turbulent concentration flux term can be approximated by assuming gradient transport and constant Prandtl number which leads to equation (3-10):

$$-\overline{c'u'_i} = \frac{\nu_t}{\sigma_c}\left(\frac{\partial C}{\partial x_i}\right) \quad \text{.....(3-10)}$$

where c' is the fluctuating concentration, u_i' is the fluctuating velocity, C is the mean concentration and σ_c is the turbulent Prandtl number for C. We assume $\sigma_c = 0.7$ in the

present simulation, according to Spalding (1971). By using equation (3-10), the equation for mean concentration can be approximated as follows:

$$U_i \frac{\partial C}{\partial x_i} = \frac{\partial}{\partial x_i}\left(\frac{\nu_t}{\sigma_c}\frac{\partial C}{\partial x_i}\right) + S \quad \text{.....(3-11)}$$

3.1.3 Equations for the two-layer method

As stated in the previous chapter, studies by Murakami et. al. (1990) and Stathopoulos & Zhou (1992) have shown that the standard k-ε model can not simulate the flow separation on the building roof. A two-layer method to simulate the wind conditions around a cubic building was adopted by Zhou & Stathopoulos (1995). In this two-layer approach, the computational region was divided into two layers, namely inner region and external region. The wind flow in all external region is computed with the standard k-ε model, whereas the flow near building surfaces (inner regions) is simulated with a near wall model. In the present study, the one-equation model proposed by Norris and Reynolds (1975) has been adopted. In this model, near the solid wall, inside the inner layer, the k-equation and ε-equation are combined as:

$$U_j \frac{\partial k}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\left(\nu + \frac{\nu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] + \nu_t\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right)\frac{\partial U_i}{\partial x_j} - \frac{k^{3/2}}{L}\left(1 + \frac{C_\varepsilon}{k^{1/2}L/\nu}\right) \quad \text{.....(3-12)}$$

where L is a length-scale defined by the formula:

$$L = C_D \kappa d \quad \text{.....(3-13)}$$

where the constant are as follows:

C_ε	C_D	κ
13.2	6.41	0.41

The eddy viscosity is given by the following equation:

$$\nu_t = C_\mu f_\mu k^{1/2} L \quad \text{.....(3-14)}$$

where ε appears as the length scale L and

$$f_\mu = 1 - \exp(-0.0198 R_d) \quad \text{.....(3-15)}$$

is a damping function, in which $R_d=dk^{1/2}/\nu$ is the turbulence Reynolds number and d is the distance from the solid wall.

3.2 The Discretization Method

A Semi-Implicit Method for Pressure-Linked Equations (SIMPLE) is used in solving the flow field (Patankar, 1980). In this approach, a control volume method with hybrid scheme has been used for discretizing the momentum equations, k-equation and ε-equation. The computation procedure is as follows:

3.2.1 Flow field

i) start from the initial pressure (P^*), eddy viscosity (ν_t^*), velocity field (U_i^*), turbulence kinetic energy (k^*) and its dissipation (ε^*).

ii) use the pressure field, ν_t field and the velocities obtained from the previous iteration P^*, ν_t^*, and U^*, V^* and W^*, to solve the momentum equations to obtain U^*, V^* and W^*.

iii) use continuity equation to get the pressure correction P'.

iv) add P' to P^* to get new pressure field P

v) use P' to get new velocity field

vi) solve k- and ε- equations to get k and ε

vii) calculate new ν_t by new k and ε

viii) treat the corrected pressure field (P) as guessed pressure (P^*), new k and ε as k^* and ε^* and new eddy viscosity (ν_t) as ν_t^*, new velocity field as U^*, V^* and W^* for next iteration, return to step ii).

3.2.2 Concentration equation --- explicit method

The concentration equation (Equation 3-11) was presented in section 3.1.2. Using time-marching method, the equation can be written as:

$$\frac{\partial C}{\partial t} = -U_i \frac{\partial C}{\partial x_i} + \frac{\partial}{\partial x_i}\left(\frac{\nu_t}{\sigma_c}\frac{\partial C}{\partial x_i}\right) + S$$

$$= -\frac{\partial}{\partial x}\left(UC - \frac{\nu_t}{\sigma_c}\frac{\partial C}{\partial x}\right) - \frac{\partial}{\partial y}\left(VC - \frac{\nu_t}{\sigma_c}\frac{\partial C}{\partial y}\right) - \frac{\partial}{\partial z}\left(WC - \frac{\nu_t}{\sigma_c}\frac{\partial C}{\partial z}\right) + S \quad \text{.....(3-16)}$$

and be discretized at point (i,j,k).

The term at left hand side can be discretized by first-order forward difference:

$$\frac{\partial C}{\partial t}\Big|_{i,j,k} = \frac{1}{\Delta t}(C^{n+1}_{i,j,k} - C^{n}_{i,j,k}) \quad \text{.....(3-17)}$$

Define:

$$\Gamma = \frac{\nu_t}{\sigma_c} \quad \text{.....(3-18)}$$

the terms at right hand side of equation (3-16) can be discretized by central difference as follows:

$$-\frac{\partial}{\partial x}(UC - \Gamma\frac{\partial C}{\partial x})\Big|_{i,j,k} = -\frac{1}{\Delta x_{i,j,k}}\left[(UC - \Gamma\frac{\partial C}{\partial x})_{i+1/2,j,k} - (UC - \Gamma\frac{\partial C}{\partial x})_{i-1/2,j,k}\right] \quad \text{.....(3-19)}$$
$$= -\frac{1}{\Delta x_{i,j,k}}\left[XFLUX_{i+1/2,j,k} - XFLUX_{i-1/2,j,k}\right]$$

$$-\frac{\partial}{\partial y}(VC - \Gamma\frac{\partial C}{\partial y})\Big|_{i,j,k} = -\frac{1}{\Delta y_{i,j,k}}\left[(VC - \Gamma\frac{\partial C}{\partial y})_{i,j+1/2,k} - (VC - \Gamma\frac{\partial C}{\partial y})_{i,j-1/2,k}\right] \quad \text{.....(3-20)}$$
$$= -\frac{1}{\Delta y_{i,j,k}}\left[YFLUX_{i,j+1/2,k} - YFLUX_{i,j-1/2,k}\right]$$

$$-\frac{\partial}{\partial z}(WC - \Gamma\frac{\partial C}{\partial z})\Big|_{i,j,k} = -\frac{1}{\Delta z_{i,j,k}}\left[(WC - \Gamma\frac{\partial C}{\partial z})_{i,j,k+1/2} - (WC - \Gamma\frac{\partial C}{\partial z})_{i,j,k-1/2}\right] \quad \text{.....(3-21)}$$
$$= -\frac{1}{\Delta z_{i,j,k}}\left[ZFLUX_{i,j,k+1/2} - ZFLUX_{i,j,k-1/2}\right]$$

The terms XFLUX, YFLUX and ZFLUX in equations (3-19), (3-20) and (3-21) are pollutants flux due to convection and diffusion across the control volume surfaces in x, y and z directions respectively. These terms could be approximated by the central difference method, upwind method, hybrid method and power-law method as follows:

(a) Central difference Scheme:

$$XFLUX_{i+1/2,j,k} = U_{i+1/2,j,k}\frac{C_{i+1,j,k}+C_{i,j,k}}{2} - \Gamma_{i+1/2,j,k}\frac{C_{i+1,j,k}-C_{i,j,k}}{\Delta x_{i+1/2}}$$

$$= U_{i+1/2,j,k}C_{i,j,k} + \left(C_{i,j,k}-C_{i+1,j,k}\right)\left(\frac{\Gamma_{i+1/2,j,k}}{\Delta x_{i+1/2}} - \frac{U_{i+1/2,j,k}}{2}\right) \quad \text{.....(3-22)}$$

$$= U_{i+1/2,j,k}C_{i,j,k} + a_X\left(C_{i,j,k}-C_{i+1,j,k}\right)$$

where

$$a_X = \frac{\Gamma_{i+1/2,j,k}}{\Delta x_{i+1/2}} - \frac{U_{i+1/2,j,k}}{2} \quad \text{.....(3-23)}$$

For the same reason:

$$YFLUX_{i,j+1/2,k} = V_{i,j+1/2,k}C_{i,j,k} + a_Y\left(C_{i,j,k}-C_{i,j+1,k}\right) \quad \text{.....(3-24)}$$

where

$$a_Y = \frac{\Gamma_{i,j+1/2,k}}{\Delta y_{j+1/2}} - \frac{V_{i,j+1/2,k}}{2} \quad \text{.....(3-25)}$$

and

$$ZFLUX_{i,j,k+1/2} = W_{i,j,k+1/2}C_{i,j,k} + a_Z\left(C_{i,j,k}-C_{i,j,k+1}\right) \quad \text{.....(3-26)}$$

where

$$a_Z = \frac{\Gamma_{i,j,k+1/2}}{\Delta z_{k+1/2}} - \frac{W_{i,j,k+1/2}}{2} \quad \text{.....(3-27)}$$

(b) Upwind Scheme:

For the case of upwind method, the value of C is equal to the value of C at the grid point on the upwind side of the face.

Thus,

if $U_{i+1/2,j,k} > 0$

$$(UC)_{i+1/2,j,k} = U_{i+1/2,j,k} C_{i,j,k} \quad \text{.....(3-28)}$$

if $U_{i+1/2,j,k} > 0$

$$(UC)_{i+1/2,j,k} = U_{i+1/2,j,k} C_{i+1,j,k} \quad \text{.....(3-29)}$$

If equation (3-28) and (3-29) is applied to XFLUX, YFLUX and ZFLUX, the same expression of the discretized equations (3-22), (3-24) and (3-26) is obtained, where

$$a_X = \frac{\Gamma_{i+1/2,j,k}}{\Delta x_{i+1/2}} + \left\| -U_{i+1/2,j,k}, 0 \right\| \quad \text{.....(3-30)}$$

$$a_Y = \frac{\Gamma_{i,j+1/2,k}}{\Delta y_{j+1/2}} + \left\| -V_{i,j+1/2,k}, 0 \right\| \quad \text{.....(3-31)}$$

$$a_Z = \frac{\Gamma_{i,j,k+1/2}}{\Delta z_{k+1/2}} + \left\| -W_{i,j,k+1/2}, 0 \right\| \quad \text{.....(3-32)}$$

where ||..|| means max(..).

(c) Hybrid Scheme:

Based on the ideas of the well-known hybrid scheme (Patankar,1980), the present simulation employs a central/upwind combination scheme and switches from one scheme to another according to the local Peclet number in order to minimize the disadvantage in each scheme. The coefficients a_X, a_Y, a_Z are derived as follows:

$$a_X = \left\| -U_{i+1/2,j,k}, \frac{\Gamma_{i+1/2,j,k}}{\Delta x_{i+1/2}} - \frac{U_{i+1/2,j,k}}{2}, 0 \right\| \quad \text{.....(3-33)}$$

$$a_Y = \left\| -V_{i,j+1/2,k}, \frac{\Gamma_{i,j+1/2,k}}{\Delta y_{j+1/2}} - \frac{V_{i,j+1/2,k}}{2}, 0 \right\| \quad \text{.....(3-34)}$$

$$a_Z = \| -W_{i,j,k+1/2}, \frac{\Gamma_{i,j,k+1/2}}{\Delta z_{k+1/2}} - \frac{W_{i,j,k+1/2}}{2}, 0 \| \quad \text{.....(3-35)}$$

It is an easy task to prove that this formulation becomes identical to the central difference scheme (equations (3-23), (3-25) (3-27)) when the local grid Peclet number (for instance, $Pe=U_{i+1/2,j,k}\Delta x_{i+1/2} \ / \ \Gamma_{i+1/2,j,k}$ along x direction) satisfies the condition $-2 \le Pe \le 2$. Outside this range, this method reduces to an upwind interpolated scheme without the diffusion/viscosity term.

(d) Power-Law Scheme:

The Power-Law scheme due to Patankar (1980) can also be applied to the present simulation with coefficients a_X, a_Y, a_Z defined as follows:

$$a_X = \frac{\Gamma_{i+1/2,j,k}}{\Delta x_{i+1/2}} \| 0, (1 - \frac{0.1 | U_{i+1/2,j,k} |}{\Gamma_{i+1/2,j,k}} \Delta x_{i+1/2})^5 \| + \| 0, -U_{i+1/2,j,k} \| \quad \text{.....(3-36)}$$

$$a_Y = \frac{\Gamma_{i,j+1/2,k}}{\Delta y_{j+1/2}} \| 0, (1 - \frac{0.1 | V_{i,j+1/2,k} |}{\Gamma_{i,j+1/2,k}} \Delta y_{j+1/2})^5 \| + \| 0, -V_{i,j+1/2,k} \| \quad \text{.....(3-37)}$$

$$a_Z = \frac{\Gamma_{i,j,k+1/2}}{\Delta z_{k+1/2}} \| 0, (1 - \frac{0.1 | W_{i,j,k+1/2} |}{\Gamma_{i,j,k+1/2}} \Delta z_{k+1/2})^5 \| + \| 0, -W_{i,j,k+1/2} \| \quad \text{.....(3-38)}$$

Comparing these expressions with equations (3-23), (3-25), (3-27), it can be found that, for $|Pe|>10$, the power-law scheme becomes identical to the hybrid scheme.

3.2.3 Concentration equation --- implicit method

Since the calculated flow field must satisfy the continuity equation (3-3), the concentration equation (3-11) can also be written as:

$$\frac{\partial(U_i C)}{\partial x_i} = \frac{\partial}{\partial x_i}\left(\Gamma \frac{\partial C}{\partial x_i}\right) + S \quad \text{.....(3-39)}$$

where Γ is defined in equation (3-18).

Equation (3-39) can be discretized as:

$$a_P \phi_P = a_E \phi_E + a_W \phi_W + a_N \phi_N + a_S \phi_S + a_T \phi_T + a_B \phi_B + b \quad \text{.....(3-40)}$$

where ϕ represents C,

$$a_E = D_e A(|P_e|) + \|-F_e, 0\| \quad \text{.....(3-41)}$$

$$a_W = D_w A(|P_w|) + \|-F_w, 0\| \quad \text{.....(3-42)}$$

$$a_N = D_n A(|P_n|) + \|-F_n, 0\| \quad \text{.....(3-43)}$$

$$a_S = D_s A(|P_s|) + \|-F_s, 0\| \quad \text{.....(3-44)}$$

$$a_T = D_t A(|P_t|) + \|-F_t, 0\| \quad \text{.....(3-45)}$$

$$a_B = D_b A(|P_b|) + \|-F_b, 0\| \quad \text{.....(3-46)}$$

$$b = S_C \Delta x \Delta y \Delta z \quad \text{.....(3-47)}$$

$$a_P = a_E + a_W + a_N + a_S + a_T + a_B - S_C \Delta x \Delta y \Delta z \quad \text{.....(3-48)}$$

The subscripts P, E, W, N, S, T and B denote the values at the grid points whereas the subscripts e, w, n, s, t and b denote the values at the interface as shown in Figure 3.1 and Figure 3.2.

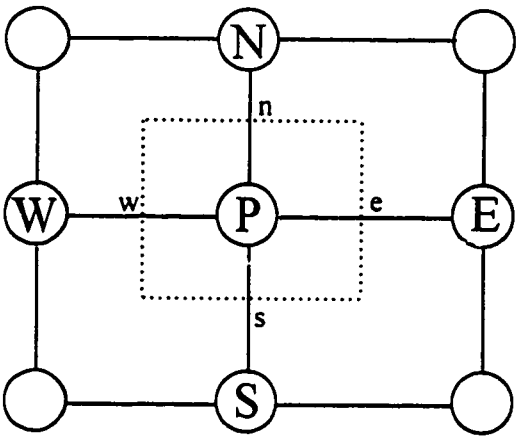


Figure 3.1 Nodes and interfaces in x-y plane

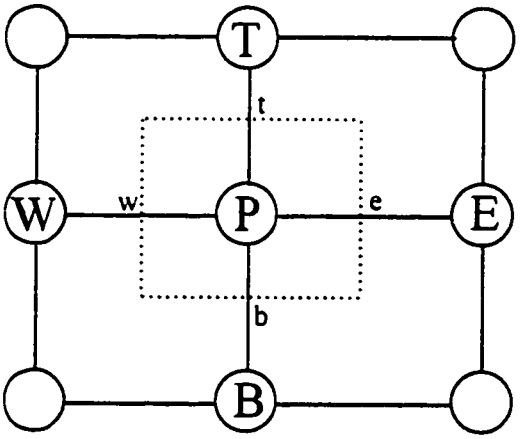


Figure 3.2 Nodes and interfaces in x-z plane

The flow rates and conductances are defined as

$$F_e = u_e \Delta y \Delta z \quad \text{.....(3-49)}$$

$$F_w = u_w \Delta y \Delta z \quad \text{.....(3-50)}$$

$$F_n = u_n \Delta y \Delta z \quad \text{.....(3-51)}$$

$$F_s = u_s \Delta y \Delta z \quad \text{.....(3-52)}$$

$$F_t = u_t \Delta y \Delta z \quad \text{(3-53)}$$

$$F_b = u_b \Delta y \Delta z \quad \text{(3-54)}$$

$$D_e = \frac{\Gamma_e \Delta y \Delta z}{(\delta x)_e} \quad \text{(3-55)}$$

$$D_w = \frac{\Gamma_w \Delta y \Delta z}{(\delta x)_w} \quad \text{(3-56)}$$

$$D_n = \frac{\Gamma_n \Delta y \Delta z}{(\delta x)_n} \quad \text{(3-57)}$$

$$D_s = \frac{\Gamma_s \Delta y \Delta z}{(\delta x)_s} \quad \text{(3-58)}$$

$$D_t = \frac{\Gamma_t \Delta y \Delta z}{(\delta x)_t} \quad \text{(3-59)}$$

$$D_b = \frac{\Gamma_b \Delta y \Delta z}{(\delta x)_b} \quad \text{(3-60)}$$

The Peclet number P_e is to be taken as the ratio of F and D; thus, $P_e = F_e/D_e$, and so on.

The function A(|P|) is as follows (Patankar, 1980):

Central difference:

$$A(|P|) = 1 - 0.5|P| \quad \text{(3-61)}$$

Upwind:

$$A(|P|) = 1 \quad \text{(3-62)}$$

Hybrid:

$$A(|P|) = \|0, 1 - 0.5|P|\| \quad \text{(3-63)}$$

Power law:

$$A(|P|) = \|0, (1 - 0.1|P|)^5\| \quad \text{.....(3-64)}$$

Appendix A includes the source code for the solution of the concentration equation (3-39) with the implementation of this method.

3.3 Grid Generation

A non-uniform staggered rectangular grid system is used in the present study. The distance between the first grid line and its adjacent solid boundary is defined by the user. Then an expanding factor can be calculated by the number of grid nodes and the computational domain size of a specific region. The successive grid lines are thus generated by this expanding factor. This procedure is repeated for each region with a solid boundary.

The use of staggered grid can avoid the wavy pattern of pressure and velocity without providing special treatment at the boundaries, or, over-specifying the boundary conditions (Patankar, 1980).

The use of non-uniform grids can optimize the utilization of computer resources because the high density grid nodes are only used in the region where they are really

required, i.e. near the building envelope and near the ground where the gradient of the parameters is generally large.

3.4 Boundary Conditions

The governing equations are basically a group of partial differential equations. Appropriate boundary conditions are required in order to solve these equations. These are usually expressed as conditions on the velocities, turbulent kinetic energy and turbulent kinetic energy dissipation.

In the staggered grid approach, the boundary location is designed so as to coincide with the face of the control volume. The velocity component normal to the boundary is, therefore, located just on the boundary, while the lateral component and other scalar variables are defined at a half cell interval away from the boundary.

3.4.1 Inlet condition of velocity

The custom power law velocity profile is used for the oncoming flow, which is

$$\begin{aligned} U(x,y,z) &= U_g\left(\frac{z}{z_g}\right)^{\alpha} \\ V(x,y,z) &= 0 \\ W(x,y,z) &= 0 \end{aligned} \qquad \text{.....(3-65)}$$

where U,V and W are the mean flow velocity components along x, y and z directions; U_g is the gradient velocity at gradient height z_g; and α is the power law component.

3.4.2 Inlet condition of turbulence properties

Inlet conditions of k and ε are considered the same as those of the oncoming flow, which is the flow undisturbed by the building. This flow is the so-called atmospheric boundary layer flow. Take the oncoming flow direction as x direction, then V=0, W=0 as indicated in 3.4.1. All the variations along x and y directions are also supposed to be zero,

i.e. $\frac{\partial}{\partial x} = 0, \quad \frac{\partial}{\partial y} = 0$.

By considering all the previous assumptions, the simplified k-equation and ε-equation can be obtained as follows:

k-equation:

$$\frac{\partial}{\partial z}\left(\frac{\nu_t}{\sigma_k}\frac{\partial k}{\partial z}\right) + \nu_t\left(\frac{\partial U}{\partial z}\right)^2 - \varepsilon = 0 \quad \text{.....(3-66)}$$

ε-equation:

$$\frac{\partial}{\partial z}\left(\frac{\nu_t}{\sigma_\varepsilon}\frac{\partial \varepsilon}{\partial z}\right) + C_1 C_\mu k\left(\frac{\partial U}{\partial z}\right)^2 - C_2\frac{\varepsilon^2}{k} = 0 \quad \text{.....(3-67)}$$

The two equations can be numerically solved with the inlet velocity profile mentioned in 3.4.1.

3.4.3 Outlet boundary conditions

It is assumed that the computation domain is large enough compared with the building. So the outlet is assumed far enough from the building so that all the variables do not change along the x direction, i.e. $\frac{\partial}{\partial x} = 0$

3.4.4 Side and top of the computational domain

Due to the same reason as 3.4.3

At the sides of the computational domain

$$\frac{\partial}{\partial y} = 0$$

At the top of the computational domain

$$\frac{\partial}{\partial z} = 0$$

3.4.5 Solid boundaries

The solid boundaries include the ground, the front wall of the building, two side walls of the building, the building roof and the leeward wall of the building. A log-linear-law wall function was used to calculate velocity at the first grid away from the wall. For a solid wall along the x_i direction,

$$U_i = \frac{(\frac{\tau_i}{\rho})\ln(EY^+)}{C_\mu^{1/4} k^{1/2} \kappa} \qquad for \quad Y^+ > 11.6 \quad \text{.....(3-68)}$$

and

$$U_i = \frac{(\frac{\tau_i}{\rho})Y^+}{C_\mu^{1/4} k^{1/2}} \qquad for \quad Y^+ \leq 11.6 \quad \text{.....(3-69)}$$

where U_i represents the component of tangential velocity along the x_i direction; τ_i is the corresponding shear stress on the wall; and Y^* is the dimensionless distance from the wall to the grid node:

$$Y^* = \frac{C_\mu^{1/4} k^{1/2} d}{\nu} \quad \text{.....(3-70)}$$

in which d is the distance from the grid node to the wall.

3.4.6 Zonal treatment of k and ε

Inside a turbulent boundary layer, the random motion of the fluid particles must die out very close to the solid boundary in order to maintain the condition of no slip at the solid-fluid interface. To accommodate this, the presence of a viscous sub-layer in the turbulent region has been established. It is reasonable to assume that the kinetic energy k and its dissipation ε have different behavior in the turbulent region and inside the sub-

layer. The zonal treatment method by Stathopoulos and Baskaran (1990) calculated the turbulent kinetic energy k and its dissipation ε by the following formula:

$$k_s = k_{es}\frac{d_s^2}{d_{es}^2} \qquad \text{.....(3-71)}$$

and

$$\varepsilon_s = \frac{2\nu k_{es}}{d_{es}^2} \qquad \text{.....(3-72)}$$

where k_{es} and k_s are the turbulence kinetic energy at the edge and within the laminar sub-layer respectively considered at the distances d_{es} and d_s from the solid boundary, ε_s is the energy dissipation at the distance d_s from the wall and ν is the fluid viscosity.

3.5 Convergence Criteria

In numerically solving non-linear equations, the estimated values of unknowns are used to calculate the coefficients in the discretization equations at every step in order to get new estimated values of unknowns. This process is repeated until further repetitions (iterations) cease to produce any significant changes in the estimated values of unknowns. The final unchanging state is called the convergence of the iterations. A common criterion of convergence is to evaluate the relative changes in the unknowns between two successive iterations until it is smaller than a certain small number. However, when heavy underrelaxation is used, the change in the dependent variables between successive iterations is intentionally slowed down, which may create an illusion of convergence although the computed solution may be far from being converged. In the present study,

the residue convergence criterion is used. Also the convergence criterion suggested by Ferziger (1993) have been studied.

3.5.1 Difference of iterates

The usually-stated criterion of iteration convergence, say for a solution ϕ, is one where the entire computational domain is checked for

$$\max|\phi^{n+1} - \phi^{n}| < \varepsilon \quad \text{.....(3-73)}$$

-- see Roache (1976). Such a criterion can be met at any stage for any ε, simply by selecting a small enough Δt or an underrelaxation factor. In order to consider the effect of Δt, $D = |\frac{\phi^{n+1} - \phi^{n}}{\Delta t}|$ is considered for each grid point

The difference of iterates for the entire computational domain is taken as the maximum D from all grid points.

3.5.2 Relative residue

The residue is the difference between two sides of the discretization equation after the computed value is substituted into the equation for each grid point. It shows how well the discretizations are satisfied by the current values of the dependent variables. In the present computation, the sum of absolute values of residues over the computational region is defined as the residue |R| for each iteration. The relative residue is defined as the ratio of the current residue $|R^n|$ to the residue after the first iteration $|R^1|$, i.e.:

$$r_n = \left|R^n\right| / \left|R^1\right| \quad \text{.....(3-74)}$$

According to Doormaal and Raithby (1984), the optimal values of relative residues are typically from 0.25 to 0.05. In the present study, the computation is considered convergent if the relative residues of all equations are less than 0.1

3.5.3 Ferziger's convergence criterion

Ferziger (1993) proposed a different convergence criterion. For a linear system, a principal eigenvalue can be estimated at each iteration step as

$$\lambda_1 = \frac{||\phi^{n+1} - \phi^n||}{||\phi^n - \phi^{n-1}||} \quad \text{.....(3-75)}$$

where ϕ^{n-1}, ϕ^n and ϕ^{n+1} represent the values of solution ϕ at three successive iteration steps and $||Q||$ represents the norm of the quantity Q ; a convenient norm is the root mean square of Q.

The error at each iteration step is then estimated by:

$$\varepsilon^n = \frac{\phi^{n+1} - \phi^n}{\lambda_1 - 1} \quad \text{.....(3-76)}$$

However, since in the present study, the eigenvalue λ_1 was found close to unity, which means that the difference between two successive iterations is so small that it is not appropriate to use this convergence criterion.

3.6 Computational Codes

As previously mentioned (section 3.2), Appendix A shows the Fortran source code of predicting wind-induced pollutant dispersion around buildings by an implicit method. The implementation of an explicit method in solving the concentration equation (section 3.2.2) is displayed in Appendix B. Both of these codes were written by the author during the course of this research project.

Appendix C is the Fortran source code for simulating the flow field around buildings by standard k-ε model which was developed in Centre for Building Studies at Concordia University ((Baskaran & Stathopoulos, 1989), (Stathopoulos & Baskaran, 1991), (Stathopoulos & Zhou, 1993)). Appendix D gives the Fortran source code for predicting the flow field around buildings by a two-layer approach which is also developed in Centre for Building Studies at Concordia University (Zhou & Stathopoulos, 1996). These codes were used as the tools to get the simulated flow field around buildings. They had been tested, verified, documented and modified to apply the convergence criterion of Ferziger's (section 3.5.3) during the present research.

Chapter 4

FLOW FIELD AND CONCENTRATION FIELD AROUND A CUBIC BUILDING

4.1 Introduction

When the atmospheric boundary flow approaches the cubic building, it decelerates along the wind direction and accelerates in the other two directions in order to pass the building. A positive pressure zone is created on the front wall of the building with a stagnation point appeared at about 2/3 of the building height because of the shear in the approaching atmospheric boundary layer. The wind above the stagnation point flows upward while the wind below the stagnation point flows downward. When the downward flow hit the ground, a horse-shoe vortex is formed at the base of the building which will be forced around the side of the building and trails off downwind, as shown in Figure 1.1.

The incoming flow separates at the front corner of the building because it cannot turn sharply enough to follow the roof and sides. After passing the building, the flow goes towards the wake of the building to form a 'cavity' region, which is bounded by the separation streamline from roof and side edges and the reattachment streamline after the building. Inside this cavity region, a pair of vertically oriented vortices interacts with the main flow field near roof level and also tail off downwind the flow, as shown in Figure 4.1. Thus the flow inside the cavity region is highly turbulent with very small mean velocity. The estimation of the boundaries of this region is very important because the

pollutant concentration from local emission is very likely build up here. Thus the pollutant concentration in the cavity region, especially on the roof of the building and the vertical section of the building might be the major concern for the designers.

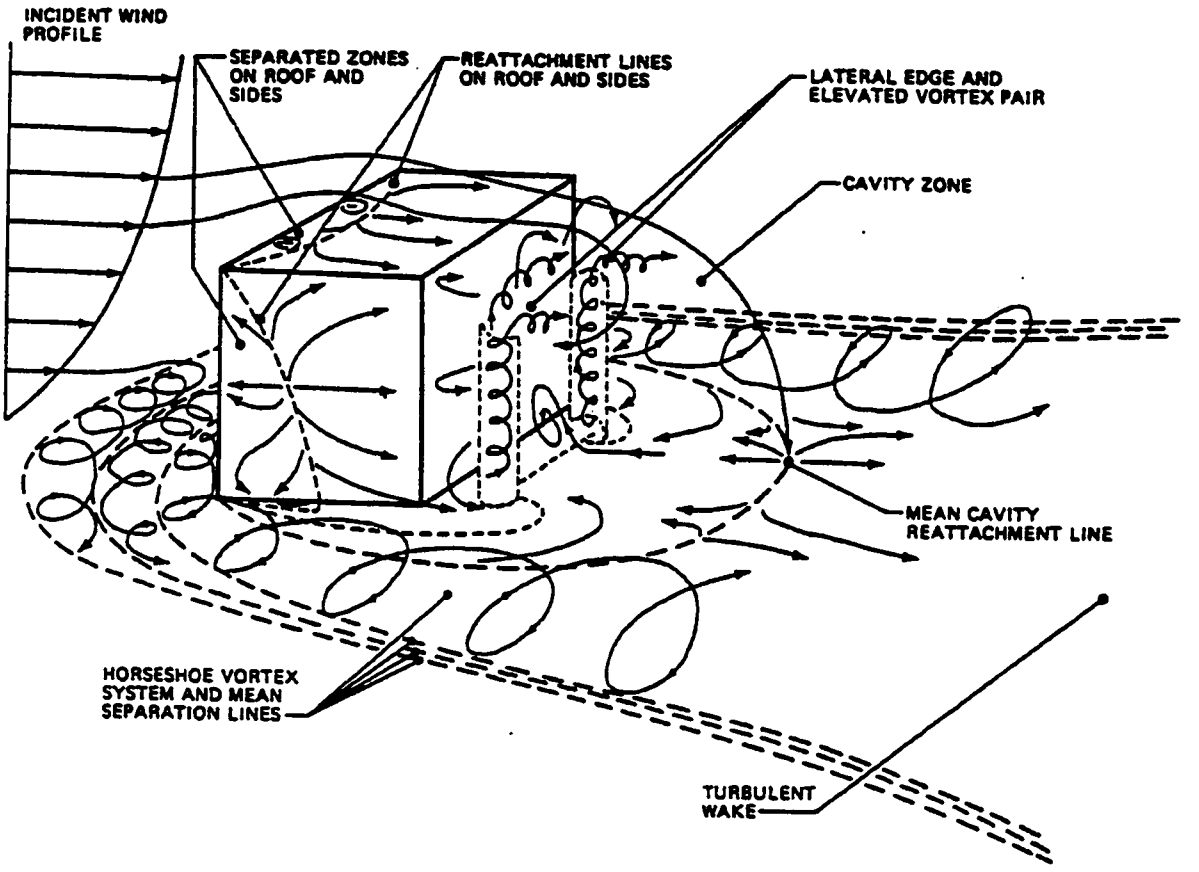


Figure 4.1 Model of flow near a sharp-edged building in a deep boundary layer

(From: Hosker (1979))

There are very few numerical studies of concentration fields in the literature, and most of them model turbulence by using the standard k-ε model. Their results were compared with those from different experiments. Therefore, it is difficult to evaluate the numerical simulation results. A comparison of similar studies could put things into perspective. The pollutant dispersion around a cubic building is selected as the case for

such a study. Results from Zhang (1993) and Brzoska et al. (1996) seem to be the only studies available. Therefore, results of the present simulation have been compared with those of these works. All simulation results were evaluated against the experiment from Li and Meroney (1983).

4.2 Numerical Simulation Set-Up

Brzoska et al. (1996) used an the experimental setup, which was similar to that of Snyder and Lawson (1993). A neutral atmospheric boundary layer was generated at a 1:200 scale for the flow approaching a cube with a side equal to 0.2 m. The boundary layer depth was 2 m and the roughness length was 1.0 mm. The velocity profile approximated a power law curve with an exponent of 0.16. The velocity at building height was calculated as 3.1m/s. The releases were at a rate equal to 10^{-4}g/s from the top center of the cube. The simulation was done by a fourth order accurate finite-element code (FEAT) employing brick elements with 20 nodes using quadratic basis functions and eight internal nodes using linear basis functions for pressure terms. Figure 4.2 shows the geometry and the parameters of the simulation.

In the simulation by Zhang (1993), the setup of wind-tunnel measurements of Castro and Robins (1976) was used. The model building in the wind tunnel was a 0.2-m cube placed in a 2-m deep simulated neutral atmospheric boundary layer. A non-buoyant effluent was emitted through a central port on the rooftop. The power-law exponent of the approaching wind velocity profile was 0.25. The simulation used computer code

TEMPEST, which is an Eulerian, finite-difference code designed to solve the time-dependent equations of motion, continuity, and energy conservation for turbulent flow in incompressible fluids. Figure 4.3 shows the geometry and parameters for Zhang's simulation.

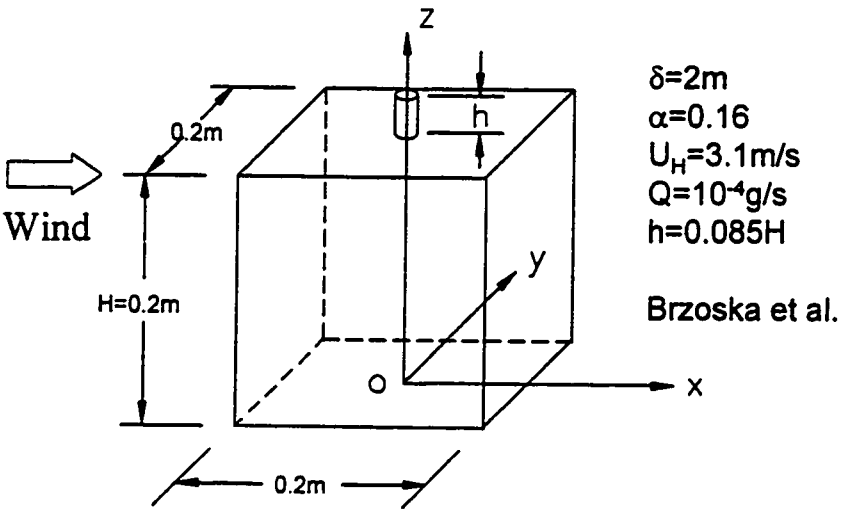


Figure 4.2 Geometry and parameters for the case of Brzoska et al (1996)

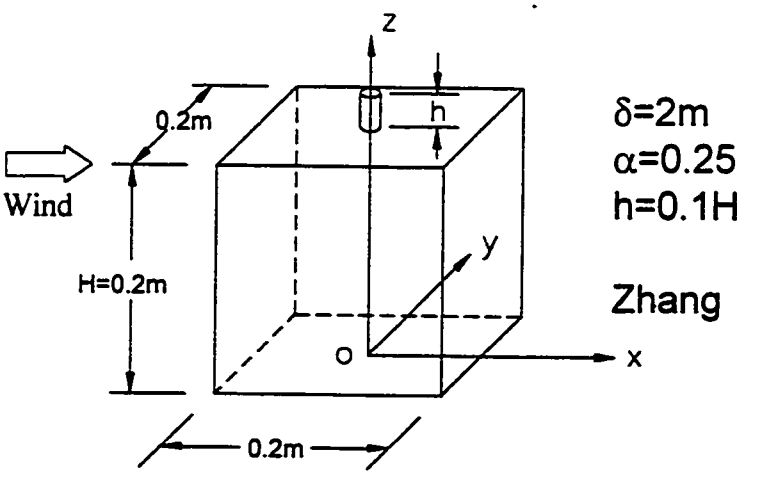


Figure 4.3 Geometry and parameters for the case of Zhang (1993)

In the present simulation, the experimental setup by Li and Meroney (1983) was used. The model building was a 0.05m cube placed in a 0.3m deep simulated atmospheric boundary layer. The power-law exponent of inlet velocity was 0.19. Velocity at building height was 3.3m/s and pure helium was released at a flow rate of 12.5cm^3/s from top center of the cube, as indicated in Figure 4.4.

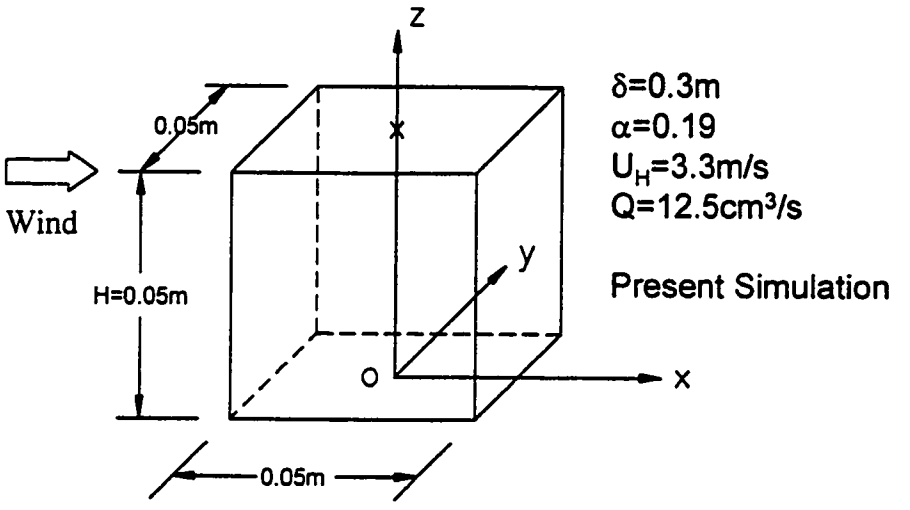
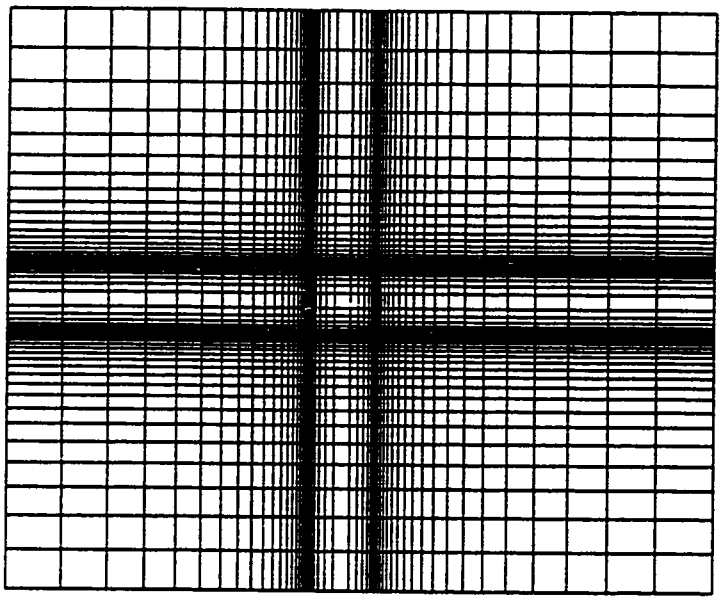
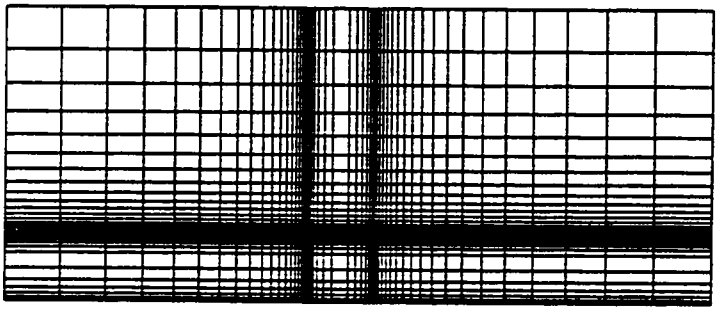


Figure 4.4 Geometry and parameters for the present simulation

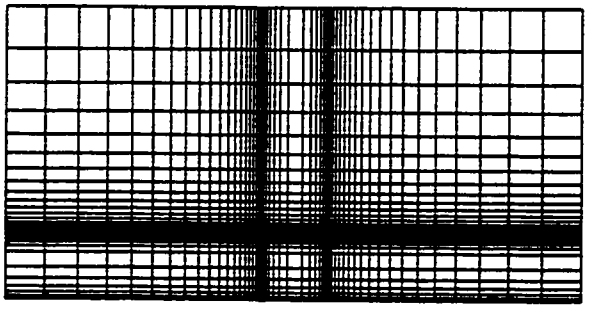
For the present simulation, a grid system of 52 x 52 x 33 is used. As shown in Figure 4.5. The grid extended for a computational domain of 60 cm x 45 cm x 25 cm. Along x-direction, there are 20 grid points in front of the building, 12 grid points along the building length span and 20 grid points behind the building. Along y-direction, there are 20 grid points each from side wall of the building to the side of the computational domain. There are 12 grid points across the building width. Along z-direction, there are



Grid 52x52x33, x-y plane



Grid 52x52x33, x-z plane



Grid 52x52x33, y-z plane

Figure 4.5 Grid system for the present simulation

12 grid points along the building height and 20 grid points from the roof of the building to the top of the computational domain. The height of the model building is 5 cm.

The results from both Zhang (1993), Li and Meroney (1983) and present simulation were calculated originally as nondimensional concentration coefficient CU_HH^2/Q where C is the concentration, U_H is the velocity at building height H, and Q is the emission rate. In order to make the results from the three studies comparable, all of the nondimensional concentration coefficients have been applied to Brozska's (1996) case to get actual concentrations.

4.3 Concentration Field in the Building Wake

All of the results were arranged by using the same conditions, that is, velocity at building height U_H is 3.1 m/s, building height H is 0.2 m and pollutant emission rate Q is 10^{-4} g/s.

Figure 4.6 shows the numerical simulation results from Brzoska et al. (1996), Zhang (1993) and present simulation. The experimental results from Li and Meroney (1983) are also shown for comparison. The vertical plane is the central alongwind plane. The horizontal plane is the plane at the roof level. The graph shows that the area near the stack has the maximum concentration. Pollutants are basically dispersed downwind from the stack and the gradient of concentration along the wind direction is smaller than that in vertical or lateral direction. For the present simulation, the plume goes downwards,

probably due to the low stack height; also the location of the stack is somewhat downwind of the center of the roof and the plume is trapped in the cavity region behind the building. Experimental results are only available for the horizontal roof plane and indicate larger lateral dispersion upwind of the stack than that of Brzoska et al. (1996). Although the experimental results do not extend outside the roof perimeter, they appear to be in better agreement with those of the present simulation in comparison to those from the other references.

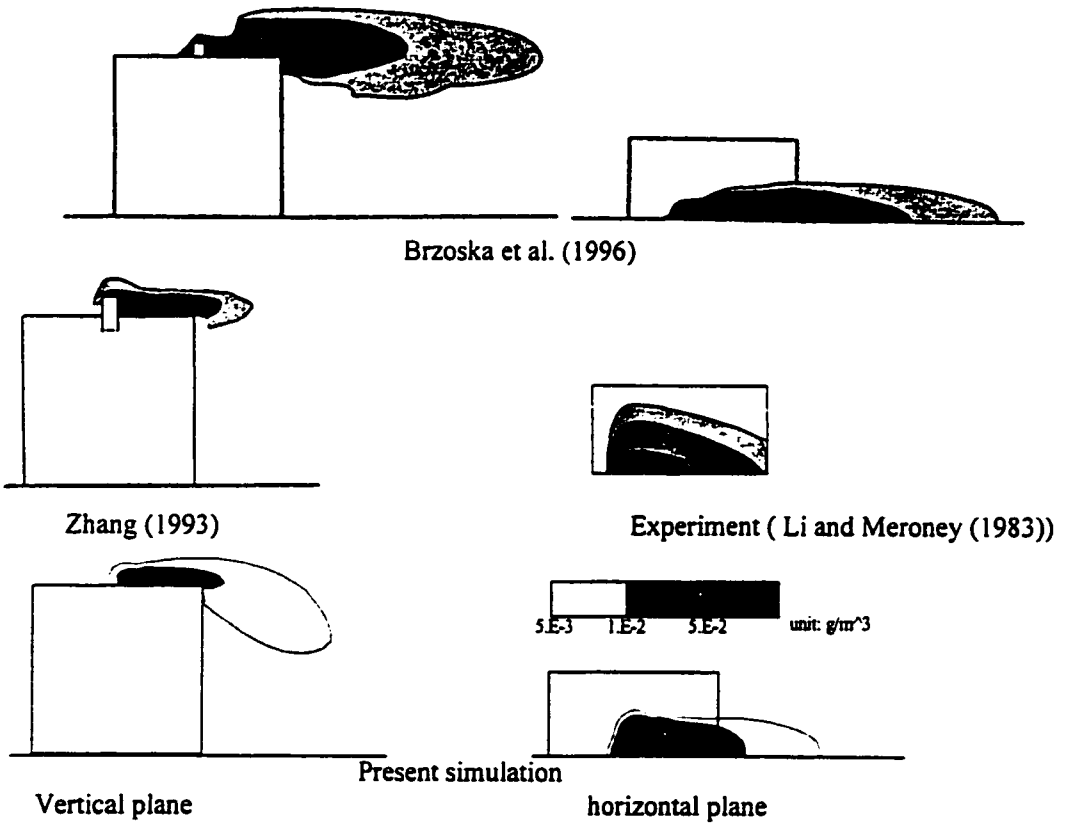


Figure 4.6 Comparison of concentrations for roof top release from a cube

However the concentration field predicted by numerical simulation appears to be located more downstream than that of the experiment both in Brzoska's case and present simulation. Another observation is that the simulation results appear decreasing faster

along the cross wind direction than that of the experiment as can be seen by comparing the distance from the centerline to the edge of the gray area. This implies that there is less lateral diffusion in the numerical simulation.

4.4 A Study of the Effect of the Stack Height

In order to avoid entrainment of exhaust gases into the wake, stacks must terminate above a certain flow recirculation height H_c. If the exhausts discharge from a lower height, pollutants will diffuse very rapidly to the roof and may enter ventilation intakes or other openings, as shown in Figure 1.1. Therefore, selecting an appropriate stack height is very important for architects and engineers. In the current practice, stack selection is largely depending on experience and some empirical formulas derived from experiments. A computer simulation can give a fast and detail evaluation of the effect of design alternatives.

Figure 4.7 shows the comparison of vertical concentration field for different stack height releases. It shows that the higher the stack, the less the plume downwash effect.

Figure 4.8 shows the comparison of horizontal concentration field for different stack height releases. The figure shows that as the stack goes higher, the concentration on the roof goes lower and the maximum concentration goes downwind.

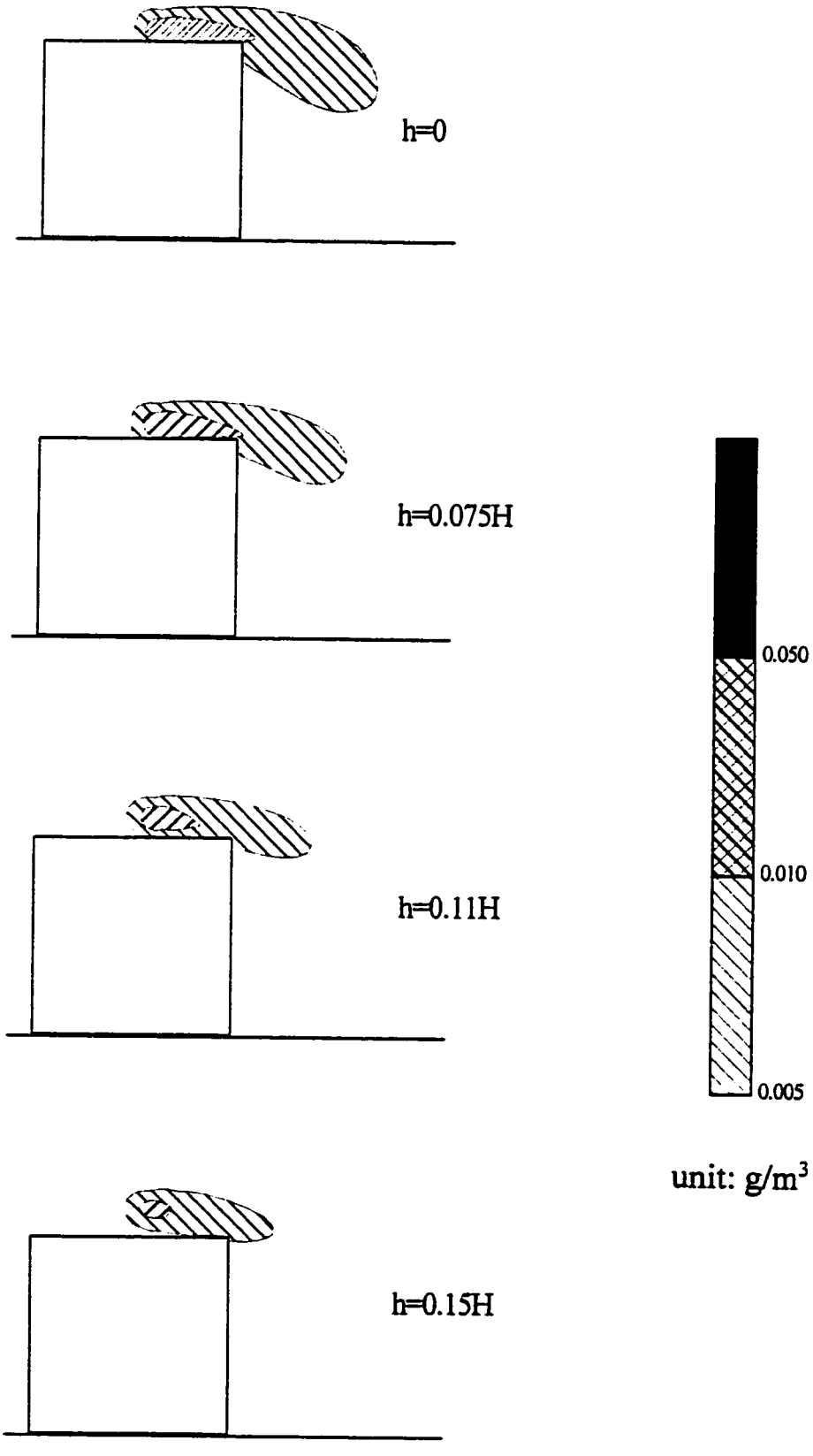


Figure 4.7 Comparison of vertical concentration field for different stack height releases

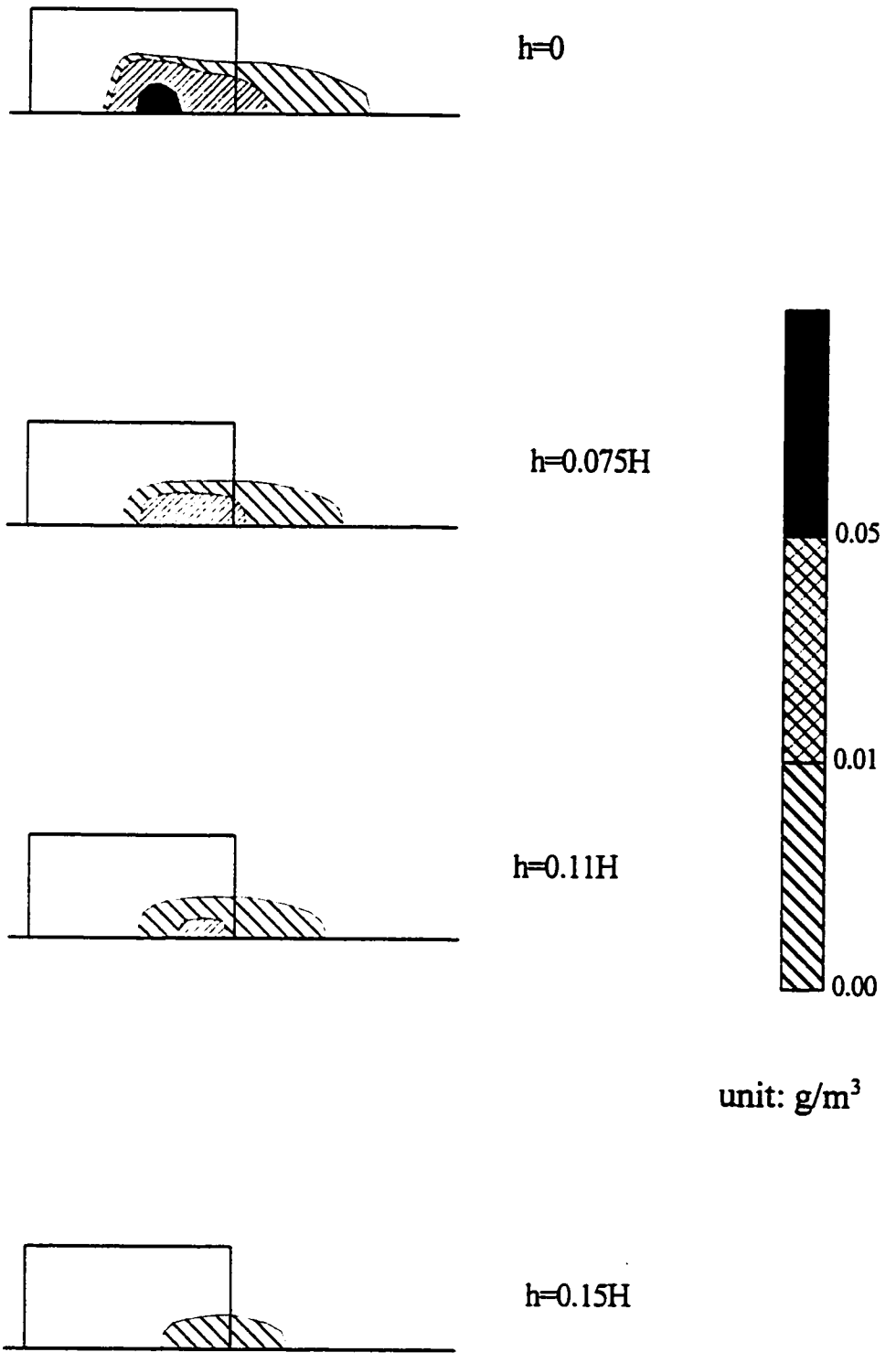


Figure 4.8 Comparison of horizontal concentration field for different stack height releases

4.5 Summary

The numerical simulation result of pollutant dispersion around a cubic building has been investigated. It is found that the prediction of the along wind concentration moves downstream. The possible explanation of this phenomenon could be that the standard k–ε model fails to predict the recirculating flow on the roof, which leads to overprediction of the velocity on the roof, thus the convection to downstream becomes stronger.

The second observation is the underprediction of the lateral diffusion. This could possibly be attributed to the isotropic turbulence model used.

From those observations it appears that a better simulation of the recirculating flow on the roof might improve the prediction of concentration field.

The effect of stack height to the concentration field has been investigated. The numerical simulation shows a great potential as a design tool in this aspect.

Chapter 5

FLOW FIELD AND CONCENTRATION FIELD AROUND A RECTANGULAR BUILDING

5.1 Introduction

The pollutant dispersion around a cubic building was discussed in detail in the previous chapter. The cubic building is a textbook case, in which the length, width and height of the building are identical. In reality, these three parameters are seldom the same. However, the cubic building case serves as a bench mark against which the effects of different methods can be contrasted and compared; it also provides a very useful, basic insight into the phenomena of pollutant dispersion around buildings.

In this chapter, pollutant dispersion around a more realistic rectangular building will be studied in detail. As stated in the previous chapters, the entrainment of plumes emitted from short stacks into the wakes of buildings can result in maximum ground-level concentrations that are significantly greater than those found for similar sources in the absence of buildings which could be a major concern for architects and engineers.

The wake of the building could be catergorized into the following areas: In the immediate leeward side of the building, there is a "cavity" region where recirculation occurs, mean velocities are reduced and the air flow is highly turbulent. The flow in the

building wake farther downstream is characterized by a high turbulence intensity and mean velocity deficit that progressively decays to background levels. Due to the complexity of the flow inside this region, it is very difficult to determine the concentration close to the building using the Gaussian plume model and associated dispersion parameters. In addition to the so-called near-wake region, after the "cavity" region (approximate 3 building heights) to about 10 building heights is the medium wake region. After 10 building heights, the flow reestablishes itself to background conditions. This region is call the far-wake region.

Huber and Snyder (1982) found that the plume from a ground-level source spread rapidly both in the vertical and lateral directions in the wake of the building while plumes released at or above 1.2 times the building height were similarly enhanced in the vertical but not so much in the lateral direction. They also found that in the immediate leeward side of the building (cavity), rapid dispersion occurs father downwind, where the flow reestablishes itself to background conditions, plume spread is controlled by the decaying turbulence. In general, it seems that for approximately 10 building heights, the net building wake effect on plume spread can be simply accounted for by changing the effective source location, so long as the background plume spread in the absence of the building influence can be characterized.

In the present study, the flow field immediately after the building will be simulated and compared with observations from Huber and Snyder (1982). The predicted concentration field from a ground source will be compared with the experimental results

(Huber and Snyder (1982)) and the Modified Gaussian Model (see chapter 2) for the ground-level concentration. The vertical profile at the near-wake field and far wake field as well as the lateral profile at the near-wake field and far wake field will also be compared with previous experimental results, the Modified Gaussian Model and the present simulaton.. The predicted concentration field from an elevated source will be compared with the experimental result (Huber et al. (1980)) and the numerical simulation result from Selvam and Huber (1995).

5.2 Numerical Simulation Set-Up

The numerical simulation set-up simulated the experiment in the following aspects: the oncoming wind velocity, the geometry of the building and the stack location. The pollutant release rate was also simulated.

As stated in chapter 3, staggered grid systems were used for all the simulations.

5.2.1 Pollutant dispersion from a ground source

The experimental study of Huber and Snyder (1982) was conducted in the Meteorological Wind Tunnel of the U.S. Environmental Protection Agency's Fluid Modeling Facility. The depth of the boundary layer, δ, was 1.8m, so that the ratio of the boundary layer thickness to the building height δ/H was 7:1. The velocity profile U/U_r

was found to fit a one-sixth power law, which is generally representative of neutral atmospheric flow over moderately smooth terrain (Davenport, 1963). The reference velocity, U_r = 2.34 m/s, was measured at 1.5 H above the surface. For this experimental study, a 1:200 scale model of a 50 m high prototype building would be an appropriate example. In this scale, the boundary layer in the wind tunnel represents a 360-m deep atmospheric boundary layer. The estimated model surface roughness length (0.065 cm) represents 13 cm in the field, which corresponds to that typical of flow over high grass. The rectangular building has a length equal to twice its height and width. The long side of the building was oriented perpendicularly to the approaching wind. The stack was placed midway along the lee side of the building, as shown in Figure 5.1.

A grid of 62 x 40 x 34 is used for the present numerical simulation, see Figure 5.2. The grid extended for a computational domain of 650 cm x 250 cm x 125 cm. Along x-direction, there are 22 grid points in front of the building, 16 grid points along the building length span and 24 grid points behind the building. Along y-direction, there are 10 grid points each from side wall of the building to the side of the computational domain. There are 20 grid points across the building width. Along z-direction, there are 14 grid points along the building height and 20 grid points from the roof of the building to the top of the computational domain. The height of the model building is 25 cm.

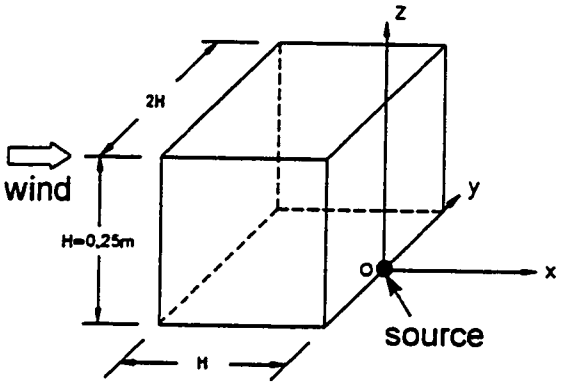
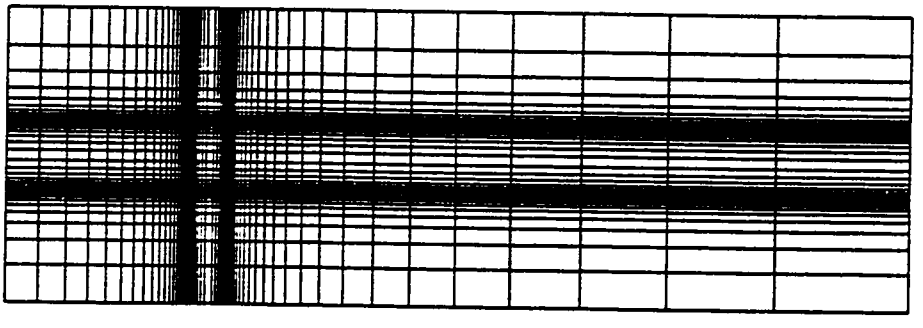
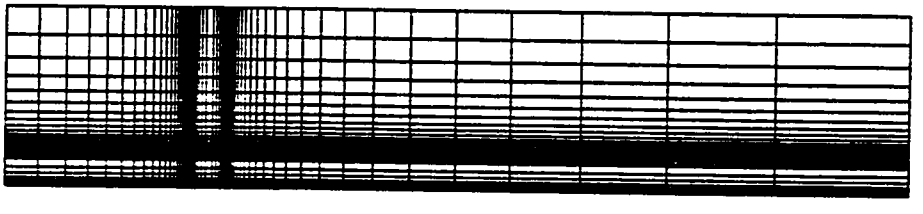


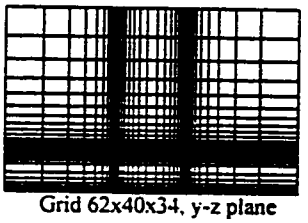
Figure 5.1 Experimental set-up for ground source, after Huber and Snyder (1982)



Grid 62x40x34, x-y plane



Grid 62x40x34, x-z plane



Grid 62x40x34, y-z plane

Figure 5.2 Grid system for the ground source

5.2.2 Pollutant dispersion from an elevated source

The results of present simulation are compared with measured concentrations from a wind-tunnel study (Huber et. al. (1980)) and with simulated results from Selvam and Huber (1995) for the case of pollutant dispersed from an elevated source. In the experimental study, a rectangular-shaped building with its length equal to twice its height and width was selected. The long side of the building was oriented perpendicularly to the approaching wind. The model building height was H = 25 cm and the depth of the boundary layer δ, was 1.8 m. The velocity profile U/U_r was found to fit a one-sixth power law, which is generally representative of neutral atmospheric flow over moderately smooth terrain (Davenport (1965)). The reference velocity, U_r = 2.34 m/s, was measured at 1.5 H above the ground. The concentration measured in the experiment was used in nondimensional form CU_rH^2/Q, where Q is the emission rate. For this study, a 1:200 scale model of a 50-m high prototype building would be an appropriate example. At this scale, the boundary layer of the wind tunnel represents a 360-m deep atmospheric boundary layer. The case is shown in Figure 5.3.

Two grid systems were used for the numerical simulation of pollutant dispersion from elevated source, for the purpose of numerical error estimation. Figure 5.4 shows the coarse grid system and Figure 5.5 shows the fine grid system. The coarse grid system has a grid of 31 x 40 x 17 which extended to a range of 500 cm x 250 cm x 125 cm. Along x-direction, there are 11 grids in front of the building, 8 grids on the building and 12 grids

behind the building. Along y-direction, there are 10 grids on each side of the building and 20 grids on the building. Along z-direction, there are 7 grids on the building and 10 grids above the building. The fine grid system counts 62 x 80x 34 points and it is extended to the same domain as the coarse grid system. The number of grids was doubled for all the directions compared with the coarse grid system. Along x-direction, there are 22 grids in front of the building, 16 grids on the building and 24 grids behind the building. Along y-direction, there are 20 grid on each side of the building and 40 grids on the building. Along z-direction, there are 14 grids on the building and 20 grids above the building.

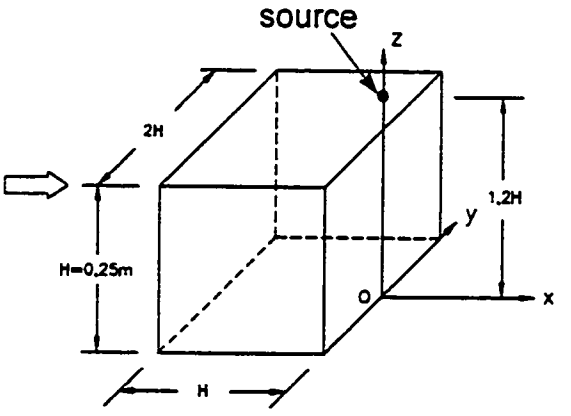
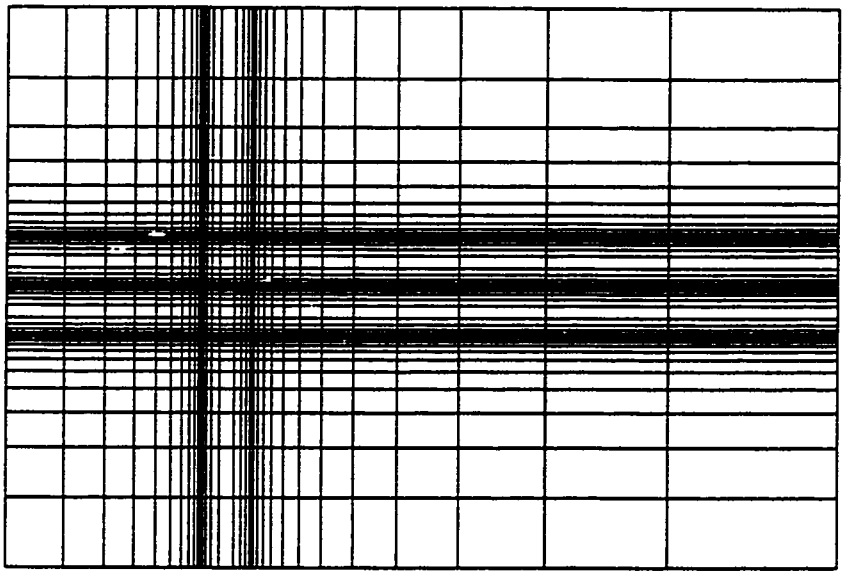
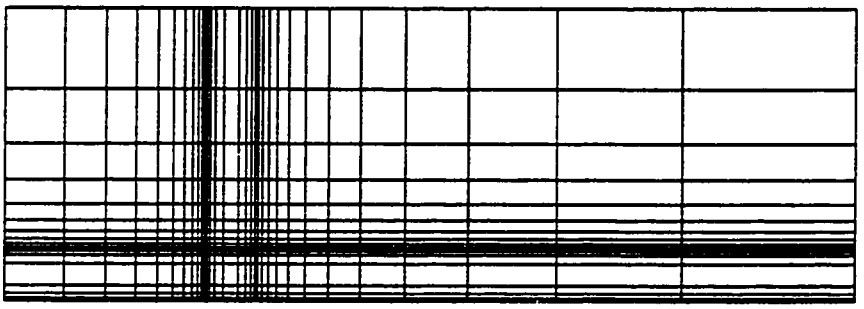


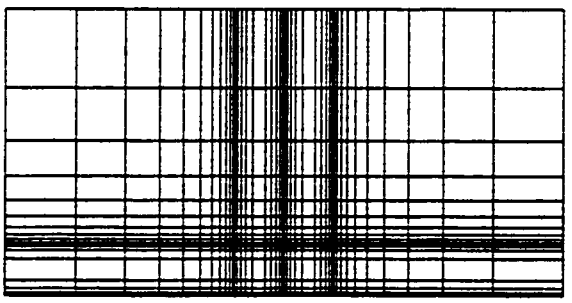
Figure 5.3 Experimental set-up for elevated source, after Selvam and Huber (1995)



Grid 31x40x17, x-y plane

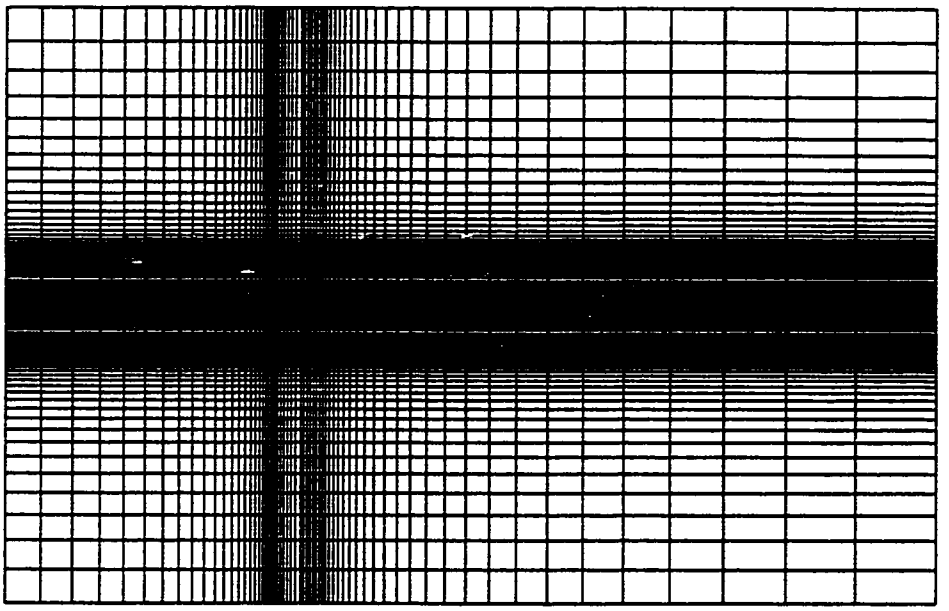


Grid 31x40x17, x-z plane

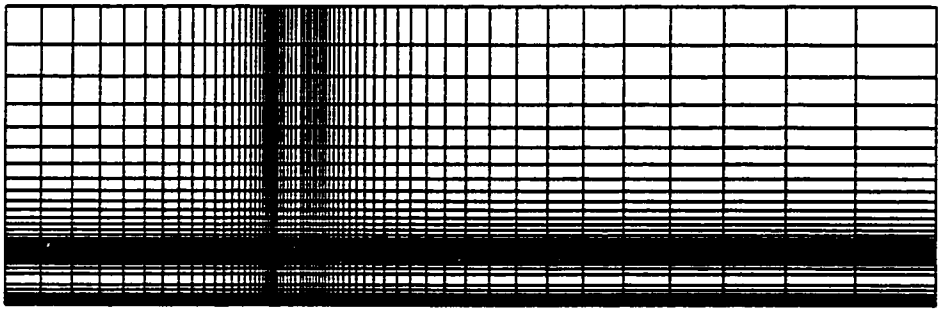


Grid 31x40x17, y-z plane

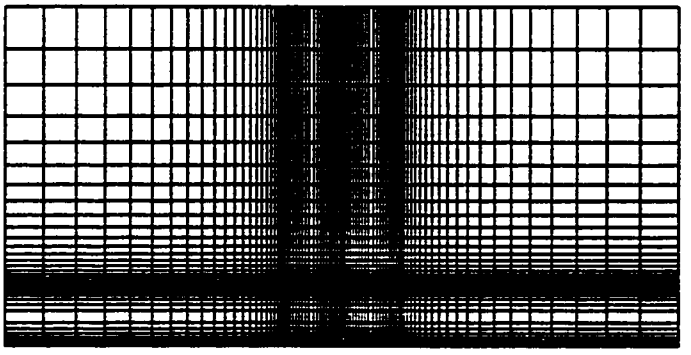
Figure 5.4 Coarse grid system for elevated source



Grid 62x80x34, x-y plane



Grid 62x80x34, x-z plane



Grid 62x80x34, y-z plane

Figure 5.5 Fine grid system for elevated source

5.3 Flow Field around the Rectangular Building

An understanding of the flow patterns near buildings and other flow obstacles should prove extremely useful in understanding and predicting effluent concentration fields near these objects. Therefore, the flow field around the rectangular building was investigated prior to investigating the concentration field.

Figure 5.6 shows the vertical wake cavity and envelope measurements in the experiment conducted by Huber and Snyder (1982). The cavity boundary was determined as the point where no smoke could be seen recirculating back to the base of the building. The wake envelope was determined as the maximum spread of smoke that was emitted from a source within the cavity region. The cavity length was found approximately 2.9 building heights in extent downwind from the leeward edge of the building.

Figure 5.7 shows the flow field predicted by the present simulation. The cavity length was predicted around 2.1 building heights. It agrees rather well with the experiment considering that the present simulation has rather coarse grid around the centerline of the rectangular building.

Figure 5.8 presents the cavity boundary and wake envelope measurements in horizontal plane of the building at half building height. The cavity length was approximately 2.5 building heights, which was also well predicted by the present simulation as 2.2 building heights, as shown in Figure 5.9.

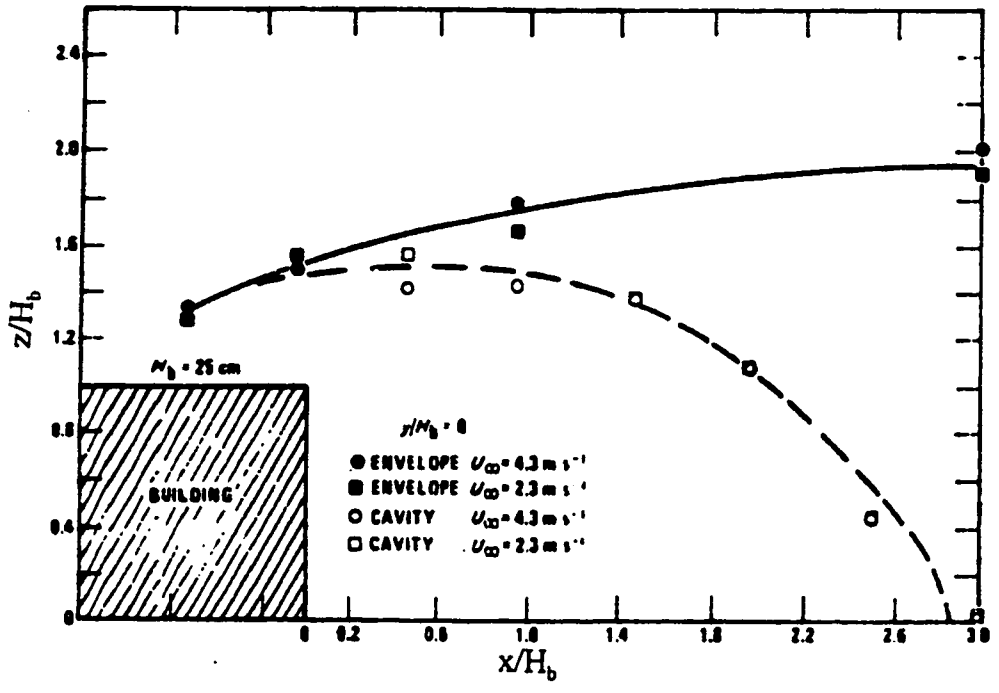


Figure 5.6 Vertical wake cavity and envelope measurements

(From: Huber and Snyder (1982))

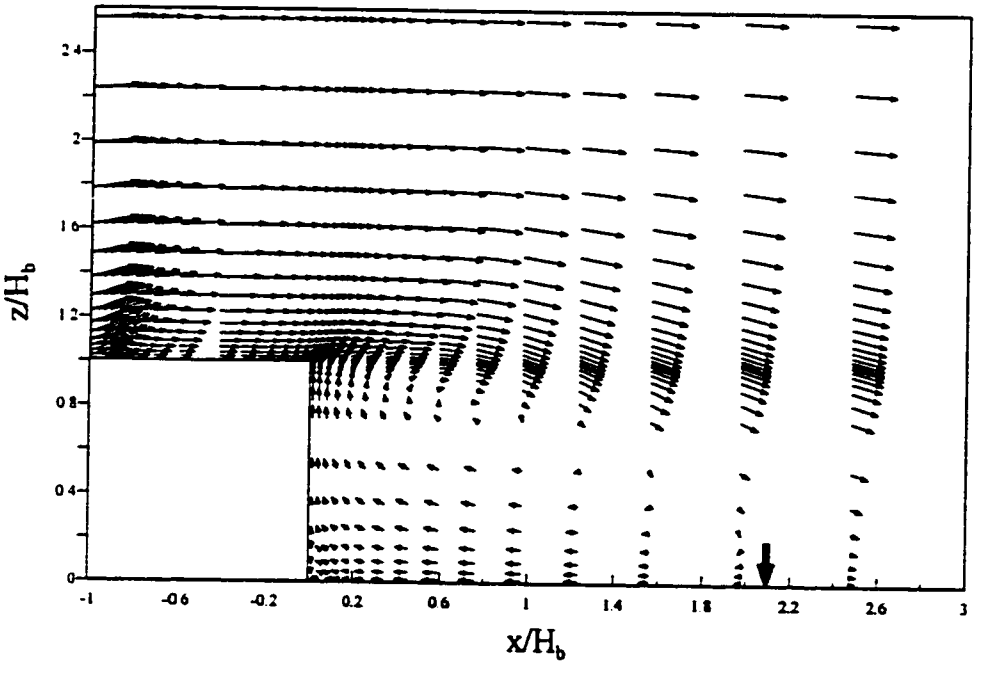


Figure 5.7 Vertical velocity vector field (present simulation)

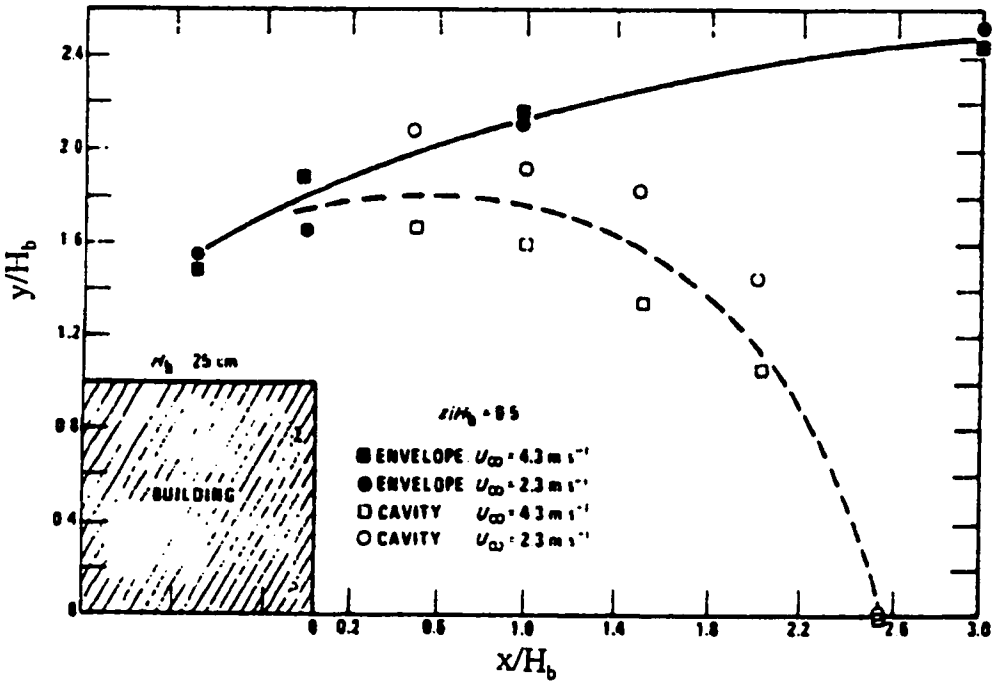


Figure 5.8 Lateral wake cavity and envelope measurements

(From: Huber and Snyder (1982))

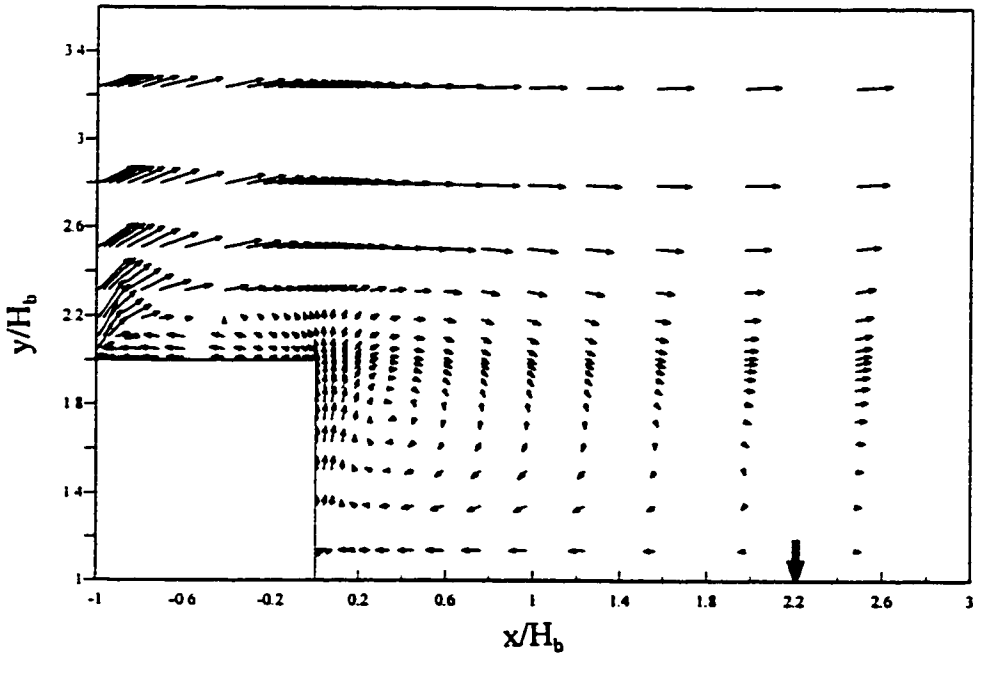


Figure 5.9 Horizontal velocity vector field (present simulation)

5.4 Concentration Field around the Rectangular Building

As stated in Chapter 3, after the flow field around the rectangular building has been solved, the concentration field is solved by the time marching method with the hybrid scheme. The results are compared with those from the experiment of Huber and Snyder (1982) and Modified Gaussian Model (chapter 2).

5.4.1 Concentration field around the rectangular building for ground source

Figure 5.10 shows the simulated longitudinal ground-level concentration for ground source. It takes about 10 hours CPU time on a DEC Alpha Station 255/300 (CPU clock speed 300MHz). The resulting ground-level concentration is very high near the back wall of the building, then decreases rapidly inside the building wake. The decrease of the concentration becomes smaller in the far-field of the building wake. The simulation result agrees extremely well with the experimental result from Huber and Snyder (1982). The Modified Gaussian Model (equation (2-6), equation (2-7))can be determined from the ground-level concentration measurements for the ground source from Huber and Snyder (1982). Figure 5.10 shows that the Modified Gaussian Model agrees very well with ground-level concentration measurements, but only for the range of $x/H \geq 3$, where x is the distance from the back wall of the building and H is the building height.

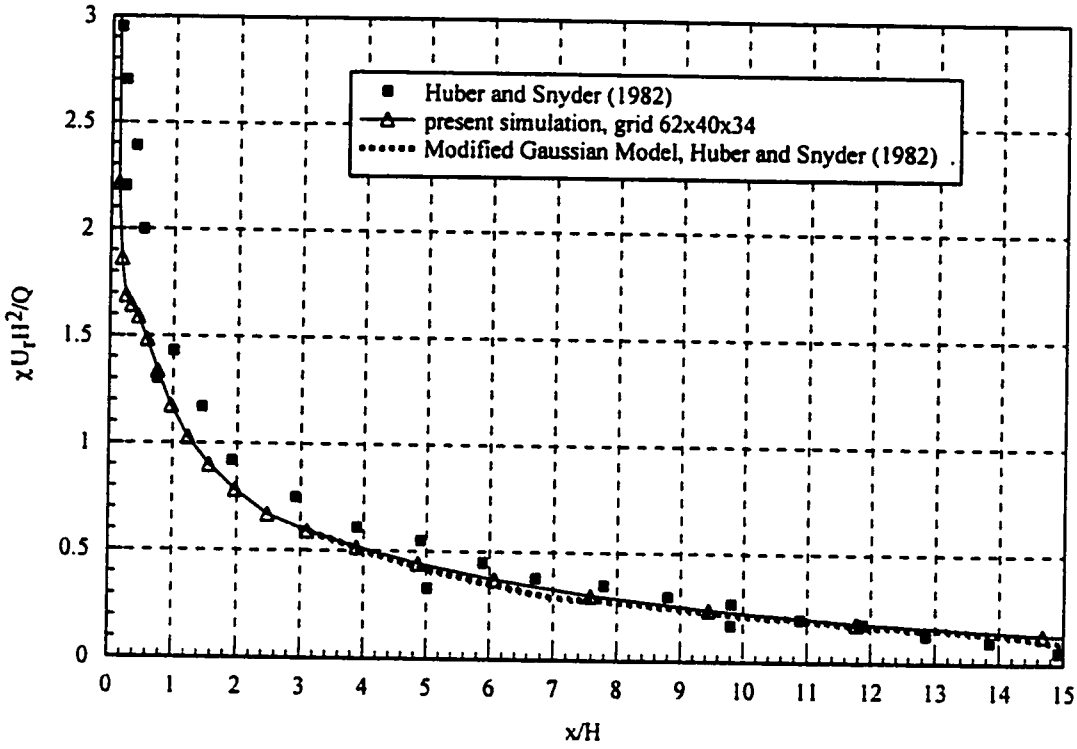


Figure 5.10 Longitudinal ground-level concentration for ground source

Figure 5.11 shows the vertical concentration profiles for ground source at x/H = 5. The measured concentration is high near the ground, then decreases as the height increases. Both the numerical simulation and the Modified Gaussian Model predicted this trend. The numerical simulation result appears better than the Modified Gaussian Model near the ground up to about half building height, but also inferior to the Modified Gaussian Model for higher heights.

Figure 5.12 presents the vertical concentration profile for the ground source at x/H = 10. Again, the measured concentration decreases as the height increases, and both the numerical simulation and the experiment follow this trend. It appears that the

numerical simulation result agrees well with the experimental measurement near the ground up to half the building height and away from the ground, around 2.5 building heights, however, it is less satisfactory between 0.5 to 2.5 building heights.

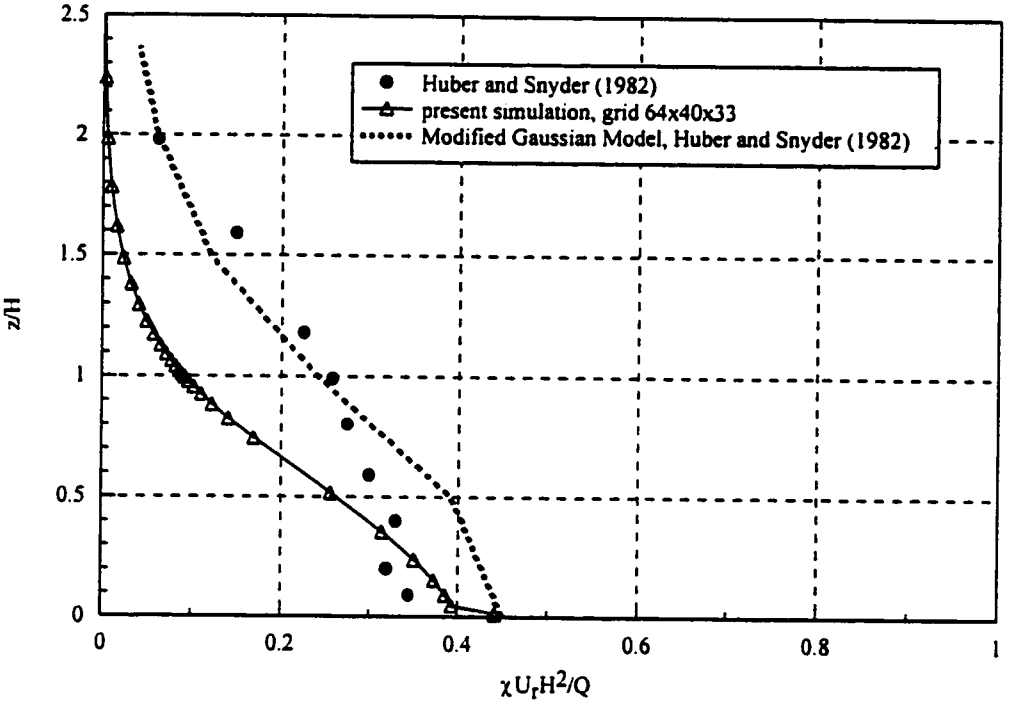


Figure 5.11 Vertical concentration for ground source at x/H = 5

Figure 5.13 presents the vertical concentration profile for the ground source at x/H = 15. In this case, the Modified Gaussian Model appears to have a better prediction than the present simulation.

From the comparisons above, it appears that generally the present simulation agrees well with experimental measurements for the vertical concentration profile in the near, medium and far wake. The present simulation agrees very well with the

experimental results near the ground while it is less satisfactory for higher heights. It seems that the present simulation predicts better than the Modified Gaussian Model for the near and medium building wake near the ground. However, the Modified Gaussian Model works better than the present simulation for the far wake. One possible explanation is that the farther from the building, the less is the building influence, and the more accurate is the concentration field described by a Gaussian or Modified Gaussian Model. The nearer to the building, the more complex the flow field, and the less likely could the plume be described by the Gaussian or Modified Gaussian Model. The present numerical simulation predicts the measured concentration field well, particularly at lower heights.

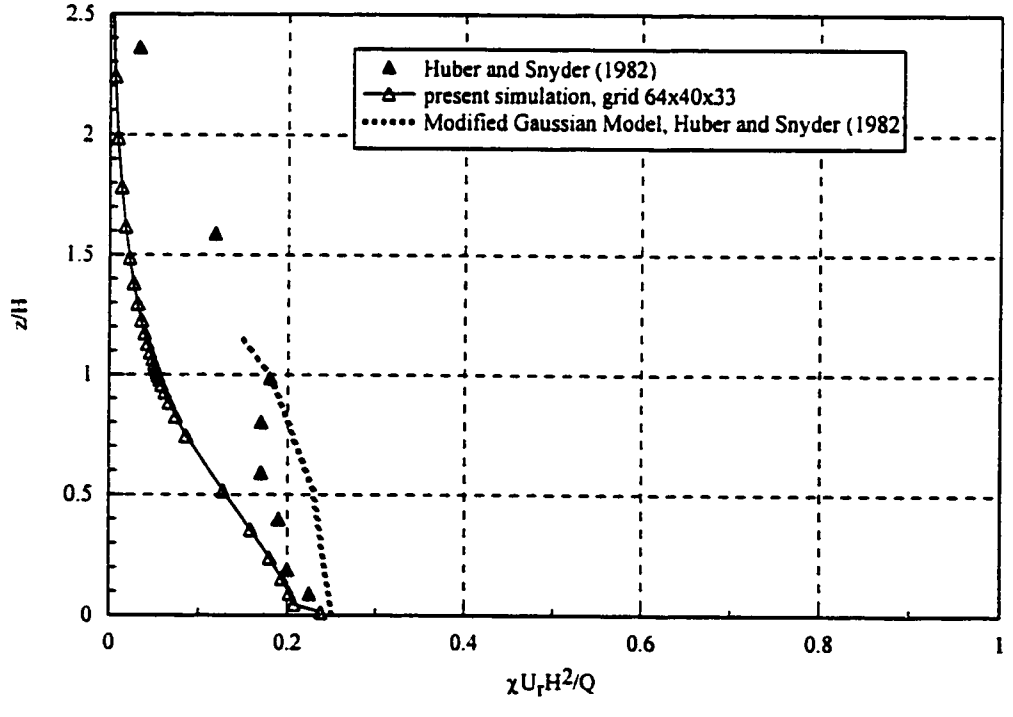


Figure 5.12 Vertical concentration for ground source at x/H = 10

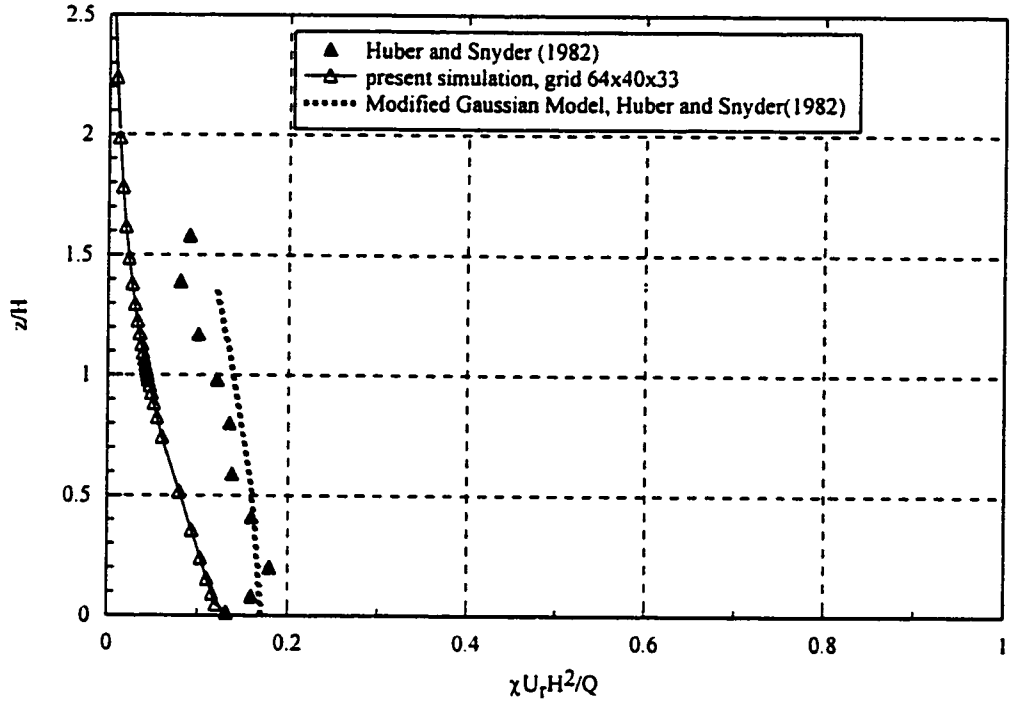


Figure 5.13 Vertical concentration for ground source at x/H = 15

Figure 5.14 shows the lateral ground-level concentration for ground source at x/H = 5. The lateral profiles are in good agreement with the Modified Gaussian Model estimates. The maximum concentrations are predicted quite well by both the present simulation and the Modified Gaussian Model. But the horizontal widths of the plume are overestimated by the present simulation while underestimated by the Modified Gaussian Model.

Figure 5.15 presents the lateral ground-level concentration for ground source at x/H=15. Neither the present simulation nor the Modified Gaussian Model has a good prediction of the maximum ground concentration. But the present simulation predicted better the concentration near the centerline. For the horizontal widths of the plumes, the

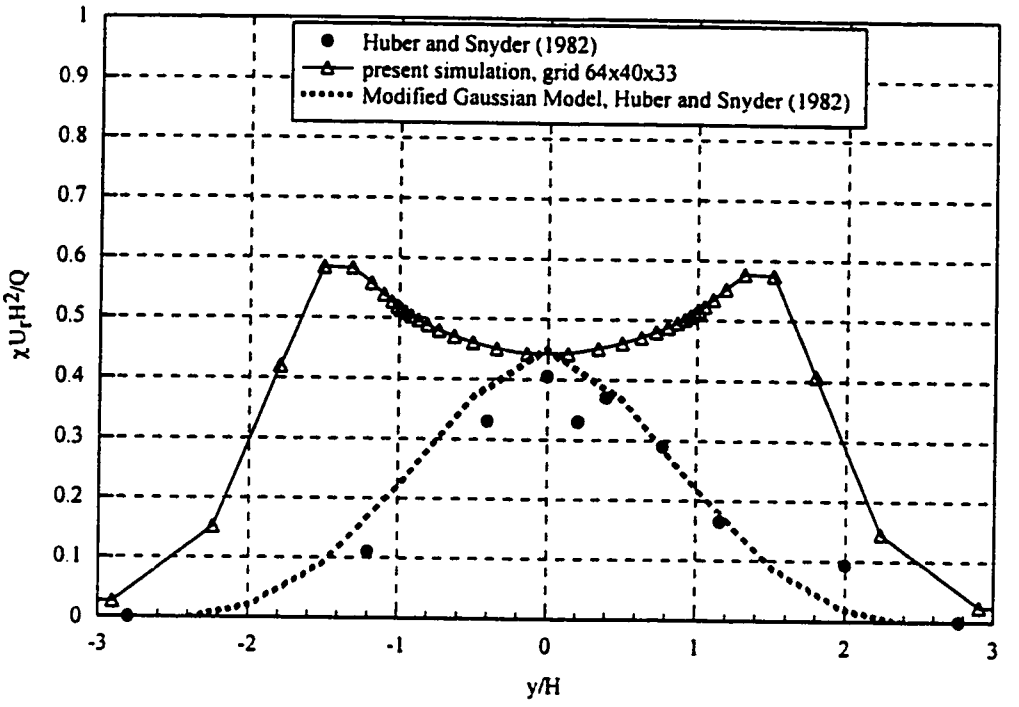


Figure 5.14 Lateral ground-level concentration for ground source at x/H = 5

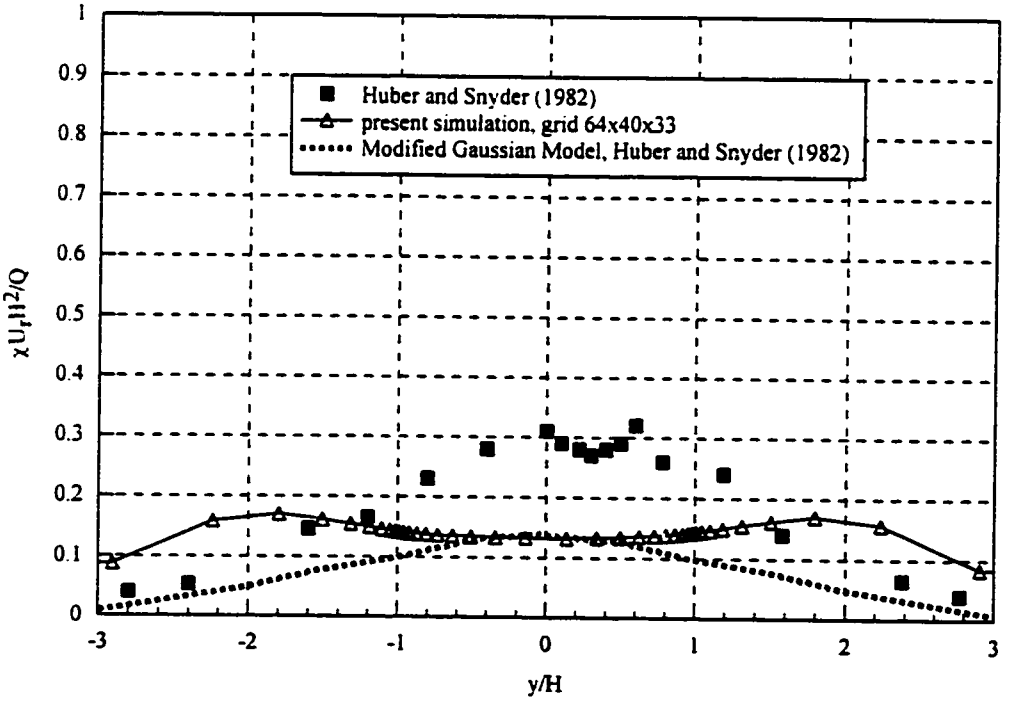


Figure 5.15 Lateral ground-level concentration for ground source at x/H = 15

present simulation overestimated the widths while the Modified Gaussian Model underestimated the widths.

5.4.2 Concentration field around the rectangular building for elevated source

Figure 5.16 shows the longitudinal ground-level concentrations along the centerline for an elevated source. The wind-tunnel experiment shows that ground-level

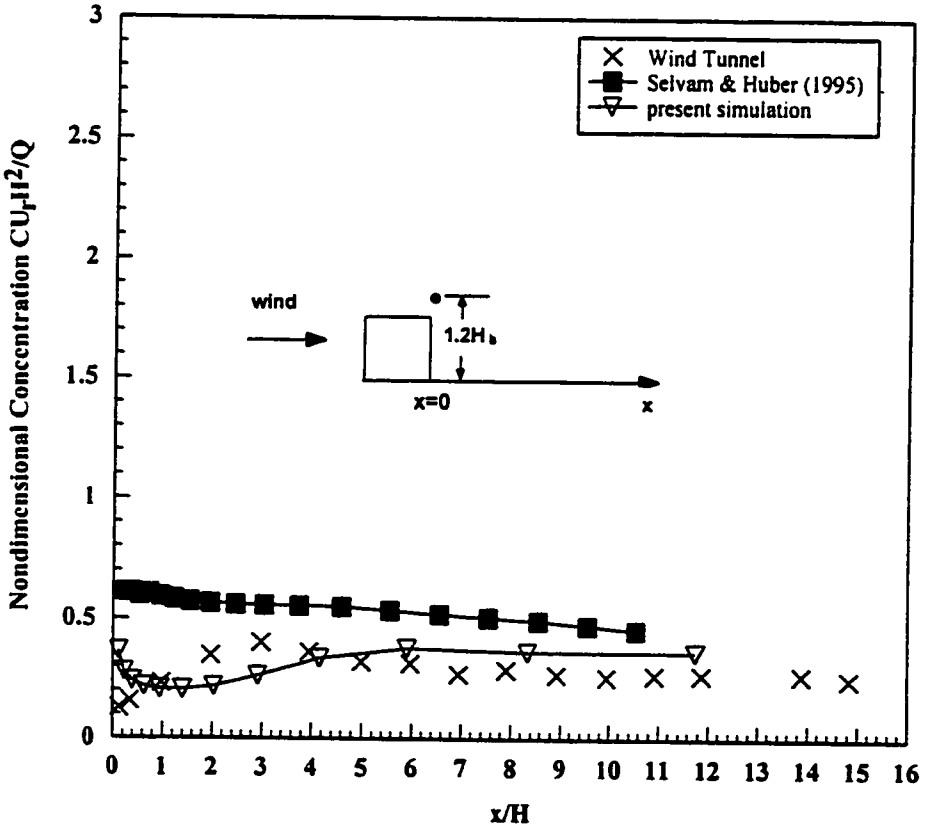


Figure 5.16 Longitudinal ground-level concentration along the centerline for elevated source

concentration is very low near the leeward wall of the building, increases farther downstream until it reaches a maximum at x/H = 3 and then decreases and becomes relatively flat after x/H = 7. The present simulation predicts higher concentration near the leeward wall, decreases along the x-direction up to x/H = 1 and then it follows a similar trend with the experiment results. The prediction from Selvam and Huber (1995) is rather poor; the simulated values are always higher than the experimental results and the ground-level concentration keeps decreasing father downstream.

Figure 5.17 shows the vertical profiles of concentration along the centerline at

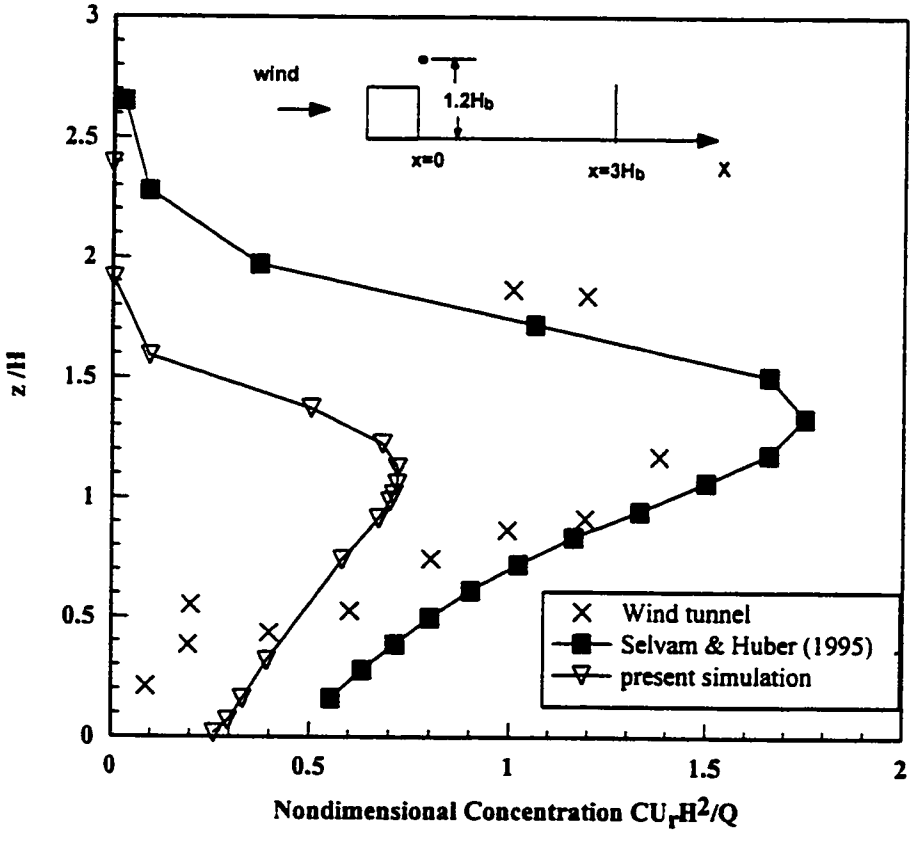


Figure 5.17 Vertical profiles of concentrations along the centerline at x/H = 3 for elevated source

3H for source position 2. The prediction is consistent with the experimental data at the lower part of the building height but it fails clearly at the higher part. The prediction from Selvam and Huber (1995) agrees much better with the experimental results at the higher part but not at the lower part.

5.5 Error Analysis

As stated in Chapter 2, error analysis is important for the evaluation of numerical simulation results. Unfortunately, the error involved in simulating the flow field and pollutant concentrations has not been investigated completely for any of the available methods. An attempt has been made to deal with this issue in the present study.

5.5.1 Introduction

Errors of a numerical method basically arise from discretization and iteration non-convergence. The discretization error here includes the error caused by coordinate transformations. In addition, the approximation of boundary conditions often contributes errors. The limited arithmetic precision of computers introduces round-off errors but these are almost always much smaller than the other errors. The present study focuses mainly on the discretization error and the other due to iteration non-convergence: the convergence error.

5.5.1.1 Discretization error

Discretization error is the difference between the solution of the approximate difference equations and the 'exact' solution of the differential equations. However, comparison with 'exact' solutions obtained by solving the problem on very fine grids is impractical for engineering problems. Using results for two different grids, techniques such as Richardson extrapolation --see Ferziger (1993) -- can be used to estimate the discretization error. For a first-order method, the error is assumed as:

$$\varepsilon_h = \phi_{exact} - \phi_h \quad \text{.....(5-1)}$$

where ϕ_h denotes the solution when the grid-point spacing is h. Expanding ε_h by Taylor Series:

$$\varepsilon_h = hx_1 + h^2 x_2 + h^3 x_3 + \ldots \quad (5\text{-}2)$$

where x_i are functions of the coordinates independent of h. Then, if the calculation is repeated with h replaced by 2h (a calculation on a grid twice as coarse), the error becomes:

$$\varepsilon_{2h} = 2hx_1 + 4h^2 x_2 + 8h^3 x_3 + \ldots \quad (5\text{-}3)$$

Now, if h is small enough, the error can be estimated as:

$$\varepsilon_h \approx hx_1 \approx \varepsilon_{2h} - \varepsilon_h = \phi_h - \phi_{2h} \quad \text{.....(5-4)}$$

and the relative discretization error can then be expressed by:

$$R\varepsilon_h = \frac{\varepsilon_h}{\phi_h} \approx 1 - \frac{\phi_{2h}}{\phi_h} \quad \text{.....(5-5)}$$

5.5.1.2. Iteration convergence error

One key issue determining accuracy of a numerical solution is iteration convergence, i.e. the final arrival at a solution of a finite-difference equation via iteration. This error is defined as the difference between the current iterate and the exact solution of the difference equation. The convergence criterion is selected to control the convergence error under a certain level. A summary of iteration convergence criteria has been provided in chapter 3.

5.5.2. The case investigated

The results of present simulation have been compared with measured concentrations from a wind-tunnel study (Huber et. al. (1980)) and with simulated results from Selvam and Huber (1995), as described in section 5.2.2.

In the numerical simulation of Selvam and Huber (1995), the flow region was divided into 45 x 42 x 25 grid points (44 x 41 x 24 control volumes). The building was modeled as a volume composed of 9 x 9 x 9 control volumes. For the calculation of flow field, the computation stopped when the relative residue was 0.03. The type of convergence criteria used for the computation of concentration field was not reported.

The present numerical study simulated the same wind tunnel experiment. The computational domain was divided into 31 x 40 x 17 grid points, as shown in Figure 5.4. A second mesh system doubled the grid points along all three directions and produced 62 x 80 x 34 grid points in order to study discretization effects, as shown in Figure 5.5. The cases studied included a point source 1.2 H above the ground at the center of the leeward building wall, as shown in Figure. 5.3.

5.5.3 Results and discussion

In order to estimate the numerical error in the present simulation, the convergence behavior in solving the flow field is shown in Figure 5.18. All three velocity component

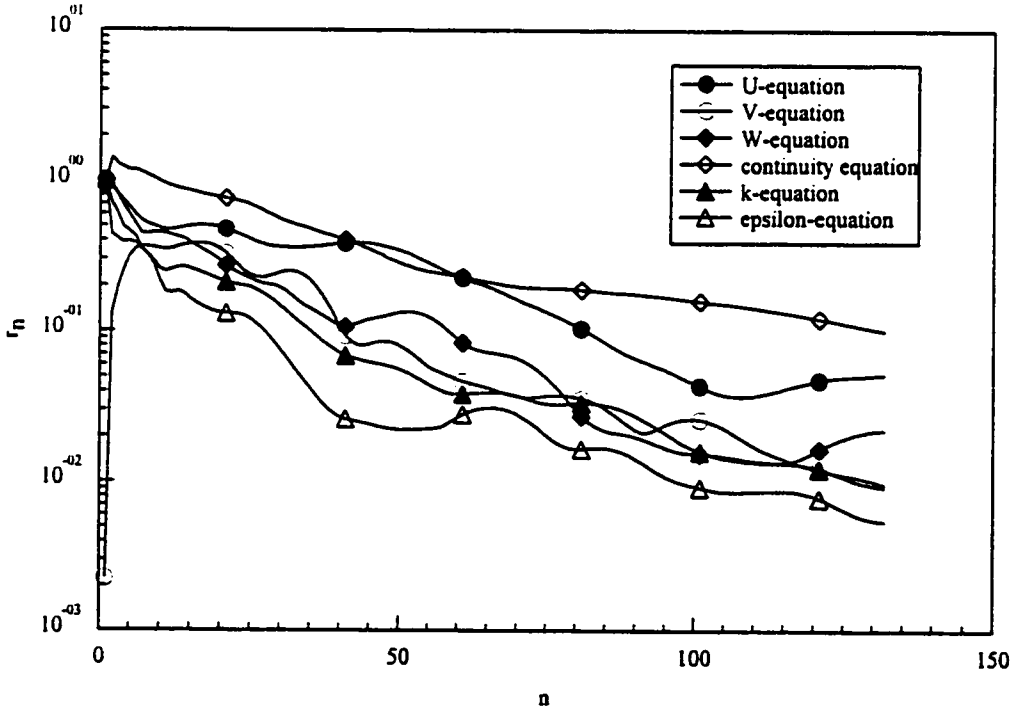


Figure 5.18 Relative residue of discretized equations vs. iteration steps in solving the flow field

residues are normalized by the first residue of the U-equation whereas the k, ε and continuity equation residues are normalized by the first residue of their respective equations. The relative residues of U,V,W momentum equations, k-equation, ε-equation and continuity equation appear to decrease rapidly with the iteration steps. The relative residue criterion of 0.1 is satisfied after 132 iterations. (See section 3.5.2 for details about relative residue).

Figure 5.19 shows the difference of iterates as a function of iteration steps for the calculation of concentration field. Clearly, maxD (defined in section 3.5.1) decreases monotonically up to approximately 2 x 10^{-4}; then it shows a small increase but it eventually decreases rapidly. As shown in the graph, it appears that 10^{-4} is a reasonable threshold value for the convergence criterion.

The influence of convergence criteria on the simulation results has also been investigated. The value of maxD = 10^{-4} has been used in this study. A smaller value has a minimum effect, whereas a larger value, say 2 x 10^{-4}, indicates significant discrepancy in the results obtained, particularly further away from the building, as shown in Figure 5.20. Results obtained by using the upwinding scheme are not too different from those produced by using the hybrid scheme – see Figure 5.20.

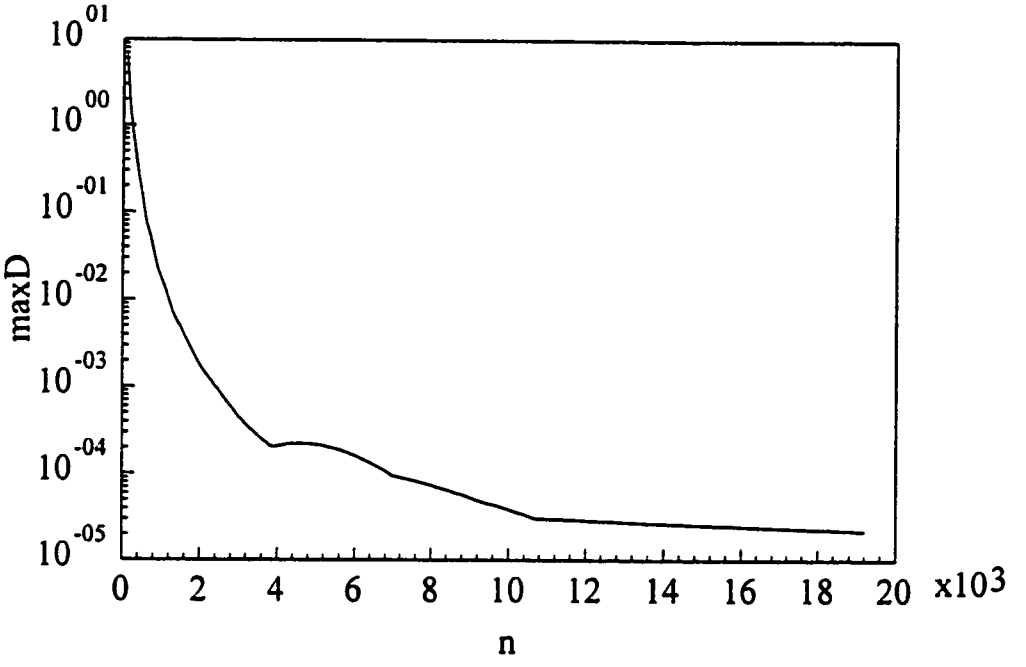


Figure 5.19 Difference of iterates vs. iteration steps in solving the concentration field

In order to examine the discretization error, a second mesh system with doubling the number of grids in three directions i.e. 62x80x34 was used. Results of longitudinal ground-level concentrations along the centerline also appear in Figure 5.20. The finer mesh system results follow the trend of the experimental data better than those of the coarse mesh system. A cubic spline interpolation algorithm based on the cubic piece-wise polynomial approximation with continuous first and second derivatives at each grid point is used to interpolate the coarse grid results to the fine grid. Then the error is estimated by Eq. (5-5). The maximum discretization error for the longitudinal ground-level concentrations is 14%. For most of the grid points the discretization error of those concentrations is around 5%.

5.6 Summary

The results of a detailed and comprehensive numerical study of the pollutant concentration field around a rectangular building were presented, analyzed and discussed.

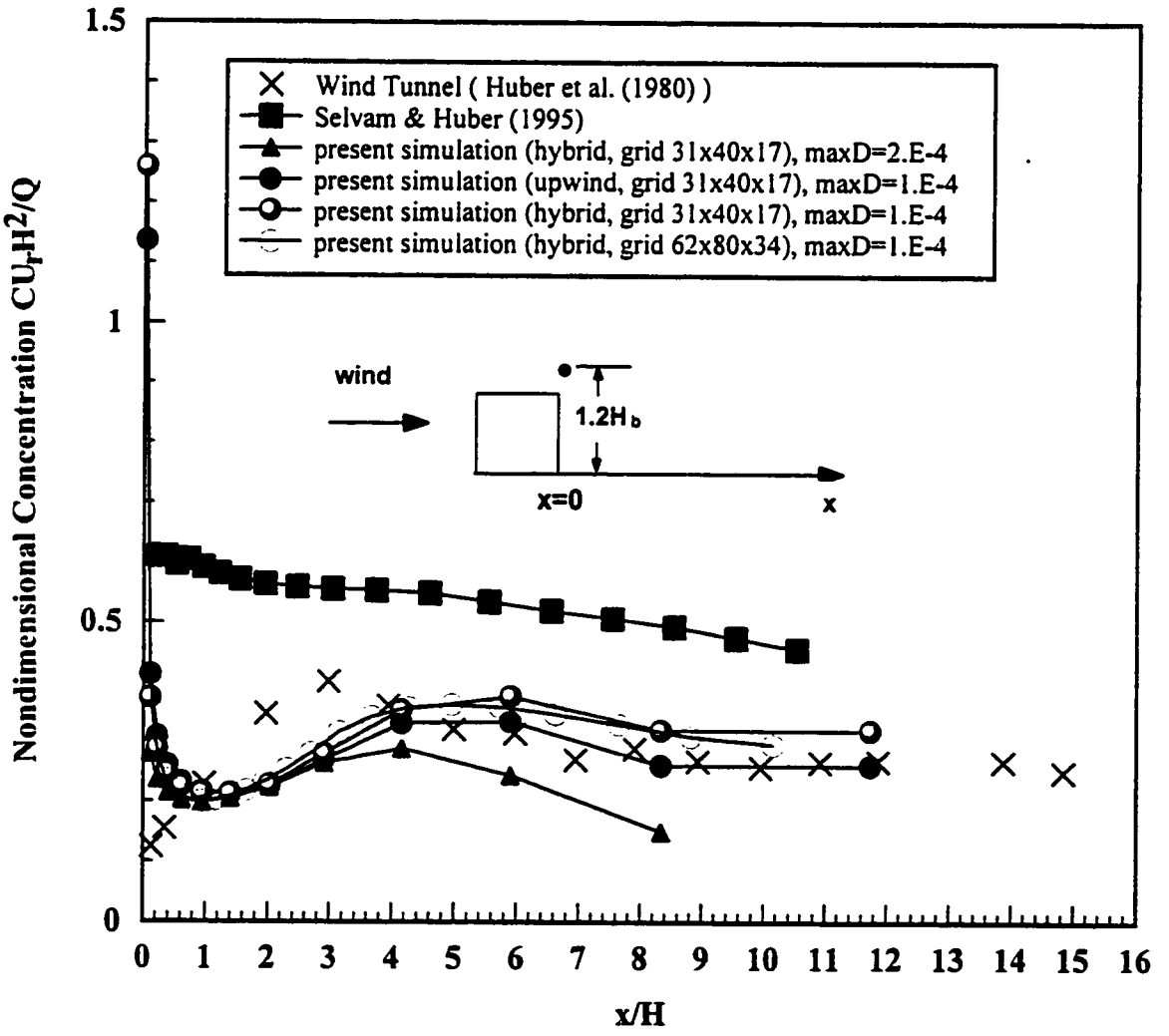


Figure 5.20 The effects of discretization and convergence criteria on longitudinal ground-level concentrations

The results from numerical simulation are in good agreement with those from the experiment and the Modified Gaussian Model. The numerical simulation results agrees very well with the experimental results near the ground while it is less satisfactory for higher heights. It seems that the present simulation predicts better than the Modified Gaussian Model for the near and medium building wake near the ground.

The influence of numerical errors on the computed results and the effects of thresholds in the discretization and convergence process have been investigated. Discretization errors were found to be less than 14% for the ground level dispersion of the case investigated.

Chapter 6

FLOW FIELD AND CONCENTRATION FIELD AROUND A CUBIC BUILDING: A TWO-LAYER APPROACH

6.1 Introduction

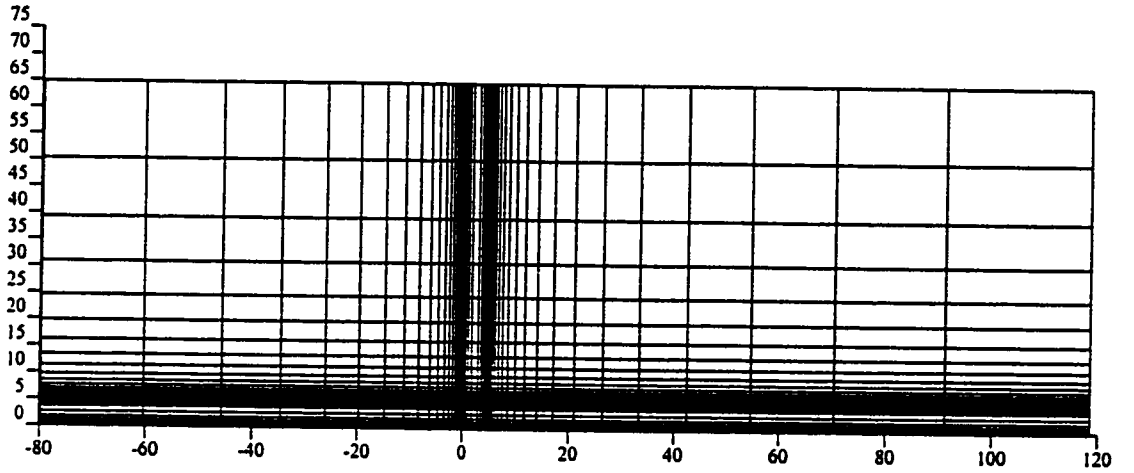
In this chapter, the pollutant dispersion around a cubic building has been evaluated by a two-layer approach. The results have been compared with those from the standard k-ε model. The experimental results from Li and Meroney (1983) were used to evaluate the numerical solutions.

6.2 Numerical Simulation Set-up

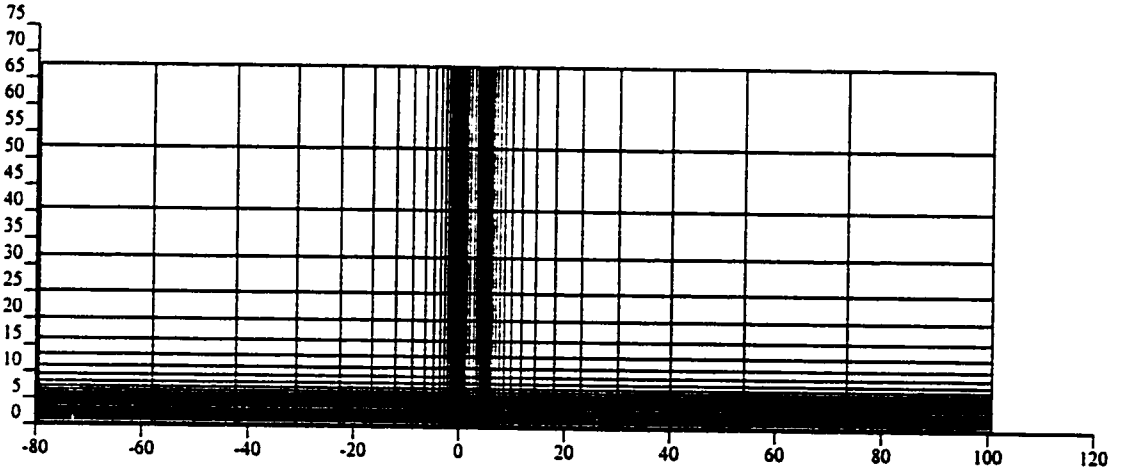
The numerical simulation set up represented the experimental conditions of Li and Meroney (1983). A cubic model building 5 cm x 5 cm x 5 cm was placed in a simulated atmospheric boundary layer flow with an oncoming velocity profile with a power-law exponent equal to 0.19. The speed at building height was 330 cm/s and the orientation of the wind was $\theta = 0^{\circ}$.

In order to evaluate the effect of the two-layer approach on the concentration values, the numerical simulation results from both standard k-ε model and two-layer

approach were compared by using similar computational domains and grid systems, as shown in Figure 6.1.



standard k-epsilon model, grid 52x52x33, x-z plane



two-layer approach, grid 54x58x33, x-z plane

Figure 6.1 The x-z plane grid outline

For the standard k-ε model, a grid of 52 x 52 x 33 extended to 225 cm x 155 cm x 75 cm was used, which is similar to the grid system in Figure 4.5 except that the computational domain is larger.

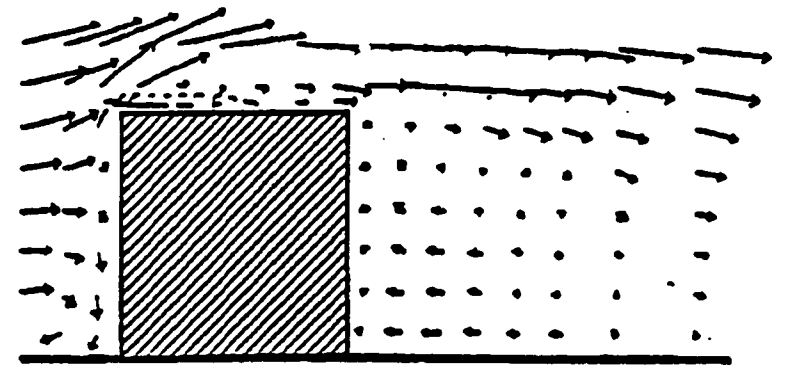
A grid 54 x 58 x 33 extended to 225 cm x 155 cm x 75 cm was used for the simulation in the case of the two-layer approach. Along x-direction, there are 20 grid points in front of the building with 7 of them inside the inner layer, 14 grid points along the building length span and 20 grid points behind the building with 7 of them inside the inner layer. Along y-direction, there are 20 grid points each from side wall of the building to the side of the computational domain with 7 grid points inside each inner layer. There are 18 grid points across the building width. Along z-direction, there are 13 grid points along the building height and 19 grid points from the roof of the building to the top of the computational domain with 7 of them inside the inner layer.

6.3 Flow Field around the Cubic Building

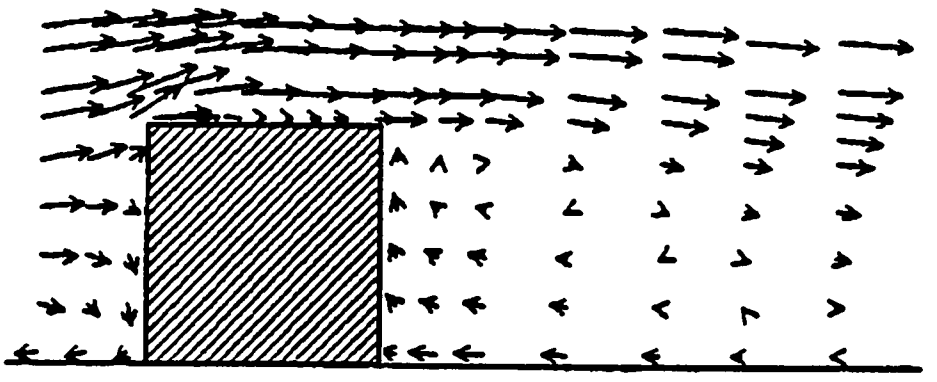
The flow field around the cubic building is investigated first because of its importance in the simulation of the concentration field.

Figure 6.2 shows the velocity vector distributions on the vertical section of a cube from Tsuchiya et al. (1996). Part (a) is the wind tunnel experiment result and part (b) is the numerical simulation result from standard k-ε model. In their simulation, a composite grid system composed of a fine and a coarse grid is used; the coarse grid is 30 x 30 x 30, the fine grid is 30 x 30 x 22. The total grid is 60 x 60 x 52.

Although the experimental results show the separation region on the roof, the numerical results from the standard k-ε model do not seem to predict it.



(a) wind tunnel test

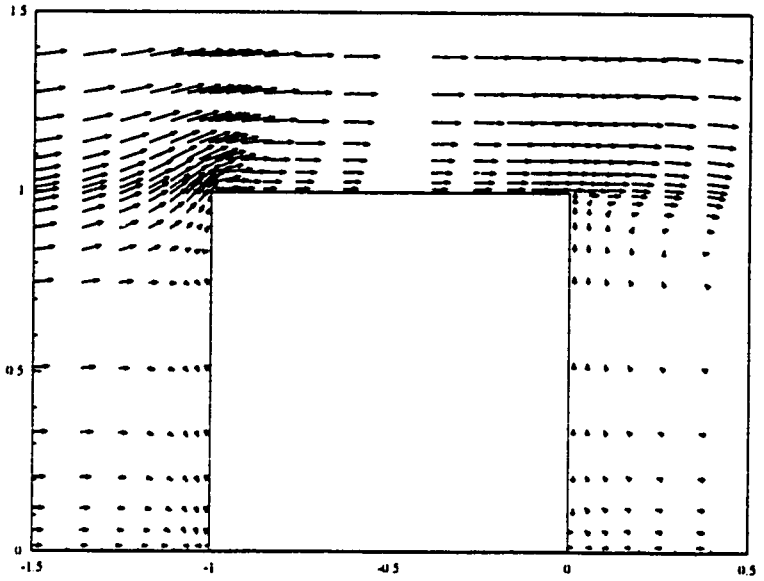


(b) standard k-ε model

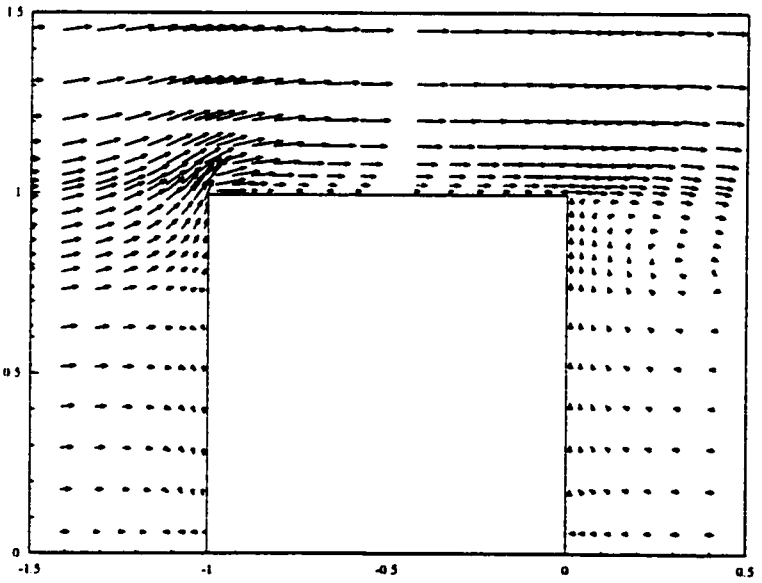
Figure 6.2 Velocity vector distributions on the vertical section of the cube, after Tsuchiya et al.(1996)

Figure 6.3 shows the velocity field at the longitudinal center plane around the front corner of a cubic building from the present simulation. The case using k-ε model

does not show any separation at the front corner so that no reverse flow was detected on the roof.



standard k-ε model, grid 52x52x33



two-layer approach, grid 54x58x33

Figure 6.3 The velocity field at the longitudinal center plane around the front corner of a cubic building

The case using the two-layer approach successfully predicts the reverse flow on the roof, indicating that there is separation at the front corner.

Figure 6.4 shows the comparison of vertical velocity profile above the roof of the building from the simulation study based on the two-layer approach and the standard k-ε model. The result from standard k-ε model does not have any negative velocity while the result from two-layer approach has negative velocity near the roof.

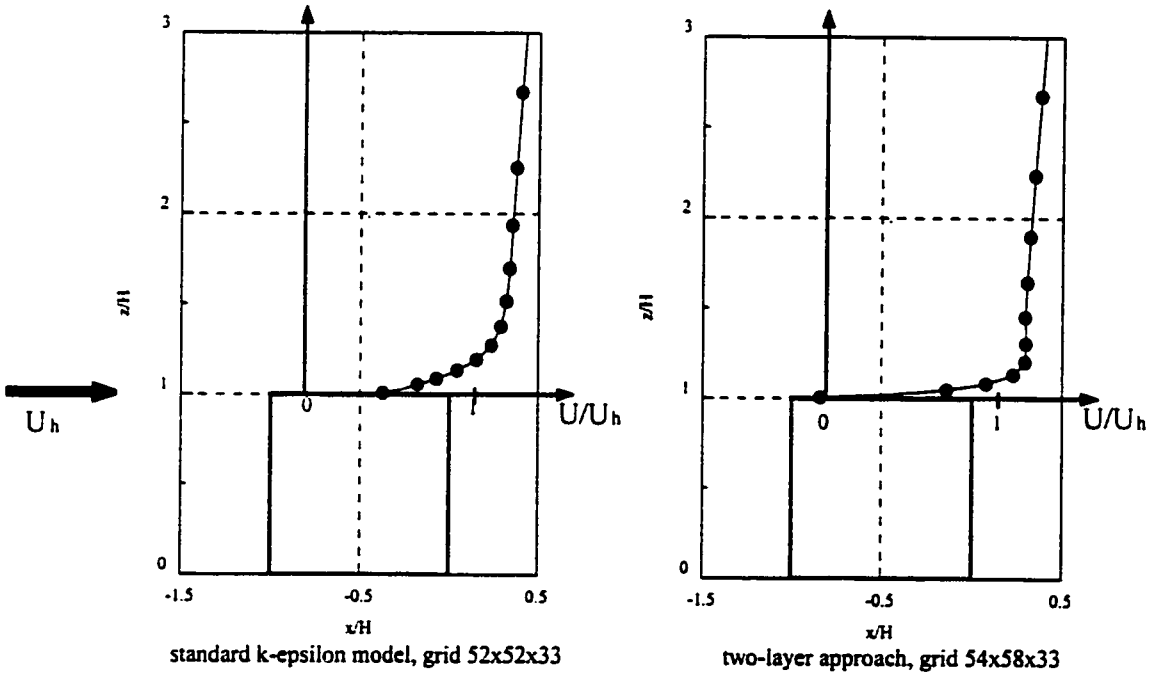


Figure 6.4 Vertical velocity profile above the roof

Figure 6.5 shows the vertical section of velocity vector field in the wake of the building. The reattachment length for the two cases are: 1.85h for k-ε model and 1.66h for the two-layer approach. Comparing with the cavity reattachment length of 1.4h from

the experiment (Snyder and Lawson, 1994) under similar conditions, the two-layer approach appears advantageous in comparison with the standard k-ε model.

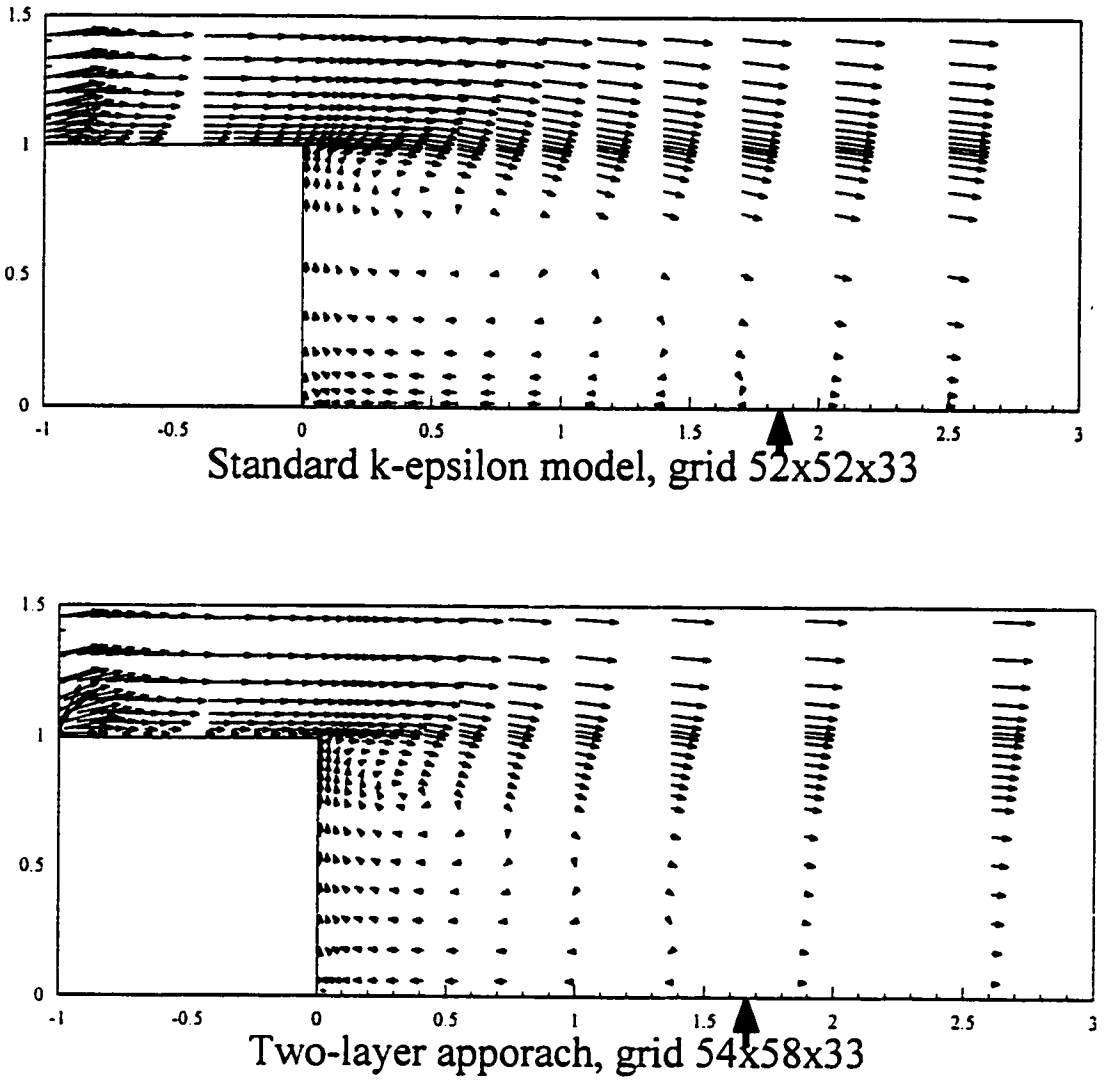


Figure 6.5 Velocity vector field in the wake of a cube

6.4 Mean Pressure Field on the Surface of a Cubic Building

Figures 6.6 and 6.7 show the mean pressure on the surface of a cube. The present simulation results are compared with the experimental data from Castro and Robins (1977). From the pressure distribution on the vertical plane, it appears that at the rear wall, the results from two-layer approach compare better with the experiment data. The same trend happens for the mean pressure on the horizontal plane. However, on the front wall, this is not the case. Generally speaking, the two-layer approach predicts results better than the standard k-ε model.

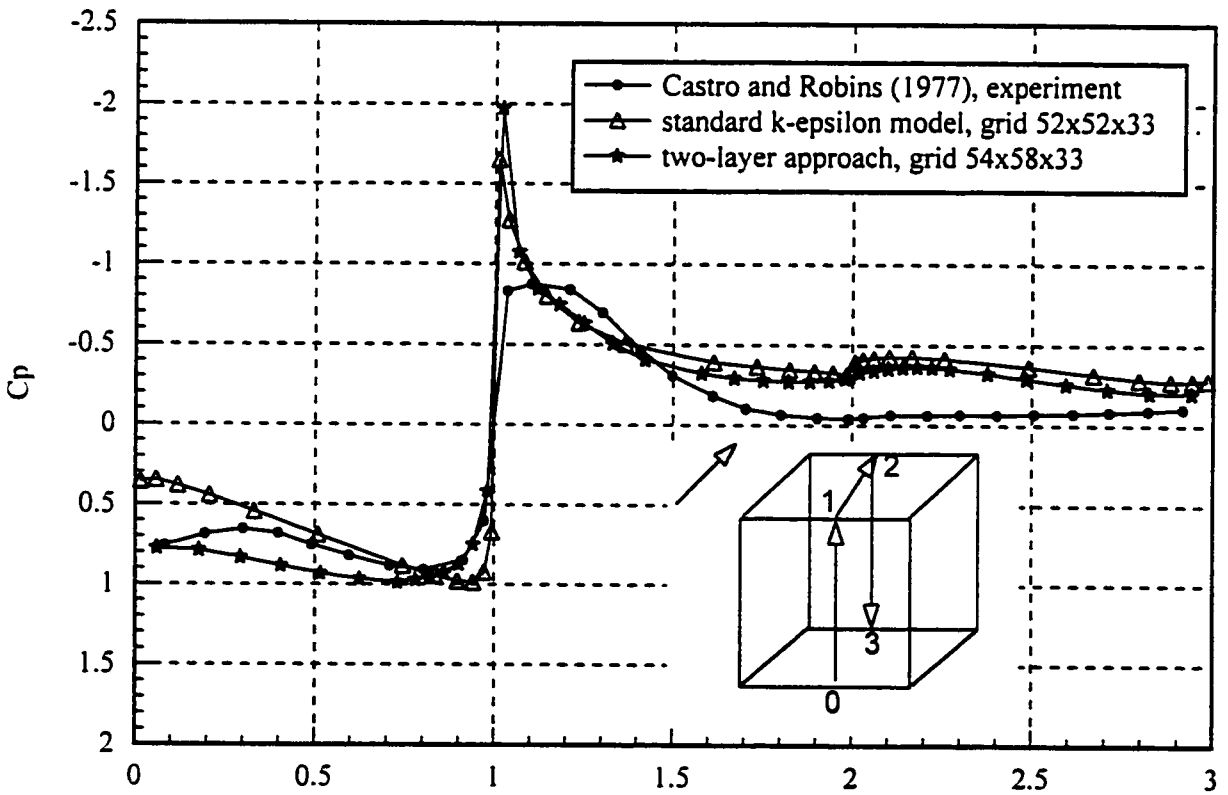


Figure 6.6 Surface pressure distribution on the vertical plane

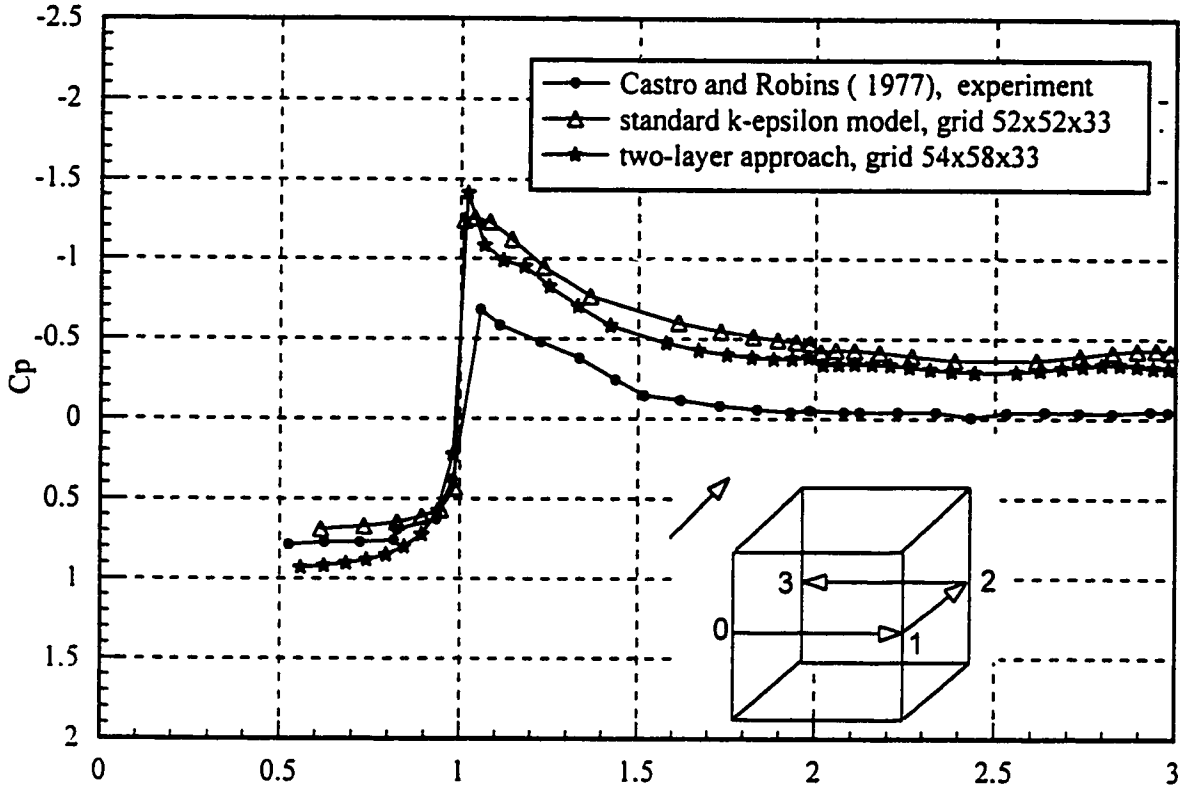


Figure 6.7 Surface pressure distribution on the horizontal plane

6.5 Concentration Field around the Cubic Building

The results and discussion presented in this section are divided into two categories. All the simulation results are compared with the experimental results from Li and Meroney (1983). Figure 6.8 illustrates the different vent locations and the definition of the coordinate system.

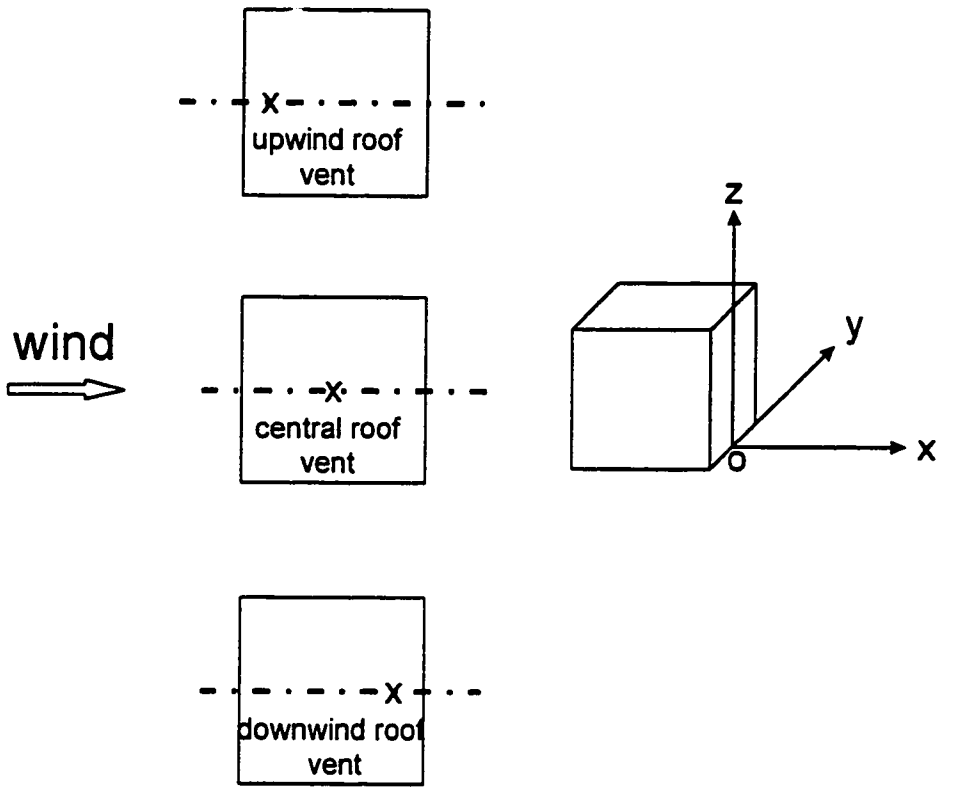


Figure 6.8 Sketch of the vent locations and the coordinate system

6.5.1 Concentration field on the building surface

Figure 6.9 shows contours of concentration coefficient K on the roof of a cubic building. Only half of the roof is presented because of the symmetry. The wind comes from the left side of the building at zero degree orientation. Both simulations from the standard k-ε model and the two-layer approach show larger roof areas than those of the

experiment influenced by K more than 100; however, the area from the two-layer approach is smaller than that from the standard k-ε model.

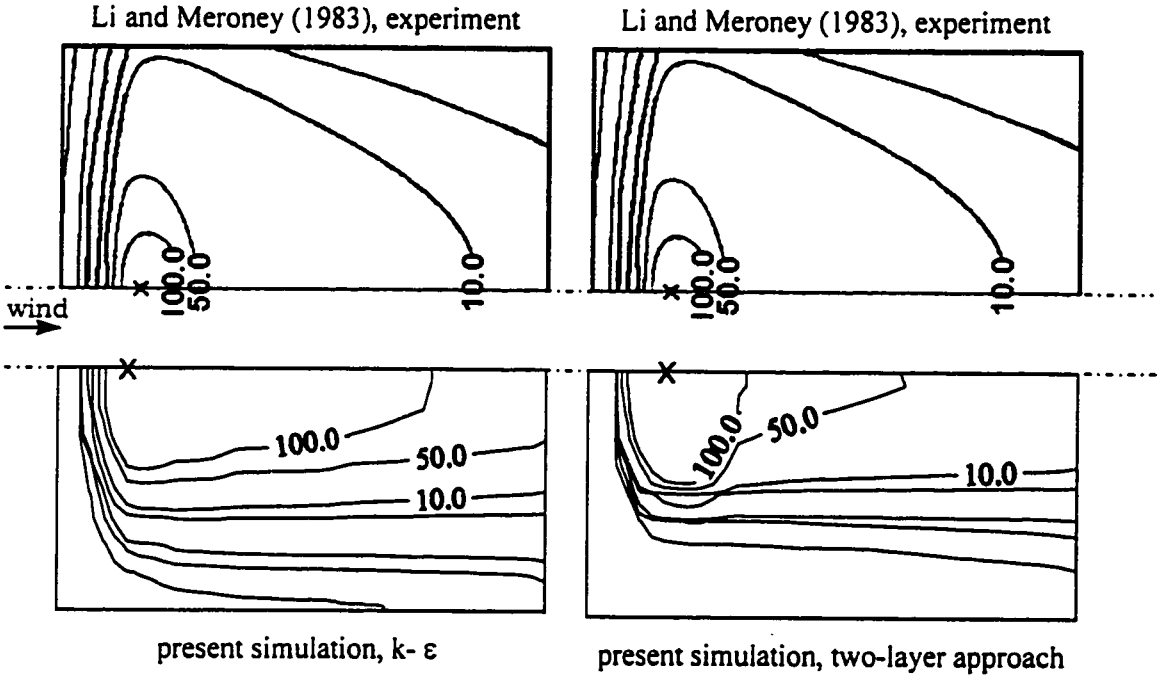


Figure 6.9 The concentration field on the roof of a cubic building (upward vent)

It also appears that along the wind direction, results from the two-layer approach are better than those from the standard k-ε model. At the centerline, the points where K is between 50 and 100 are simulated more accurately with the two-layer approach than with the k-ε model. However the stretch in crosswind direction of K contours from the two-layer approach is smaller than that from the standard k-ε model, which, in turn, is smaller than that from the experiment. Again, this may be attributed to the characteristics of the isotropic turbulence model. Overall, it would be difficult to claim, based on the roof concentration predictions, that the two-layer method is completely adequate. This

observation also agrees with Delaunay et al. (1996), who used the Rodi two-layer approach (Rodi, 1991). Clearly, although the separation near the front corner is predicted by the two-layer approach, other details of the flow and dispersion mechanisms are not predicted adequately. However, this may not be the case for other surfaces of the building.

Figure 6.10 shows the contours of concentration coefficients K on the back wall of a cubic building. Only half of the back wall is presented because of symmetry. The experimental results show that K values in most of the area of the back wall lie between 0.5 and 1.0, which was well predicted by the two-layer approach. The results from the standard k-ε model lie mostly between 1.0 and 5.0.

The K value near the center of the top edge of the back wall is around 5.0 from the experiment and this is well predicted by the two-layer approach; however, it is predicted twice as high from the standard k-ε model.

Overall, it is clear that the two-layer approach provides better concentration results on the back wall of the cubic building in comparison with those obtained by the standard k-ε model.

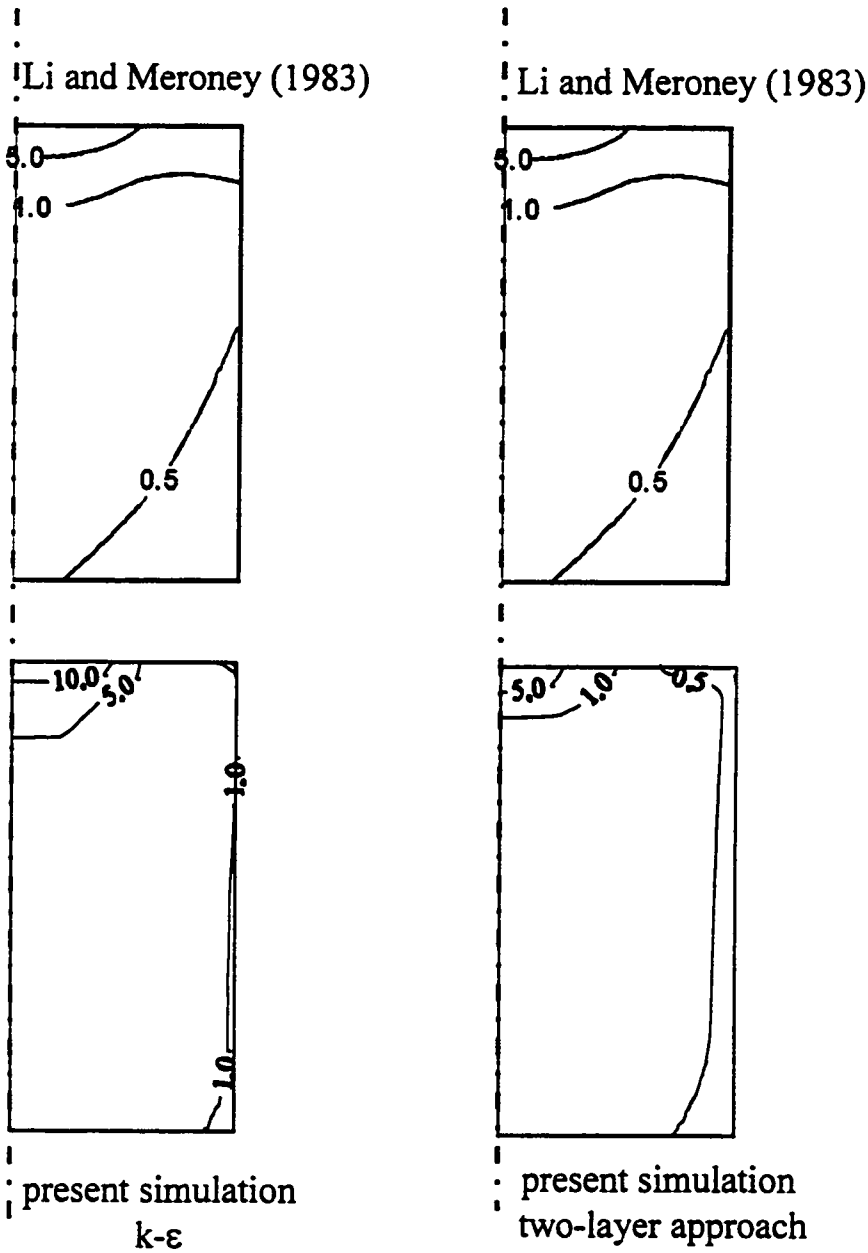
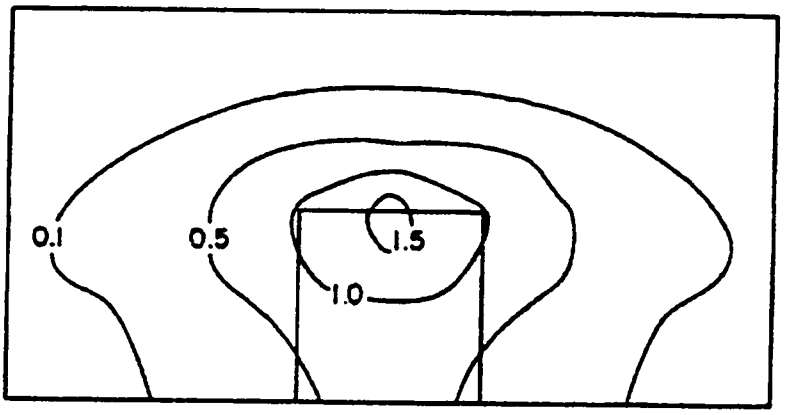


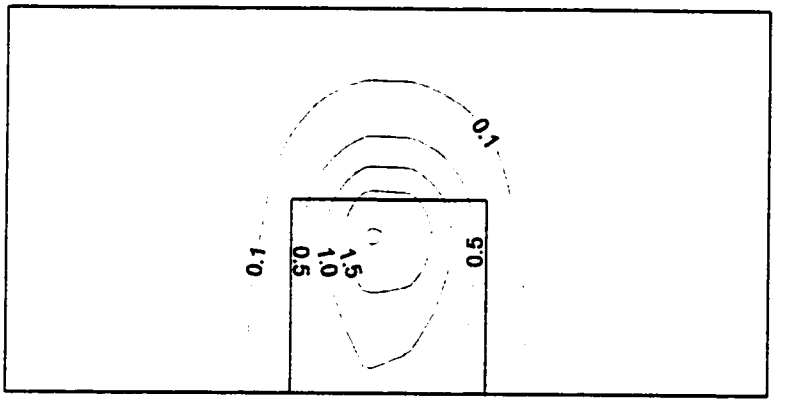
Figure 6.10 The concentration field on the rear wall of a cubic building (upward vent)

6.5.2 Concentration field in the near-wake region behind the building

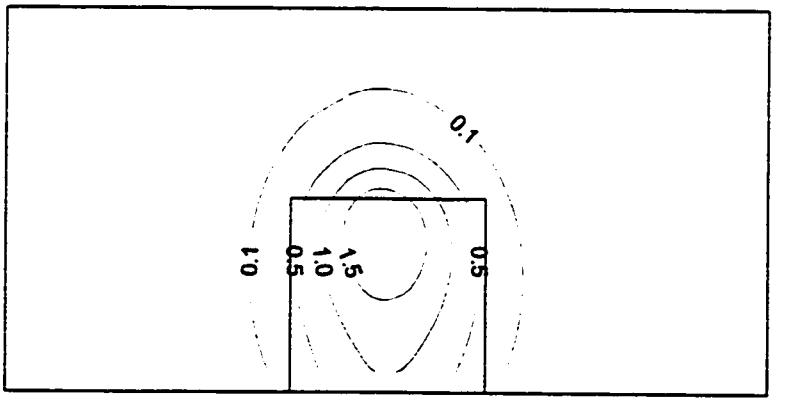
Figures 6.11-6.13 show the numerical simulation results of concentration-coefficient isopleths in the wake region for the central roof vent release. Li and Meroney's experimental results are also shown for comparison. The K contours were plotted on the y-z plane for different x/H sections. The values of the contours are 0.1, 0.5, 1.0, 1.5, 2.0 respectively. The numerical result shows that K isopleths tend to form closed continuous curves with their centers near the vent height. The values of K decrease from center to the outer part of the contours. For the two-layer approach, the values at the center show good agreement with the experiment while the values predicted by standard k-ε model are larger. The gradient of the contours appears bigger than that of the experiment, which is probably due to the lateral diffusion coefficient in the numerical simulation being less than that from the experiment. With the increase of x/H, the contours expand in y and z directions and the values at the center decrease, which shows the convection and diffusion of the pollutants from the released point to downstream of the building. Generally, the values of the contours show good agreement with Li and Meroney's experiment except at locations far from the center, where lower concentration values were computed. Again, this could be attributed to the lower lateral diffusion in the numerical simulation.



Li and Meroney (experiment, 1983)

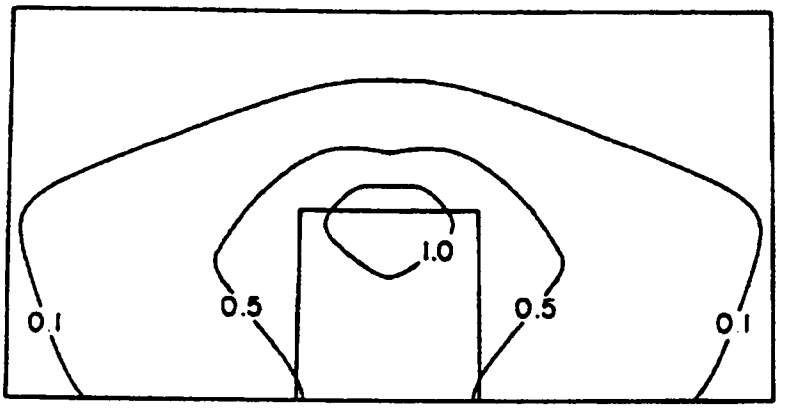


standard k-ε model

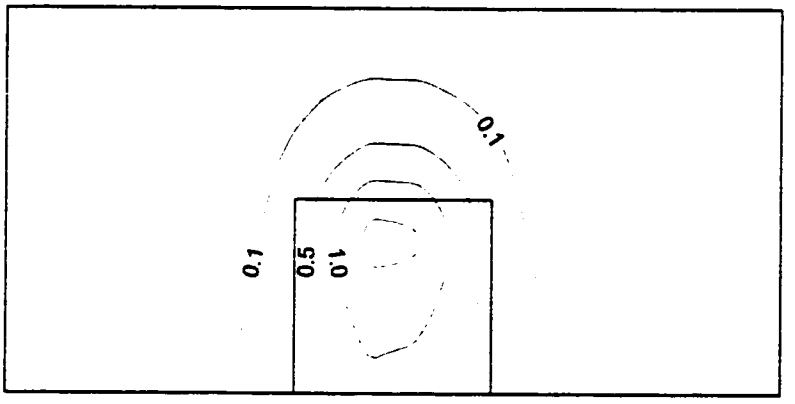


two-layer approach

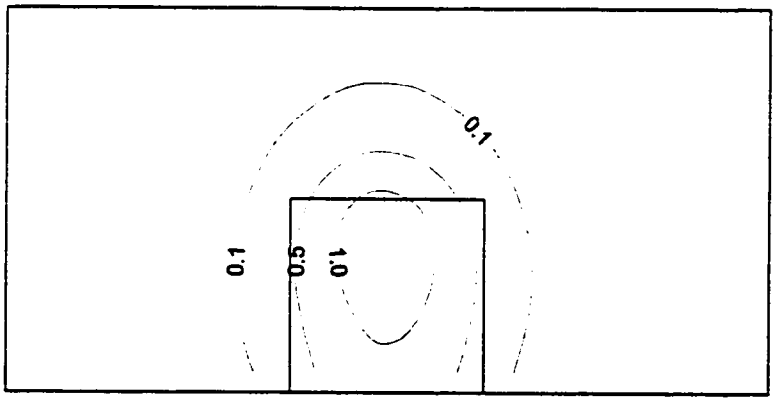
Figure 6.11 Concentration coefficient contour in near-wake region for central roof vent release at x/H = 2.0



Li and Meroney (experiment, 1983)

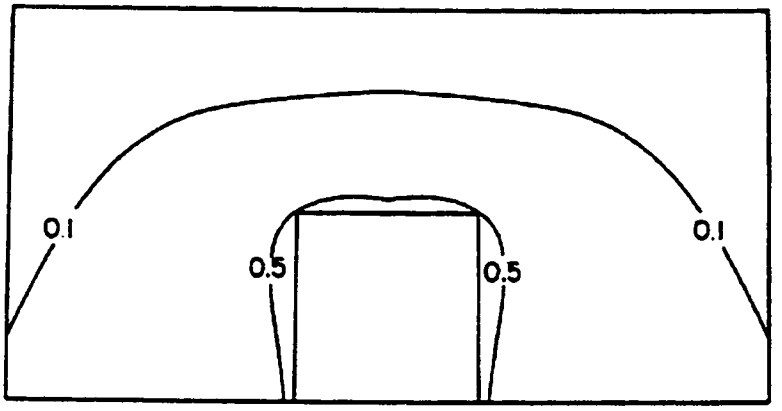


standard k-ε model

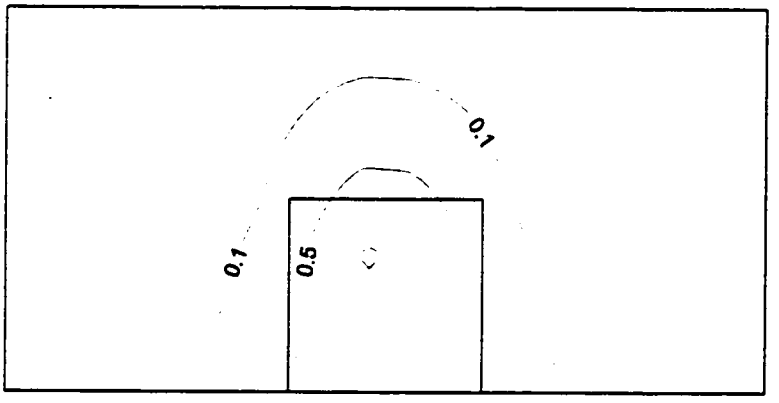


two-layer approach

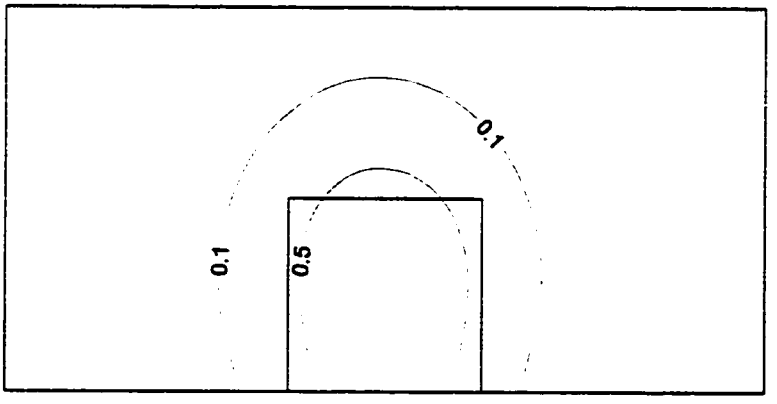
Figure 6.12 Concentration coefficient contour in near-wake region for central roof vent release at x/H = 2.5



Li and Meroney (experiment, 1983)



standard k-ε model



two-layer approach

Figure 6.13 Concentration coefficient contour in near-wake region for central roof vent release at x/H = 3.5

Figures 6.14-6.17 show the vertical mean concentration profiles at y/H = 0.0 and down-wind roof vent release at various x locations. At x/H = 0.5, the results from the two-layer approach agree very well with the experiment near the ground (less than half building height), but are less satisfactory around the building height. The results from standard k-ε model do not agree with the experiment as well as the two-layer approach. However, it is to be noted that the uncertainty of experiment curves is rather high considering that they are based on very limited number of points.

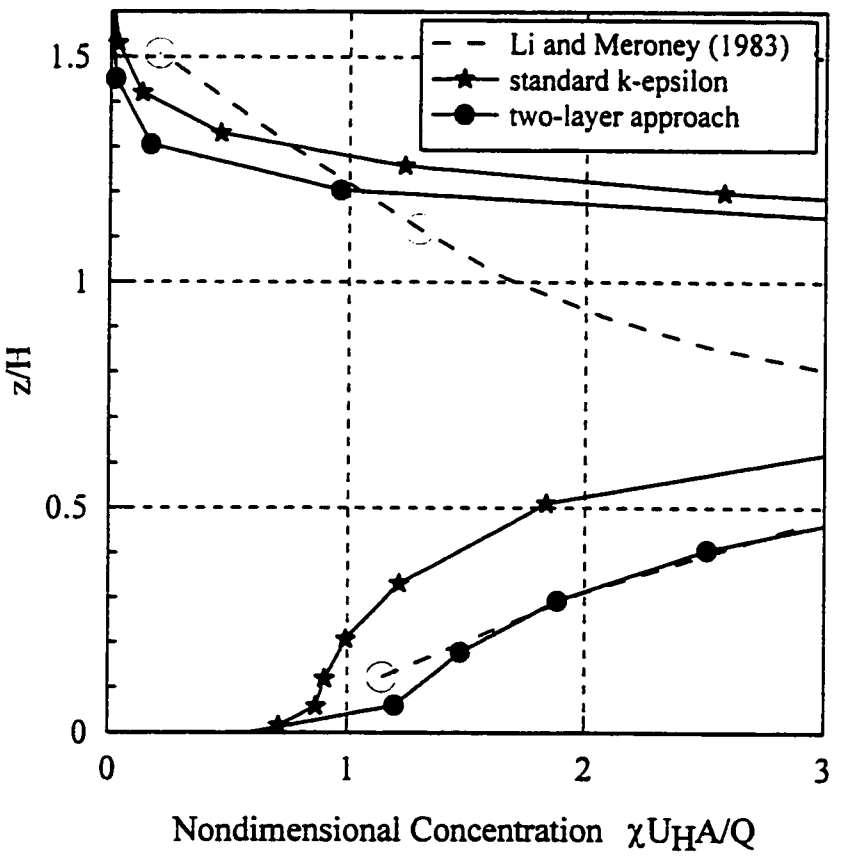


Figure 6.14 Vertical concentration profiles at y/H = 0.0, x/H = 0.5 for downwind roof vent release

At x/H = 2.5, the experimental results show that the concentration is around 1.0 near the ground, then gradually increases to maximum at around half building height; then it decreases with the increase of the height. The present simulation predicted this trend very well with the two-layer approach slightly better than the standard k-ε model.

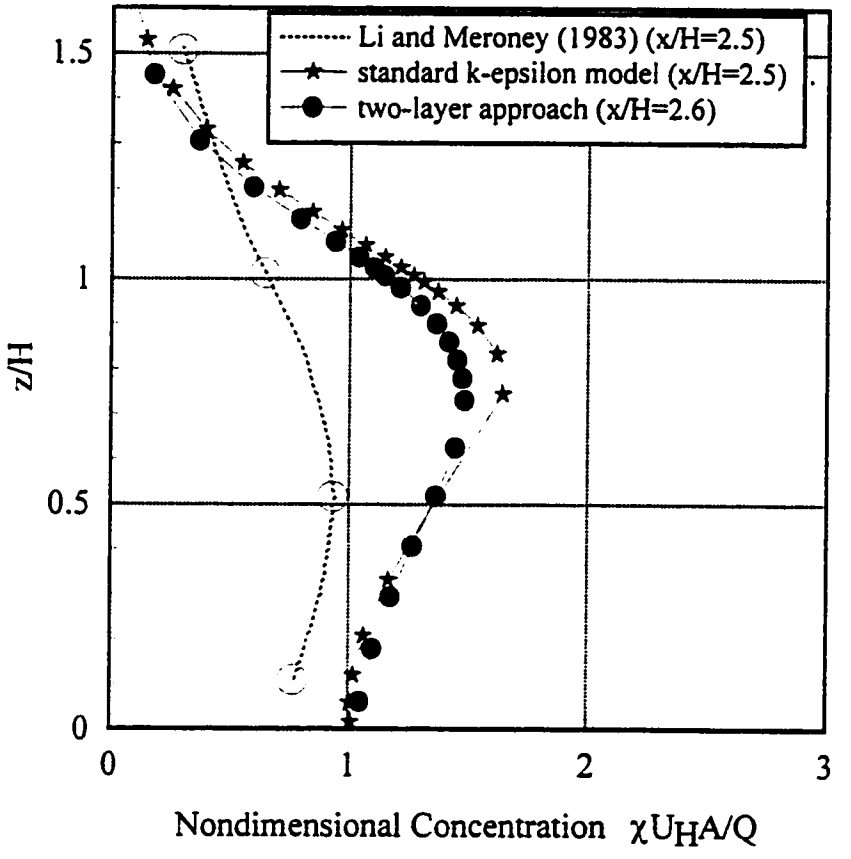


Figure 6.15 Vertical concentration profiles at y/H = 0.0, x/H = 2.5 for downwind roof vent release

At x/H = 3.5, the results from the standard k-ε model and the two-layer method are very similar. They all predicted the shape of the concentration profile well.

At x/H = 4.5, the present simulation results agree well with the experimental data; the two-layer approach appears better than the utilization of the standard k-ε model.

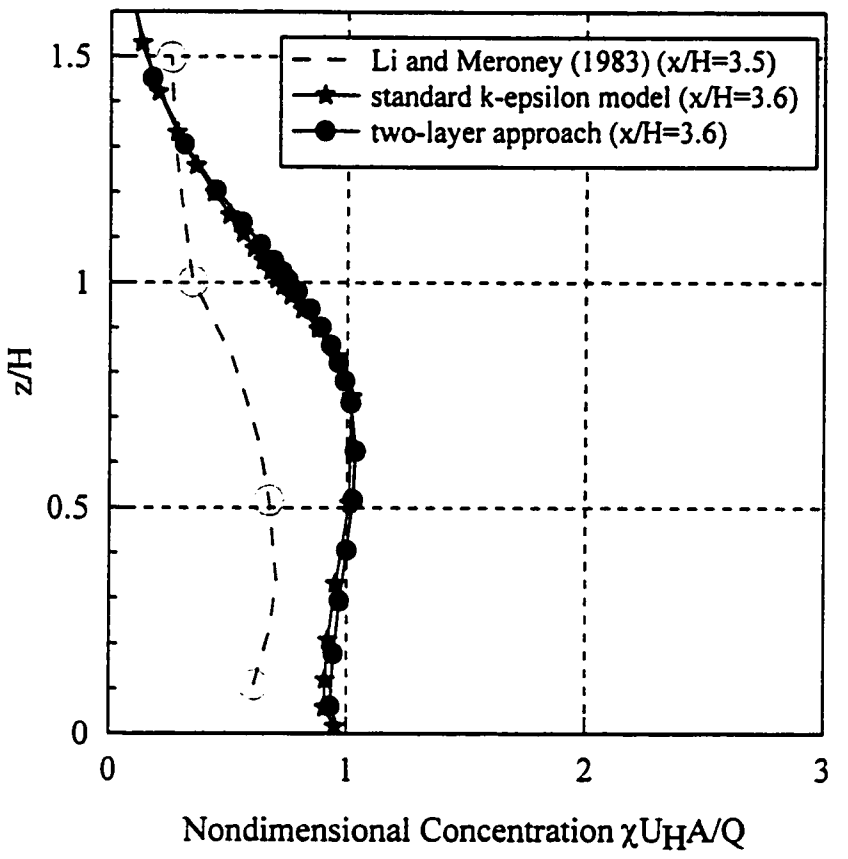


Figure 6.16 Vertical concentration profiles at y/H = 0.0, x/H = 3.5 for downwind roof vent release

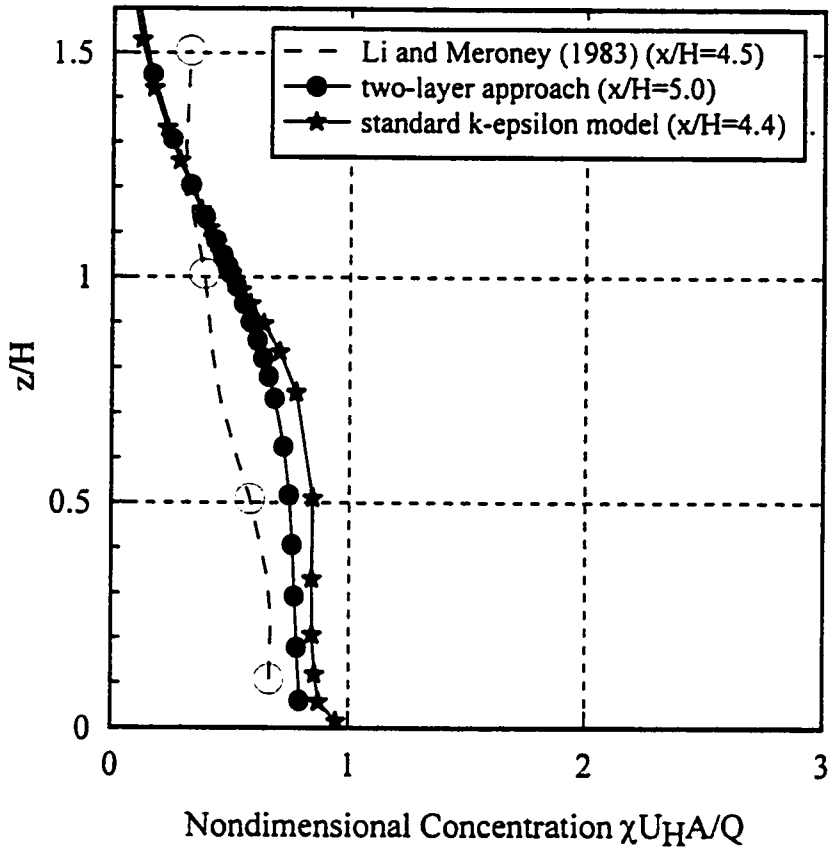


Figure 6.17 Vertical concentration profiles at y/H = 0.0, x/H = 4.5 for downwind roof vent release

6.6 Summary

The successful prediction of recirculating flow on the roof by a two-layer approach gives better along-wind concentration than the standard k-ε model.

The better flow prediction by the two-layer approach on the roof leads to a better simulation of the wake region, which in turn gives better simulation results of concentration on the back wall of the building.

However the two-layer approach does not seem to work well for roof concentration along the cross-wind direction; this is probably due to the drawback of the isotropic turbulence model used.

Chapter 7

CLOSURE

7.1 Concluding Remarks

A detailed and comprehensive study of wind-induced pollutant dispersion around buildings has been carried out. Research in this area was reviewed and analyzed. The results from various studies were compared. Based on the review and the comparison with various studies, a more effective methodology for the numerical evaluation of dispersion of pollutants around buildings has been proposed. The results from this two-layer approach were evaluated in order to show the effectiveness of this method.

Computed results of the two-layer approach shows various degrees of improvement for the concentration field in the wake region and on the back wall of the building. Results from two-layer approach also show improvement on the building roof for along-wind direction. However the two-layer approach does not seem to work well for roof concentration along the cross-wind direction; this is probably due to the drawback of the isotropic turbulence model used.

The previous numerical studies on pollutant dispersion around rectangular buildings only investigated the concentration profiles for along-wind direction or vertical direction, and these were only for limited situations. In the present study, a systematic

examination of pollutant concentration in the wake of a rectangular building has been carried for two different release sources. The concentration profiles have been investigated for along-wind, across-wind and vertical directions for various locations. The numerical simulation results have been compared with those from the experiments and from the Modified Gaussian Model. Results from numerical simulation generally agree well with those from the experiments. It was found out that the numerical simulation predicts better than the Modified Gaussian Model inside the near building wake at low heights.

The influence of numerical errors on the computed results and the effects of thresholds in the discretization and convergence process have been investigated, for the first time in this problem. Discretization errors were found to be less than 15% for the ground level dispersion of the case investigated.

7.2 Research Contributions

- A systematic numerical study of pollutant dispersion around a rectangular building was carried out and results were compared extensively with experimental data.

- In order to avoid the drawbacks of the standard k-ε model, a two-layer approach was developed to simulate the pollutant dispersion around a cubic building. The new approach shows improvement of simulation results, particularly in the wake region.

•Designed two new codes to simulate the pollutant dispersion around buildings.

- The influence of numerical errors on the computed results has been investigated for the simulation of pollutant dispersion around buildings; this appears for the first time in the literature of computational wind engineering.

7.3 Recommendations for Further Work

In spite of the significant effort put on the code development, the present study of numerical simulation of pollutant dispersion around buildings has only managed to survey limited cases. However, the two new codes developed and the error estimation method introduced in the course of present simulation provide a solid foundation for future research and investigation. A number of improvements to the present approach can be suggested and may be summarized as follows:

- In the present study, only normal wind conditions and simple rectangular and cubic building geometry have been investigated for the air pollutant dispersion around buildings. Further study of oblique wind direction and more complex building geometry like Z-, L- shaped and even groups of buildings may help obtain a better and more complete understanding of pollutant dispersion around buildings.
- The error estimation in the present study, is still very primitive. The influence of convergence error has been investigated but the convergence error itself has not been accurately quantified like the discretization error. All errors have been estimated only

in a limited area around the building. Further research in quantifying the numerical convergence error and extending the error estimation to the whole computation domain should give a much more reliable numerical simulation result and may make significant improvement in the robustness of the numerical simulation.

- In the present study, the popular upwind, hybrid and power-law scheme were used to discretize the advection term. More advanced numerical discretization schemes like skew-upwind and QUICK could be used to improve the numerical simulation results.
- A more advanced turbulence model such as Reynolds Stress Model may improve the numerical simulation results and it should be examined.
- The turbulent concentration flux term has been approximated by assuming gradient transport and constant Prandtl number leading to equation (3-10). In the present study, it was found out that lateral diffusion has been undervalued, which could have been caused by this assumption. Further study investigating the mechanism of pollutant dispersion may lead to a better estimation of the turbulent concentration flux, and this could improve numerical simulation results significantly.

REFERENCES

A.A. Amsden and F.H. Harlow, The SMAC Method: A Numerical Technique for Calculating Incompressible Flows, *Report LA-4370, Los Alamos Scientific Laboratory,* Los Alamos, NM, 1970

ASHRAE handbook - 1997 fundamentals

B. Basara and B.A. Younis, Progress in the prediction of turbulent wind loading on buildings, *J. Wind Engrg. & Ind. Aerodyn., 41-44,* 2863-2874, 1992

A. Baskaran and T. Stathopoulos Computational Evaluation of Wind Effects on Buildings, *Building & Environment 24(4),* 325-333, 1989

M.A. Brzoska, D. Stock and B. Lamb, Three Dimensional Finite Element Numerical Simulations of AirFlow and Dispersion around Buildings, *9th AMS Conf. On Appl. Air Pollution Meteorology, with A&WMA,* Atlanta, Georgia, January 28 - February 2, 1996

T.D. Bui and A.K. Oppenheim, Ealuation of Wind Effects on Model Buildings by the Random Vortex Method, *Applied Numerical Mathematics 3,* 195-207, 1987

D.R. Chapman, Computational Aerodynamics Development and Outlook, *AIAA Journal 17(12),* 1293-1313, 1979

I.P. Castro and A.G. Robins, The Flow Around a Surface-Mounted Cube in Uniform and Turbulent Streams, *J. Fluid Mech., 79(2),* 307-335, 1977

I. Celik, Numerical Uncertainty in Fluid Flow Calculations: Need for Future Research, *J. Fluids Engineering, 115,* 194-195, 1993

I. Celik and Wei-Ming Zhang, Application of Richardson Extrapolation to Some Simple Turbulent Flow Calculations, *FED-Vol. 158, Quantification of Uncertainty in Computational Fluid Dynamics, ASME 1993,* 29-38, 1993

I.R. Cowan, I.P. Castro and A.G. Robins, Numerical Considerations for Simulations of Flow and Dispersion around Buildings, *J. Wind Engrg. & Ind. Aerodyn. 67&68,* 535-545, 1997

A.G. Davenport, The Relationship of Wind Structure to Wind Loading, *Paper 2. Proc. Conf. on Wind Effects on Buildings and Structures,* (Nat. Phys. Lab., H.M.S.O., London, June 1965) 54-102.

P. Dawson, D.E. Stock and B. Lamb, The Numerical Simulation of Airflow and Dispersion in Three-Dimensional Atmospheric Recirculation Zones, *Journal of Applied Meteorology 30,* 1005-1024, 1991

W.A. Dalgliesh, Statistical Treatment of Peak Gusts on Cladding, *J. Structral Div., Proc. ASCE 97(ST9),* 2173-2187, 1971

D. Delaunay, D. Lakehal, C. Barre and C. Sacre, Numerical and Wind-Tunnel Simulation of Gas Dispersion around a Rectangular Building, *J. Wind Engrg. & Ind. Aerodyn. 67&68,* 721-732, 1997

A.O. Demuren and R.V. Wilson, Estimating Uncertainty in Computations of Two-Dimensional Separated Flows, *FED-Vol. 158, Quantification of Uncertainty in Computational Fluid Dynamics, ASME 1993, 9-18,* 1993

J.P.V. Doormaal. and G.D. Raithby, Enhancements of the SIMPLE Method for Predicting Incompressible Fluid Flows, *Numerical Heat Transfer 7,* 147-163, 1984

E.J. Eaton, Cladding and the Wind, *J. Structural Div., Proc. ASCE 102(ST5),* 1043-1058, 1976

Sixin Fan, B. Lakshminarayana and M. Barnett, Low-Reynolds-Number k-ε Model for Unsteady Turbulent Boundary-Layer Flows, *AIAA Journal 31(10),* 1777-1784, 1993

J.H. Ferziger, A Computational Fluid Dynamicist's View of CWE, *J. Wind Engrg & Ind. Aerodyn. 46-47,* 879-880, 1993

J.H. Ferziger, Simulation of complex turbulent flows: recent advances and prospects in wind engineering, *J. Wind Engrg. & Ind. Aerodyn. 46-47,* 195-212, 1993

J.H. Ferziger, Estimation and Reduction of Numerical Error, *FED vol. 158, Symp. On Quantification of Uncertainty in Computational Fluid Dynamics, ASME Fluid Engineering Division, Summer Meeting, Washington, D.C., June 20-24,* 1-8, 1993

W. Frank, Large-eddy-simulation of the flow around building models, *J. Wind Engrg. & Ind. Aerodyn. 46-47,* 213-218, 1993

U. Frisch and S.A.Orszag, Turbulence: Challenges for Theory and Experiment, *Physics Today January 1990,* 24-32, 1990

A. Gadilhe, L. Janvier and G. Barnaud, Numerical and Experimental Modelling of the Tridimensional Turbulent Wind Flow through an Urban Square, *J. Wind Engrg. & Ind. Aerodyn. 52,* 231-236, 1992

K. Haggkvist, U. Svensson and R. Taesler, Numerical Simulations of Pressure Fields Around Buildings, *Building & Environment 24(1),* 65-72, 1989

J. Halitsky, Concentration Coefficient in Atmospheric Dispersion Calculations, *ASHRAE Transaction 91(2B),* 1722-36, 1985

J. Halitsky, Atmospheric dilution of fume hood exhaust gases, *American Industrial Hygiene Association Journal,* 1982

J. Halitsky, Gas Diffusion Near Buildings, *ASHRAE Transaction 69,* 464-485, 1963

Taeyoung Han, Computational Analysis of Three-Dimensional Turbulent Flow Around a Bluff body in Ground Proximity, *AIAA Journal 27(9),* 1213-1219, 1989

T. Hanson, D.M. Summers and C.B. Wilson, A Three-Dimensional Simulation of Wind Flow Around Buildings, *International J. Numerical Methods in Fluids 6,* 113-127, 1986

Jianming He and C.C.S. Song, Computation of Turbulent Shear Flow over Surface-Mounted Obstacle, *J. Engineering Mechanics 118(11),* 2282-2297, 1992

H.L. Higson et al., Concentration Measurements around an Isolated Building: A Comparison between Wind Tunnel and Field Data, *Atmospheric Environment 28(11),* 1827-1836, 1994

A.H. Huber, Wind Tunnel and Gaussian Plume Modeling of Building Wake Dispersion, *Atmospheric Environment 25A(7),* 1237-1249, 1991

A.H. Huber, Performance of a Gaussian Model for Cnetrerline Concentrations in the Wake of Buildings, *Atmospheric Environment 22(6),* 1039-1050, 1988

A.H. Huber, Evaluation of a Method for Estimating Pollution Concentration Downwind of Influencing Buildings, *Atmospheric Environment 18(11),* 2313-2338, 1984

A.H. Huber and **W.H. Snyder,** Wind Tunnel Investigation of the Effects of a Rectangular-Shaped Building on Dispersion of Effluent from Short Adjacent Stacks, *Atmospheric Environment 16(12),* 2837-2848, 1982

A.H. Huber, W.H. Snyder and R.E. Lawson, Jr., The Effects of a Squat Building on Short Stack Effluents: A Wind-tunnel Study, Report EPA-600/4-80-055, U.S. Environment Protection Agency, Research Triangle Park, N.C., 1980

G.E. Karniadakis and S.A. Orszag, Nodes, Modes and Flow Codes, *Physics Today March 1993,* 34-42, 1993

M. Kato and B.E. Launder, The Modelling of Turbulent Flow around Stationary and Vibrating Square Cylinders, *9th Symp. on Turbulent Shear Flows*, Kyoto, 10-4-1 — 10-4-6, 1993

S. Kato, S. Murakami, Y. Utsumi and K Mizutani, Application of Massive Parallel Computer to

Computational Wind Engineering, *J. Wind Engrg. & Ind. Aerodyn. 46-47*,393-400, 1993

S. Kawamoto and T. Tanahashi, High-speed GSMAC-FEM for wind engineering, *Computer Methods in Applied Mechanics and Engineering 112,* 219-226 1994

L.T. Kisielewicz and A. Tabbal, Validated computational aerodynamics for trains, *J. Wind Engrg. & Ind. Aerodyn.49,* 449-458, 1993

K. Kondo, S. Murakami and A. Mochida, Numerical Prediction of Flow field around 2D Square Rib Using Revised k-ε Model, *J. Wind Engrg. 63,* 35-36, 1995

T. Kurabuchi, J.B. Fang and R.A. Grot., A Numerical Method for Calculating Indoor Airflows Using a Turbulence Model, *National Institute of Standards and Technology, Report No.: NISTIR 89-4211,* 1990

B. Lamb and D. Cronn, Fume Hood Exhaust Re-entry into a Chemistry Building, *Am. Ind. Hyg. Assoc. J. 47(2),* 115-123, 1986

D. Lakehal and W. Rodi, Calculation of the Flow Past a Surface-Mounted Cube with Two-Layer Turbulence Models, , *J. Wind Engrg. & Ind. Aerodyn. 67&68,* 65-78, 1997

B.E. Launder and M. Kato, Modelling Flow-Induced Oscillations in Turbulent Flow around a Square Cylider, ASME Fluid Engrg. Conf., Washington DC, June 1993

B.E. Launder and D.B. Spalding, The Numerical Computation of Turbulent Flows *Comput. Methods Appl. Mech. Engrg. 3,* 269-289, 1974

D. Laurence and J-D. Mattei, Current State of Computational Bluff Body Aerodynamics, *J. Wind Engrg. & Ind. Aerodyn. 49,* 23-44, 1993

B. Leitl, P. Klin, M. Rau and R.N. Meroney, Concentration and Flow Distributions in the Vicinity of U-Shaped Buildings: Wind-Tunnel and Computational Data, *Second International Symposium on Computational Wind Engineering,* Fort Collins, Co., 1996

Wen-whai Li and R.N. Meroney, Gas Dispersion Near a Cubical Model Building. Part I. Mean Concentration Measurements, *J. Wind Engrg. & Ind. Aerodyn. 12,*15-33, 1983

S. Majumdar and W. Rodi, Three-Dimensional Computation of Flow Past Cylindrical Structures and Model Cooling Towers, *Building & Environment 24(1),* 3-22, 1989

E.H. Mathews and J.P. Meyer, Numerical Modelling of Wind Loading on a Film Clad Greenhouse, *Building & Environment 22(2),* 129-134, 1989

R.N. Meroney, Bluff-Body Aerodynamics Influence on Transport and Diffusion of Hazardous Gases: Shelterbelts and Vapour Barrier Fences, *J. Wind Engrg. & Ind. Aerodyn. 49,* 141-156, 1993

U. B. Meta, Some Aspects of Uncertainty in Computational Fluid Dynamics Results, *J. Fluids Engineering, 113,* 538-543, 1991

M.H. Mirzai et al., Wind Tunnel Investigation of Dispersion of Pollutants due to Wind Flow around a Small Building, *Atmospheric Environment 28(11),* 1819-1826, 1994

D. Moon, A. Albergel, F. Jasmin and Gerard Thibault, The Use of the MERCURE CFD Code to Deal with an Air Pollution Problem due to Building Wake Effects, *J. Wind Engrg. & Ind. Aerodyn. 67&68,* 781-791, 1997

S. Murakami, A. Mochida and S. Sakamoto, Numerical Study on Velocity-Pressure Field and Wind Forces for Bluff Bodies by k-ε, ASM and LES, *J. Wind Engrg. & Ind. Aerodyn. 41-44,* 2841-2852, 1992

S. Murakami, A. Mochida and Y. Hayashi, Numerical Simulation of Velocity Field and Diffusion Field in an Urban Area, *Energy and Buildings 15-16,*345-356, 1991

S. Murakami, A. Mochida and Y. Hayashi, Examining the k-ε Model by Means of a Wind Tunnel Test and Large-Eddy Simulation of the Turbulence Structure around a Cube, *J. Wind Engrg. & Ind. Aerodyn.35,* 87-100, 1990

S. Murakami, Computational Wind Engineering, *J. Wind Engrg. & Ind. Aerodyn.35,* 517-537, 1990

S. Murakami and A. Mochida, Three-dimensional Numerical Simulation of Turbulent Flow around Buildings Using the k-ε Turbulence Model, *Building and Environment, 24(1),* 51-64, 1989

S. Murakami, A. Mochida and K. Hibi, Three-Dimensional Numerical Simulation of Air Flow Around a Cubic Model by Means of Large Eddy Simulation, *J. Wind Engrg. & Ind. Aerodyn. 25,* 291-305, 1987

Y. Nakamura, Bluff-Body Aerodynamics and Turbulence, *J. Wind Engrg. & Ind. Aerodyn. 49,* 65-78, 1993

L.H. Norris and W.C. Reynolds, Turbulent Channel Flow with a Moving Wavy Boundary, Report No. FM-10, Stanford University, Department of Mechanical Engineering, Stanford, CA, USA, 1975

Y. Ogawa, S. Oikawa and K. Uehara, Field and Wind Tunnel Study of The Flow and Diffusion around a Model Cube - II. Nearfield and Cube Surface Flow and Concentration Patterns, *Atmospheric Environment 17(6),* 1161-1171, 1983

Y. Ogawa, S. Oikawa and K. Uehara, Field and Wind Tunnel Study of The Flow And Diffusion Around A Model Cube - I. Flow Measurement, *Atmospheric Environment 17(6),* 1145-1159, 1983

D.A. Paterson and J.D. Holmes, Computation of Wind Pressures on Low-rise Structures, *J. Wind Engrg. & Ind. Aerodyn. 41-44*, 65-78, 1992

D.A. Paterson and C.A. Apelt, Computation of Wind Flows Over Three-Dimensional Buildings, *J. Wind Engrg. & Ind. Aerodyn. 24*, 193-213, 1986

A.E. Perry et al., Some Recent Ideas and Developments in the Modeling of Wall Turbulence, *Near-Wall Turbulent Flows*, 1-25, 1993

J.A. Peterka, Selection of Local Peak Pressure Coefficients for Wind Tunnel Studies of Buildings, *J. Wind Engrg. & Ind. Aerodyn. 13*, 477-488, 1983

P.J. Richards and **R.P. Hoxey**, Computational and Wind Tunnel Modelling of Mean Wind Loads on the Silsoe Structure Building, *J. Wind Engrg. & Ind. Aerodyn. 41-44*, 1641-1652, 1992

P.J. Roache, *Computational Fluid Dynamics*, Hermosa Publishers, Albuquerque, New Mexico, 1972

P.J. Roache, Perspective: A Method for Uniform Reporting of Grid Refinement Studies, *J. Fluids Engineering, 116*, 405-413, 1995

P.J. Roache, A Method for Uniform Reporting of Grid Refinement Studies, *FED-Vol. 158, Quantification of Uncertainty in Computational Fluid Dynamics, ASME 1993, 109-120*, 1993

A.G. Robins and I.P.Castro, A Wind Tunnel Investigation of Plume Dispersion in the Vicinity of a Surface Mounted Cube - I. The Flow Field, *Atmospheric Environment 11*,291-297, 1977

A.G. Robins and I.P. Castro, A Wind Tunnel Investigation of Plume Dispersion in the Vicinity of a Surface Mounted Cube - II. The Concentration Field, *Atmospheric Environment 11*, 299-311, 1977

W. Rodi et al., One-Equation Near-Wall Turbulence Modelling With the Aid of Direction Simulation Data, *J. Fluids Engrg. 115*:196-205, 1993

W. Rodi, On the Simulation of Turbulence Modelling and Their Applications, *J. Wind Engrg. & Ind. Aerodyn. 46-47*, 183-191, 1993

W. Rodi, Experience with two-layer models combining the k-ε model with a one-equation model near the walls, *AIAA paper*, AIAA-91-0216, 1991

P.J. Saathoff, T. Stathopoulos and M. Dobrescu, Effects of Model Scale in Estimating Pollutant Dispersion near Buildings, *J. Wind Engrg. & Ind. Aerodyn. 54-55*, 549-559, 1995

Shigehiro Sakamoto et al., Numerical study on flow past 2D square cylinder by Large Eddy Simulation: Comparison between 2D and 3D computations, *J. Wind Engrg. & Ind. Aerodyn. 50*,

61-68, 1993

A. Sabaibi and V.P. Manno, New Performance Measures for Assessing Convective Algorithms: Application to 1-D Problems, *FED-Vol. 158, Quantification of Uncertainty in Computational Fluid Dynamics, ASME 1993,* 39-52, 1993

R.P. Selvam and A.H. Huber, Computer Modeling of Pollutant Dispersion around Buildings: Current Status, *Proceedings of the 9th Int. Conf. on Wind Engrg.,* New Delhi, India 596-605, 1995

R.P. Selvam and J.D. Holmes, Numerical Simulation of Thunderstorm downdrafts, *J. Wind Engrg. & Ind. Aerodyn. 41-44,* 2817-2825, 1992

R.P. Selvam and P.B. Konduru, Computational and Experimental Roof Corner Pressures on the Texas Tech Building, *J. Wind Engrg. 52,* 36-41, 1992

R.P. Selvam and D.A. Paterson, Computation of Pressures on and Velocities near the Texas Tech Buildings: Using Staggered and Non-Staggered Grids, *Ninth Structures Congress '91,* 643-646, 1991

W.H. Snyder and R.E. Lawson, Wind-tunnel Measurements of Flow Fields in the Vicinity of Buildings, *8th AMS Conf. On Appl. Air Pollution Meteorology, with AWMA,* Jan. 23-28, Nashville, TN, 1994

D.B. Spalding, Concentration fluctuations in a round turbulent free jet, *Chem. Engrg. Sci. 26,* 95-107, 1971

C.G. Speziale, Analytical Methods for the Development of Reynolds-Stress Closures in Turbulence, *Annu. Rev. Fluid Mech. 23,* 107-57, 1991

T. Stathopoulos, PDF of Wind Pressure on Low-Rise Buildings," *J.Structural Div. Proc. ASCE 106(ST5),* 973-989, 1980

T. Stathopoulos and A. Baskaran, Computation of 3-D Turbulent Wind Effects on Buildings, *Structures Congress '91,* 635-638, 1991

T. Stathopoulos and Y.S. Zhou, Numerical Simulation of Wind-induced Pressures on Buildings of Various Geometries, *J. Wind Engineering 52,* 477-488, 1992

T. Stathopoulos and Y.S. Zhou, Computation of Wind Pressures on L-shaped Buildings, *J. Engineering Mechanics, ASCE 119(8),* 1526-1541, 1993

T. Stathopoulos and Y.S. Zhou, Numerical Simulation of Wind-induced Pressures on Buildings of Various Geometries, *J. Wind Engrg. & Ind. Aerodyn. 46-47,* 419-430, 1993

T. Stathopoulos, Computational Wind Engineering: Past Achievements and Future Challenges, *J. Wind Engrg. & Ind. Aerodyn. 67&68,* 509-532, 1997

S. Takakura and Y. Suyama, Numerical Simulation of Flowfield aroun Builings in Urban Area, *J. Wind Engineering 52,* 237-242, 1992

T. Tamura, Current Research by the FDM, *J. Wind Engrg. & Ind. Aerodyn. 46-47,* 893-896, 1993

R.S. Thompson, Concentrations from above-roof Releases of Laboratory Exhausts --- a Wind Tunnel Study, *ASHRAE 97(2),* 563-572, 1997

S.D. Trent and L.L. Eyler, TEMPEST, A Three-Dimensional Time-Dependent Computer Program for Hydrothermal Analysis, Volume 1: Numerical Methods and Input Instructions, *PNL-4348 Pacific Northwest Laboratory*, Battelle, WA., 1989

M. Tsuchiya, S. Murakami, A. Mochida, K. Kondo and Y. Ishida, Development of New k-ε Model for Flow and Pressure Fields around Bluff Body, *J. Wind Engrg. & Ind. Aerodyn. 67&68,* 169-182, 1997

G.M. Turkiyyah and D.A. Reed, A Computational Modelling Environment for Wind Engineering, 1993

G.M. Turkiyyah, D. Reed and Jiyan Yang, Random Vortex Models in Wind Enginering, *Proceedings of the First Congress on Computing in Civil Engineering, 1,* 99-106, 1994

G.M. Turkiyyah, D. Reed, C. Viozat and C. Lin, Parallel performance of a three-dimensional grid-free vortex method for wind engineering simulations, *J. Wind Engrg. & Ind. Aerodyn. 67&68,* 953-954, 1997

D.C. Wilcox, Comparison of Two-Equation Turbulence Models for Boundary Layers with Pressure Gradient, *AIAA Journal 31(8),* 1414-1421, 1993

K. Wark and C.F. Warner, *Air Pollution: Its Origin and Control*, New York, HarpperCollins, 1998

D.J. Wilson and R.E. Britter, Estimates of Building Surface Concentrations From Nearby Point Sources, *Atmospheric Environment 16(11),* 2631-2646, 1982

S. Yamamura and Y. Kondo, Numerical Study on Relationship Between Builing and Groun-Level Wind Velocity, *J. Wind Engrg., 52,* 243-248, 1992

Z. Yang and T.H. Shih, New Time Scale Based k-ε Model for Near-Wall Turbulence" *AIAA Journal 31(7),* 1191-1198, 1993

Kyung-Soo Yang and J.H. Ferziger, Large-Eddy Simulation of Turbulent Obstacle Flow Using a Dynamic Subgrid-Scale Model, *AIAA Journal 31(8),* 1406-1413, 1993

C.X. Zhang, Numerical predictions of turbulence recirculationg flows with a k-e model, *J. Wind Engrg. & Ind. Aerodyn. 51,* 177-201, 1994

Y.Q. Zhang, *Numerical Simulation of Flow and Dispersion around Buildings,* Ph.D. thesis, Department of Marine, Earth and Atmospheric Science, North Carolina State University, 1993

Y.S. Zhou and T. Stathopoulos, Application of Two-Layer Methods for the Evaluation of Wind Effects on a Cubic Building, *ASHRAE Transactions 102(1),* Paper AT-96-10-2, 1996

Appendix A

Fortran source code for WIPDB_I --- Wind-Induced Pollutant Distribution around Buildings by Implicit method

A.1 The main program

```
**************************************************************************
C       This program predicts the wind-induced concentration field around buildings by
C       an implicit method (section 3.2)
C
C       Author: Ye Li
C       Date of Last Modification: 1998/1/28
**************************************************************************
C
        PROGRAM WIPDB_I
C
C       The array size of all the arrays
        INCLUDE 'CAIM.PAR'
C
C       The coefficients of the discretization equation
        INCLUDE 'COEF1.INC'
        INCLUDE 'CONCENTRATION.INC'
        INCLUDE 'CONVER.INC'
        INCLUDE 'GRID2.INC'
C
C       The index of Do Loops
        INCLUDE 'INDEX.INC'
C
C       The source term in the concentration equation
        INCLUDE 'SOURCE.INC'
C
C       The number, release rate and the initial value of pollutant source
        INCLUDE 'SOURCE1.INC'
C
C       The relaxation factor
        INCLUDE 'FACTOR.INC'
C
        REAL*8 RMAX, RES, VOL, TEMP

C********** finish of the included file ****************************
C
        CALL RFLOW
C       .
C       Initialize the concentration field
C
        CALL INI
C
        CALL INPUT
C
        CALL SETUP
C
```

```
C       Get the original source term
C
        CALL GETSOR
C
        KSTP = 0
C
 100    CALL SETUP2
C
        KSTP = KSTP + 1
        CALL SOLVE
        RES = 0.
        RMAX = 0.
        IMAX = 1
        JMAX = 1
        KMAX = 1
        DO 200 I = IST, L2
        DO 200 J = JST, M2
        DO 200 K = KST, N2
         VOL = XCV(I) * YCV(J) * ZCV(K)
         TEMP = AIM(I,J,K)*F(I-1,J,K) + AIP(I,J,K)*F(I+1,J,K) +
     +     AJM(I,J,K)*F(I,J-1,K) + AJP(I,J,K)*F(I,J+1,K) +
     +       AKM(I,J,K)*F(I,J,K-1) + AKP(I,J,K)*F(I,J,K+1) +
     +           SOURCE(I,J,K)*VOL -
     +           AP(I,J,K)*RELAX*F(I,J,K)
         IF ( ABS(TEMP). GT. RMAX) THEN
          RMAX = ABS(TEMP)
          IMAX = I
          JMAX = J
          KMAX = K
         END IF
         RES = RES + ABS(TEMP)
C       Concentration can never be negative
C       IF ( F(I,J,K) .LT. 0. ) THEN
C        WRITE(*,*) 'Negative concentration'
C        GO TO 999
C       END IF
C
C       Check the value of the coefficients
C       INCLUDE 'CAIM.DBG1'
C
 200    CONTINUE
        IF( KSTP .EQ. 1) THEN
                RNORM = RES
        END IF
        RES = RES / RNORM
C
        INCLUDE 'STEP.INC'
C
        IF ( (RES .GE. RESMAX) .AND. (KSTP .LE. MAXIT) ) GOTO 100
C
 999    CALL OUTPUT
C
        STOP
        END
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCC
      SUBROUTINE RFLOW
C*****************************************************************************
C
C     READ THE FLOWFIELD
C
C******* the included files are here ***********************************
C
C     The array size of all the arrays
      INCLUDE 'CAIM.PAR'
C
C     Building Location
      INCLUDE 'BLOC.INC'
C
C     Data from reading the output of the calculation of flow field
      INCLUDE 'FIELD.INC'
C
C     Control volume face location
      INCLUDE 'CVFL.INC'
C
C     The index of Do Loops
      INCLUDE 'INDEX.INC'
C
C********** finish of the included file ****************************

      CHARACTER*60 FNAME

      WRITE(*,*) 'ENTER MAIN FILE NAME'
      CALL OLDOPN
      WRITE(*,*) 'READING ... '

      READ(1) L1, M1, N1
C     X(I) IS THE VALUE OF X AT GRID POINTS
      READ(1) (X(I), I=1,L1)
C     Y(J) IS THE VALUE OF Y AT GRID POINTS
      READ(1) (Y(J), J=1,M1)
C     Z(K) IS THE VALUE OF Z AT GRID POINTS
      READ(1) (Z(K), K=1,N1)
      DO 100 L=1,4
        READ(1) LPOSE
        DO 100 K=1,N1
          READ(1) KPOSE
          DO 100 J=1,M1
            READ(1) JPOSE
            READ(1) (U(I,J,K,L), I=1,L1)
100   CONTINUE
C
      DO 200 L=5,7
        READ(1) LPOSE
        DO 200 K=1,N1
          READ(1) KPOSE
          DO 200 J=1,M1
```

```
          READ(1) JPOSE
          READ(1) (U(I,J,K,4), I=1,L1)
 200   CONTINUE
       CLOSE(1)

 C     READ THE GRID FILE INCLUDING THE BUILDING LOCATION
 C
       WRITE(*,*) 'ENTER THE GRID LOCATION FILE'
       READ(*, '(20A)') FNAME
       OPEN(2, FILE=FNAME, STATUS='OLD')
       READ(2,*)IIMAX, JJMAX, KKMAX
       READ(2, '(8F10.3)') (XX(I), I=1,IIMAX)
       READ(2, '(8F10.3)') (YY(J), J=1,JJMAX)
       READ(2, '(8F10.3)') (ZZ(K), K=1,KKMAX)
       READ(2,*) (LB(I), I=1,10)
       CLOSE (2, STATUS='KEEP')

 C     Set the velocity inside building equals to zero

       DO 300 I = LB(2), LB(3)
       DO 300 J = LB(6), LB(7) - 1
       DO 300 K = 2, LB(9) - 1
        U(I,J,K,1) = 0.
 300   CONTINUE
 C
       DO 400 I = LB(2), LB(3) - 1
       DO 400 J = LB(6), LB(7)
       DO 400 K = 2, LB(9) - 1
        U(I,J,K,2) = 0.
 400   CONTINUE

       DO 500 I = LB(2), LB(3) - 1
       DO 500 J = LB(6), LB(7) - 1
       DO 500 K = 2, LB(9)
        U(I,J,K,3) = 0.
 500   CONTINUE

 C     Set the turbulent eddy viscosity inside the building to zero

       DO 600 I = LB(2), LB(3) - 1
       DO 600 J = LB(6), LB(7) - 1
       DO 600 K = 2, LB(9) - 1
        U(I,J,K,4) = 0.
 600   CONTINUE

       RETURN
       END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCC
       SUBROUTINE INI
C**********************************************************************
C      This subroutine is used to initialize the concentration field
C
```

```
C
C
C
C*************************************************************************
C
C       The array size of all the arrays
        INCLUDE 'CAIM.PAR'
C
        INCLUDE 'CONCENTRATION.INC'
C
C       The index of Do Loops
        INCLUDE 'INDEX.INC'
C
C---------------------------------------------------------------
C
        DO 100 I = 1, L1
        DO 100 J = 1, M1
        DO 100 K = 1, N1
         F(I,J,K) = 0.
100     CONTINUE

        RETURN
        END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCC
        SUBROUTINE INPUT
C*************************************************************************
C
C       The convergence monitor parameters
        INCLUDE 'CONVER.INC'
C
C       The relaxation factor
        INCLUDE 'FACTOR.INC'
C
C       The reference velocity and area
        INCLUDE 'REFPAR.INC'

C       WRITE(*,*) 'ENTER THE FILE NAME FOR MIDWAY OUTPUT'
C       READ(*,'(60A)') FMDOUT
C       WRITE(*,*) 'ENTER THE FILE NAME FOR MIDWAY INPUT'
C       READ(*,'(60A)') FMDIN
C       WRITE(*,*)' KOP   = INPUT SWITCH'
C       WRITE(*,*)
        WRITE(*,*)' MAXIT = MAXIMUM NUMBER OF ITERATIONS'
    WRITE(*,*)
    WRITE(*,*)' RESMAX= MAXIMUM NORMALIZED ERROR FOR CONVERGENCE'
    WRITE(*,*)
        WRITE(*,*)' RELAX = UNDERRELAXATION FACTOR'
        WRITE(*,*)
C     WRITE(*,*)' DIVERG= MINIMUM NORMALIZED ERROR FOR DIVERGENCE'
C     WRITE(*,*)
C       WRITE(*,*)' DS    = THE MINIMUM GRID SIZE'
C       WRITE(*,*)
```

```
C    WRITE(*,*)' TYPICAL VALUES 3000, 0.000001, 20., 0.0001'
    WRITE(*,*) ' TYPE MAXIT,RESMAX,RELAX'
    READ(*,*)  MAXIT,RESMAX,RELAX
C      WRITE(*,*) 'UREF=REFERENCE VELOCITY'
C      WRITE(*,*)
C      WRITE(*,*) 'AREF=REFERENCE AREA OF THE BUILDING'
C      WRITE(*,*)
       WRITE(*,*) 'TYPE UREF, AREF'
       READ(*,*) UREF, AREF
C
       RETURN
       END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCC
       SUBROUTINE SETUP
C************************************************************************
C******* the included files are here ************************************
C
C      The array size of all the arrays
       INCLUDE 'CAIM.PAR'
C
       INCLUDE 'CVFL.INC'
C
C      Data from reading the output of the calculation of flow field
       INCLUDE 'FIELD.INC'
C
       INCLUDE 'GRID2.INC'
C      The index of Do Loops
       INCLUDE 'INDEX.INC'
C
C********** finish of the included file ***************************

       L2 = L1 - 1
       L3 = L1 - 2
       M2 = M1 - 1
       M3 = M1 - 2
       N2 = N1 - 1
       N3 = N1 - 2

C************************************************************************
C      The grid system is the so-called practice A grid system where
C      faces located midway between the grid points
C************************************************************************
C      XU(I) IS THE LOCATION OF THE C.V. FACES; i.e. THE LOCATION
C      OF U(I,J,K)
       DO 10 I = 1, L1
              XU(I) = XX(2*I-1)
10     CONTINUE
C
C      YV(J) IS THE LOCATION OF THE C.V. FACES; i.e. THE LOCATION
C      OF V(I,J,K)
C
```

```
      DO 20 J = 1, M1
            YV(J) = YY(2*J-1)
 20   CONTINUE
C
C     ZW(K) IS THE LOCATION OF THE C.V. FACES; i.e. THE LOCATION
C     OF W(I,J,K)
C
      DO 30 K = 1, N1
            ZW(K) = ZZ(2*K-1)
 30   CONTINUE
C
C     XDIF(I) IS THE DIFFERENCE X(I)-X(I-1)
C
      DO 40 I = 2, L1
       XDIF(I) = X(I) - X(I-1)
 40   CONTINUE
C
C     XCV(I) IS THE X-DIRECTION WIDTHS OF MAIN C.V.'S
C
      DO 50 I = 2, L1
       XCV(I) = XU(I+1) - XU(I)
 50   CONTINUE
C
C     XCVS(I) IS THE X-DIRECTION WIDTH OF THE STAGGERED C.V.
C     FOR U(I,J,K)
C
      DO 60 I = 2, L1
       XCVS(I) = XDIF(I)
 60   CONTINUE
C
C     XCVI(I) IS THE PART OF XCV(I) THAT OVERLAPS ON THE C.V.
C     FOR U(I,J,K)
C
      DO 70 I = 1, L1
       XCVI(I) = X(I) - XU(I)
 70   CONTINUE
C
C     XCVIP(I) IS THE PART OF XCV(I) THAT OVERLAPS ON THE C.V.
C     FOR U(I+1,J,K)
C
      DO 80 I = 1, L2
       XCVIP(I) = XU(I+1) - X(I)
 80   CONTINUE
C
C     YDIF(I) IS THE DIFFERENCE Y(J)-Y(J-1)
C
      DO 140 J = 2, M1
       YDIF(J) = Y(J) - Y(J-1)
 140  CONTINUE
C
C     YCV(J) IS THE Y-DIRECTION WIDTHS OF MAIN C.V.'S
C
      DO 150 J = 2, M1
       YCV(J) = YV(J+1) - YV(J)
```

```
 150    CONTINUE
C
C       YCVS(J) IS THE Y-DIRECTION WIDTH OF THE STAGGERED C.V.
C       FOR V(I,J,K)
C
        DO 160 J = 2, M1
         YCVS(J) = YDIF(J)
 160    CONTINUE
C
C       YCVI(J) IS THE PART OF YCV(J) THAT OVERLAPS ON THE C.V.
C       FOR V(I,J,K)
C
        DO 170 J = 1, M1
         YCVI(J) = Y(J) - YV(J)
 170    CONTINUE
C
C       YCVIP(J) IS THE PART OF YCV(J) THAT OVERLAPS ON THE C.V.
C       FOR V(I,J+1,K)
C
        DO 180 J = 1, M2
         YCVIP(J) = YV(J+1) - Y(J)
 180    CONTINUE
C
C       ZDIF(K) IS THE DIFFERENCE Z(K)-Z(K-1)
C
        DO 240 K = 2, N1
         ZDIF(K) = Z(K) - Z(K-1)
 240    CONTINUE
C
C       ZCV(K) IS THE Z-DIRECTION WIDTHS OF MAIN C.V.'S
C
        DO 250 K = 2, N1
         ZCV(K) = ZW(K+1) - ZW(K)
 250    CONTINUE
C
C       ZCVS(K) IS THE Z-DIRECTION WIDTH OF THE STAGGERED C.V.
C       FOR W(I,J,K)
C
        DO 260 K = 2, N1
         ZCVS(K) = ZDIF(K)
 260    CONTINUE
C
C       ZCVI(K) IS THE PART OF ZCV(K) THAT OVERLAPS ON THE C.V.
C       FOR W(I,J,K)
        DO 270 K = 1, N1
         ZCVI(K) = Z(K) - ZW(K)
 270    CONTINUE
C
C       ZCVIP(K) IS THE PART OF ZCV(K) THAT OVERLAPS ON THE C.V.
C       FOR W(I,J,K+1)
        DO 280 K = 1, N2
         ZCVIP(K) = ZW(K+1) - Z(K)
 280    CONTINUE
```

```
      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCC
      SUBROUTINE SETUP2
C****************************************************************************
C     COEFFICIENTS FOR THE CONCENTRATION EQUATION
C
C******* the included files are here ***********************************
C
C     The array size of all the arrays
      INCLUDE 'CAIM.PAR'
C
      INCLUDE 'BLOC.INC'
      INCLUDE 'COEF.INC'
C     The coefficients of the discretization equation
      INCLUDE 'COEF1.INC'
      INCLUDE 'CONCENTRATION.INC'
C
C     The relaxation factor
      INCLUDE 'FACTOR.INC'
C
C     Data from reading the output of the calculation of flow field
      INCLUDE 'FIELD.INC'
C     Gamma
      INCLUDE 'GAM.INC'
      INCLUDE 'GRID2.INC'
C     The index of Do Loops
      INCLUDE 'INDEX.INC'
      INCLUDE 'SOURCE.INC'
C     The number, release rate and the initial value of source
      INCLUDE 'SOURCE1.INC'
C
C********** finish of the included file ****************************
C
      IST = 2
      JST = 2
      KST = 2

C****************************************************************************
C     Because the interface of the control volume is placed midway
C     between P and E, we have Ke = 2KpKe/(Kp+Ke)
C****************************************************************************
C
      CALL GAMSOR
C
C     Set Sp to Huge inside the building
      CALL SETSOR

C     For the convenience of Do-Loop, AIM(2,J,K) is calculated first.
C     There needs no special treatment of GAMMA etc. in x-direction.
C     The same is true for AJM(I,2,K)
```

```
      DO 610 K=2,N2
      DO 610 J=2,M2

      AREA = YCV(J)*ZCV(K)
      FLOW = AREA * U(2,J,K,1)
      DIFF = AREA * 2.0 * GAM(1,J,K) * GAM(2,J,K) / ( GAM(1,J,K) +
     +  GAM(2,J,K) + 1.0E-30 ) / XDIF(2)
      CALL DIFLOW
      AIM(2,J,K) = ACOF + AMAX1(0., FLOW)

610   CONTINUE

      DO 620 K=2, N2
      DO 620 I=2, L2

      AREA = XCV(I)*ZCV(K)
      FLOW = AREA * U(I,2,K,2)
      DIFF = AREA * 2. * GAM(I,1,K) * GAM(I,2,K) / ( GAM(I,1,K) +
     +  GAM(I,2,K) + 1.E-30 ) / YDIF(2)
      CALL DIFLOW
      AJM(I,2,K) = ACOF + AMAX1(0., FLOW)

620   CONTINUE

C     Since no heat flux through the ground, AKM(I,J,2) should be zero

      DO 630 J=2, M2
      DO 630 I=2, L2

      AKM(I,J,2) = 0.

630   CONTINUE

      DO 640 K=2,N2
      DO 640 J=2,M2
      DO 640 I=2,L2
C
      AREA = YCV(J)*ZCV(K)
      FLOW = AREA * U(I+1,J,K,1)
      DIFF = AREA * 2. * GAM(I,J,K) * GAM(I+1,J,K) / ( GAM(I+1,J,K)
     +  + GAM(I,J,K) + 1.0E-30 ) / XDIF(I+1)
      CALL DIFLOW
      AIM(I+1,J,K) = ACOF + AMAX1(0.,FLOW)
      AIP(I,J,K) = AIM(I+1,J,K) - FLOW
C
      AREA = XCV(I) * ZCV(K)
      FLOW = AREA * U(I,J+1,K,2)
      DIFF = AREA * 2. * GAM(I,J,K) * GAM(I,J+1,K) /
     +  ( GAM(I,J+1,K) + GAM(I,J,K) + 1.0E-30 ) / YDIF(J+1)
      CALL DIFLOW
      AJM(I,J+1,K) = ACOF + AMAX1(0., FLOW)
      AJP(I,J,K) = AJM(I,J+1,K) - FLOW
C
      AREA = XCV(I) * YCV(J)
```

```
        FLOW = AREA * U(I,J,K+1,3)
        DIFF = AREA * 2. * GAM(I,J,K) * GAM(I,J,K+1)/
     +   ( GAM(I,J,K+1) + GAM(I,J,K) + 1.0E-30 ) / ZDIF(K+1)
        CALL DIFLOW
        AKM(I,J,K+1) = ACOF + AMAX1(0., FLOW)
        AKP(I,J,K) = AKM(I,J,K+1) - FLOW
C
        VOL = XCV(I) * YCV(J) * ZCV(K)
        AP(I,J,K) = (-AP(I,J,K)*VOL+AIP(I,J,K)+AIM(I,J,K)+AJP(I,J,K)+
     +   AJM(I,J,K)+AKP(I,J,K)+AKM(I,J,K) ) / RELAX
        CONB(I,J,K) = SOURCE(I,J,K)*VOL + (1.-RELAX)*AP(I,J,K)*F(I,J,K)
 640    CONTINUE
C
C       No flux through front and rear walls

        DO 650 J = LB(6), LB(7) - 1
        DO 650 K = 2, LB(9) - 1
         AIP( LB(2)-1, J, K ) = 0.
         AIM( LB(3), J, K ) = 0.
 650    CONTINUE
C
C       No flux through roof
C
        DO 660 I = LB(2), LB(3) - 1
        DO 660 J = LB(6), LB(7) - 1
         AKM(I, J, LB(9) ) = 0.
 660    CONTINUE
C
C       No flux through side walls
C
        DO 670 I = LB(2), LB(3) - 1
        DO 670 K = 2, LB(9) - 1
         AIP( I, LB(6)-1, K ) = 0.
         AJM( I, LB(7), K ) = 0.
 670    CONTINUE
C
        RETURN
        END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCC
        SUBROUTINE GAMSOR
C***********************************************************************
c******* the included files are here ***********************************
C
C       The array size of all the arrays
        INCLUDE 'CAIM.PAR'
C
C       Data from reading the output of the calculation of flow field
        INCLUDE 'FIELD.INC'
C
C       Gamma
        INCLUDE 'GAM.INC'
C
C       The index of Do Loops
```

```
      INCLUDE 'INDEX.INC'
C
c********** finish of the included file ***************************
C
C     THE EDDY VISCOSITY
C
      ZSC = 0.9
C
      DO 1000 I=1,L1
      DO 1000 J=1,M1
      DO 1000 K=1,N1
       GAM(I,J,K) = U(I,J,K,4)/ZSC
 1000 CONTINUE
C
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCC
      SUBROUTINE GETSOR
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCC
C     This subroutine is used to get the position and release rate
C     of the concentration source, and then calculated the
C     corresponding source term in the concentration equation
C
C     Author: Ye Li
C     Date: 1998/01/19
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCC
C
C     The array size of all the arrays
      INCLUDE 'CAIM.PAR'
C
C     Data from reading the output of the calculation of flow field
      INCLUDE 'FIELD.INC'
C
C     The extended grid system
      INCLUDE 'GRID2.INC'
C
C     The index of Do Loops
      INCLUDE 'INDEX.INC'
C
C     The number, release rate and the initial value of source
      INCLUDE 'SOURCE1.INC'
C
C-----------------------------------------------------------
C
C     Initialize the source term
      DO 100 I = 1, L1
      DO 100 J = 1, M1
      DO 100 K = 1, N1
       SOURCE(I,J,K) = 0.
 100  CONTINUE
C
```

```
         WRITE(*,*) 'TYPE THE NUMBER OF SOURCES'
         READ(*,*) NUM_OF_SOURCE
         WRITE(*,*) 'TYPE THE SOURCE_RATE'
         READ(*,*) SOURCE_RATE
C
         DO 200 L = 1, NUM_OF_SOURCE
C
          WRITE(*,*) 'TYPE THE SOURCE POSITION POSX, POSY,POSZ'
          READ(*,*) POSX, POSY, POSZ
C
         LOCX = 1
         LOCY = 1
         LOCZ = 2
C
         DO 110 I = 1, L1
          TEMP = ABS( X(I) - POSX )
          IF( TEMP .LE. ABS( X( LOCX ) - POSX ) ) THEN
           LOCX = I
          END IF
 110     CONTINUE
C
         DO 120 J = 1, M1
          TEMP = ABS( Y(J) - POSY )
          IF( TEMP .LE. ABS( Y( LOCY ) - POSY ) ) THEN
           LOCY = J
          END IF
 120     CONTINUE
C
         DO 130 K = 2, N1
          TEMP = ABS( Z(K) - POSZ )
          IF( TEMP .LE. ABS( Z( LOCZ ) - POSZ ) ) THEN
           LOCZ = K
          END IF
 130     CONTINUE
C
         SOURCE(LOCX,LOCY,LOCZ) = SOURCE_RATE / XCV(LOCX) /
   /      YCV(LOCY) / ZCV(LOCZ)
C
        WRITE(*,*) 'POSX=',X(LOCX),'POSY=',Y(LOCY),'POSZ=',Z(LOCZ)
        WRITE(*,*) 'LOCX=',LOCX, 'LOCY=',LOCY,'LOCZ=',LOCZ
C
 200    CONTINUE

        RETURN
        END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCC
        SUBROUTINE SETSOR
C***********************************************************************
C       The array size of all the arrays
        INCLUDE 'CAIM.PAR'
C
        INCLUDE 'BLOC.INC'
C
```

```
C       The coefficients of the discretization equation
        INCLUDE 'COEF1.INC'
C
C       The index of Do Loops
        INCLUDE 'INDEX.INC'
C
C***************************************************************************
C       The source term is expressed by Sc+Sp*Phi(p)
C     SOURCE ---> Sc is defined in subroutine GETSOR
C       AP   ---> Sp
C
        DO 100 I = 1, L1
        DO 100 J = 1, M1
        DO 100 K = 1, N1
C
         AP(I,J,K) = 0.
C
 100    CONTINUE
C
C       In order to set the concentration inside the building to zero,
C       the technique of huge Sc and Sp is used so that Sc+Sp*Phi(p)=0.
C       Phi(p) = - Sc/Sp
C       Here we set Sp=-1.e+30, and Sc=-Sp*Phi(desired)=0.0
C
        DO 110 I = LB(2), LB(3) - 1
        DO 110 J = LB(6), LB(7) - 1
        DO 110 K = 2, LB(9) - 1
         AP(I,J,K) = -1.E+30
 110    CONTINUE
C
        RETURN
        END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCC
        SUBROUTINE DIFLOW
C***************************************************************************
        INCLUDE 'COEF.INC'
C***************************************************************************
        ACOF=DIFF
        IF (FLOW .EQ. 0.) RETURN
        TEMP = DIFF - ABS(FLOW)*0.1
        ACOF = 0.
        IF (TEMP .LE. 0.) RETURN
        TEMP = TEMP / DIFF
        ACOF = DIFF * TEMP ** 5
        RETURN
        END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCC
        SUBROUTINE SOLVE
C***************************************************************************
C
C       The array size of all the arrays
```

```
      INCLUDE 'CAIM.PAR'
C
      INCLUDE 'CONCENTRATION.INC'
C     The coefficients of the discretization equation
      INCLUDE 'COEF1.INC'
C     The index of Do Loops
      INCLUDE 'INDEX.INC'
      INCLUDE 'SOURCE.INC'
C********** finish of the included file ******************************
C     In order to avoid underflow, real*8 for the value needed
C     1998/01/29
      REAL*8 TEMP, DENOM
      REAL*8  PT(100), QT(100)
C
      ISTF = IST - 1
      JSTF = JST - 1
      KSTF = KST - 1
      IT1 = L2 + IST
      IT2 = L3 + IST
      JT1 = M2 + JST
      JT2 = M3 + JST
      KT1 = N2 + KST
      KT2 = N3 + KST
C*************************************************************************
C     Because of symmetry, the sweep along the y+ and y- direction has
C     the same effect, so that we only sweep along y+ direction.
C     Because a sweep from upstream to downstream would produce much
C     faster convergence than a sweep against the stream, we only
C     sweep along x+ direction.
C*************************************************************************
C
C     For x-direction grid line
C     Sweep along y+ and z+ direction
C
      DO 90 K = KST, N2
      DO 90 J = JST, M2
      PT(ISTF) = 0.
      QT(ISTF) = F(ISTF,J,K)
      DO 70 I = IST, L2
      DENOM = AP(I,J,K) - PT(I-1)*AIM(I,J,K)
      PT(I) = AIP(I,J,K) / DENOM
      TEMP = CONB(I,J,K) + AJP(I,J,K)*F(I,J+1,K) +
     +     AJM(I,J,K)*F(I,J-1,K) + AKP(I,J,K)*F(I,J,K+1) +
     +     AKM(I,J,K)*F(I,J,K-1)
      QT(I) = ( TEMP + AIM(I,J,K)*QT(I-1) ) / DENOM
70    CONTINUE
      DO 80 II = IST, L2
      I = IT1 - II
80    F(I,J,K) = F(I+1,J,K)*PT(I) + QT(I)
90    CONTINUE
C-------------------------------------------------------------
C     Sweep along y+, z- direction
C
      DO 190 KK = KST, N3
```

```
      DO 190 J = JST, M2
      K = KT2 - KK
      PT(ISTF) = 0.
      QT(ISTF) = F(ISTF,J,K)
      DO 170 I = IST, L2
      DENOM = AP(I,J,K) - PT(I-1)*AIM(I,J,K)
      PT(I) = AIP(I,J,K) / DENOM
      TEMP = CONB(I,J,K) + AJP(I,J,K)*F(I,J+1,K) +
     +  AJM(I,J,K)*F(I,J-1,K) + AKP(I,J,K)*F(I,J,K+1) +
     +  AKM(I,J,K)*F(I,J,K-1)
      QT(I) = ( TEMP + AIM(I,J,K)*QT(I-1) ) / DENOM
  170 CONTINUE
      DO 180 II = IST, L2
      I = IT1 - II
  180 F(I,J,K) = F(I+1,J,K)*PT(I) + QT(I)
  190 CONTINUE
C-----------------------------------------------------------------
C-----------------------------------------------------------------
C
C     For y-direction grid line
C     Sweep along x+ and z+ direction
C
      DO 290  K = KST, N2
      DO 290  I = IST, L2
      PT(JSTF) = 0.
      QT(JSTF) = F(I,JSTF,K)
      DO 270 J = JST,M2
      DENOM = AP(I,J,K) - PT(J-1)*AJM(I,J,K)
      PT(J) = AJP(I,J,K) / DENOM
      TEMP = CONB(I,J,K) + AIP(I,J,K)*F(I+1,J,K) +
     +  AIM(I,J,K)*F(I-1,J,K) + AKP(I,J,K)*F(I,J,K+1) +
     +  AKM(I,J,K)*F(I,J,K-1)
      QT(J) = ( TEMP + AJM(I,J,K)*QT(J-1) ) / DENOM
  270 CONTINUE
      DO 280 JJ =JST, M2
      J = JT1-JJ
  280 F(I,J,K) = F(I,J+1,K)*PT(J) + QT(J)
  290 CONTINUE
C-----------------------------------------------------------------
C
C     Sweep along x+ and z- direction
C
      DO 390 KK = KST, N3
      DO 390 I = IST, L2
      K = KT2 - KK
      PT(JSTF) = 0.
      QT(JSTF) = F(I,JSTF,K)
      DO 370 J = JST, M2
      DENOM = AP(I,J,K) - PT(J-1)*AJM(I,J,K)
      PT(J) = AJP(I,J,K) / DENOM
      TEMP = CONB(I,J,K) + AIP(I,J,K)*F(I+1,J,K) +
     +  AIM(I,J,K)*F(I-1,J,K) + AKP(I,J,K)*F(I,J,K+1) +
     +  AKM(I,J,K)*F(I,J,K-1)
      QT(J) = ( TEMP + AJM(I,J,K)*QT(J-1) ) / DENOM
```

```
 370    CONTINUE
        DO 380 JJ = JST, M2
        J = JT1 - JJ
 380    F(I,J,K) = F(I,J+1,K)*PT(J) + QT(J)
 390    CONTINUE
C-----------------------------------------------------------------
C-----------------------------------------------------------------
C
C       For z-direction grid line
C       Sweep along x+ and y+ direction
C
        DO 490  J = JST, M2
        DO 490 I = IST, L2
        PT(KSTF) = 0.
        QT(KSTF) = F(I,KSTF,K)
        DO 470 K = KST,N2
        DENOM = AP(I,J,K) - PT(K-1)*AKM(I,J,K)
        PT(K) = AKP(I,J,K) / DENOM
        TEMP = CONB(I,J,K) + AIP(I,J,K)*F(I+1,J,K) +
     +  AIM(I,J,K)*F(I-1,J,K) + AJP(I,J,K)*F(I,J+1,K) +
     +  AJM(I,J,K)*F(I,J-1,K)
        QT(K) = ( TEMP + AKM(I,J,K)*QT(K-1) ) / DENOM
 470    CONTINUE
        DO 480 KK =KST, N2
        K = KT1-KK
 480    F(I,J,K) = F(I,J,K+1)*PT(K) + QT(K)
 490    CONTINUE

        RETURN
        END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCC
        SUBROUTINE OLDOPN
C***********************************************************************
        CHARACTER*60 MANFIL, ANS
        LOGICAL STAT
3100    READ(*, '(A60)') MANFIL
        INQUIRE(FILE=MANFIL, EXIST=STAT)
        IF( .NOT. STAT) THEN
         WRITE(*,*) 'FILE IS NON EXISTANT, REENTER FILENAME.'
         GO TO 3100
        ENDIF

        OPEN(UNIT=1, FILE=MANFIL, STATUS='OLD', FORM='UNFORMATTED')
        REWIND(1)
        RETURN
        END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCC
        SUBROUTINE OUTPUT
C***********************************************************************
C
```

```
C       The array size
C       The array size of all the arrays
        INCLUDE 'CAIM.PAR'
C
        INCLUDE 'CONCENTRATION.INC'
C
C       The reference velocity and area
        INCLUDE 'REFPAR.INC'
C       Data from reading the output of the calculation of flow field
        INCLUDE 'FIELD.INC'
C       The index of Do Loops
        INCLUDE 'INDEX.INC'
C
C       The number, release rate and the initial value of source
        INCLUDE 'SOURCE1.INC'
C
C*******************************************************************************
C
        CHARACTER*60 FNAME
C
        WRITE(*,*) 'ENTER THE OUTPUT FILE NAME'
        READ(*, '(A60)') FNAME
        OPEN(7, FILE=FNAME, STATUS='NEW', FORM='UNFORMATTED')
        WRITE(7) L1, M1, N1
        WRITE(7) (X(I), I=1,L1)
        WRITE(7) (Y(J), J=1,M1)
        WRITE(7) (Z(K), K=1,N1)
C
        DO 2900 K=1,N1
        DO 2900 J=1,M1
        DO 2900 I=1,L1
         F(I,J,K) = F(I,J,K)*UREF*AREF/(SOURCE_RATE*NUM_OF_SOURCE)
2900    CONTINUE

         DO 3000 K=1,N1
          WRITE(7) K
         DO 3000 J=1,M1
            WRITE(7) J
         DO 3000 I=1,L1
            WRITE(7) F(I,J,K)
3000    CONTINUE
        CLOSE(7, STATUS='KEEP')
C
        RETURN
        END
```

A.2 The included files

caim.par

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C       Define the maximum array size in three directions respectively
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      parameter (ix=65, iy=60, iz=40)
```

bloc.inc

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C       The building location
C       Authour: Ye Li
C       Date: 1998/01/16
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      COMMON /BLOC/ LB(10)
```

coef.inc

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCC
C
C       The strength of convection Fe, diffusion conductance De
C       and function A|P|
C
C       Author: Ye Li
C       Date: 1998/01/19
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCC
      COMMON/COEF/FLOW,DIFF,ACOF
```

coef1.inc

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCC
C
C       The coefficients of the discretization equation
C
C       Author: Ye Li
C       Date: 1998/01/19
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C       In order to avoid underflow, all the coefficients are
```

```
C       assigned real*8, 1998/01/29
C
        REAL*8 AIP, AIM, AJP, AJM, AKP, AKM, AP
C
        COMMON /COEF1/ AIP(IX,IY,IZ), AIM(IX,IY,IZ), AJP(IX,IY,IZ),
     +  AJM(IX,IY,IZ), AKP(IX,IY,IZ), AKM(IX,IY,IZ), AP(IX,IY,IZ)
```

concentration.inc

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCC
C
C       The concentration
C
C       Author: Ye Li
C       Date: 1998/01/19
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCC
C
C       In order to avoid underflow, use real*8 for F. 1998/01/29
C
        REAL*8 F
        COMMON /CONCENTRATION/ F(IX,IY,IZ)
```

conver.inc

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C       The control parameter for iteration
C
C       Author: Ye Li
C       Date: 1998/01/18
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        COMMON /CONVER/ RESMAX, MAXIT
```

cvfl.inc

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C       The coordinates of grid and half grid points
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        COMMON/GRID/ XX(2*IX), YY(2*IY), ZZ(2*IZ)
```

factor.inc

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C       The underrelaxation factor
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C       In order to avoid underflow, real*8 for relax
        REAL*8 RELAX
        COMMON /FACTOR/ RELAX
```

field.inc

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C       Data from reading the output of the calculation of flow field
C
C       Author: Ye Li
C       Date:   1998/01/16
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

        COMMON /FIELD/ U(IX,IY,IZ,4), X(IX), Y(IY), Z(IZ)
```

gamma.inc

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C       see definition (3-18)
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

        COMMON /GAM/ GAM(IX,IY,IZ)
```

grid2.inc

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C       All the parameters related to a control volume
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
        COMMON /GRID2/ XU(IX), XDIF(IX), XCV(IX), XCVS(IX),
     +                 XCVI(IX), XCVIP(IX), YV(IY), YDIF(IY),
     +                 YCV(IY), YCVS(IY), YCVI(IY), YCVIP(IY),
     +                 ZW(IZ), ZDIF(IZ), ZCV(IZ), ZCVS(IZ),
     +                 ZCVI(IZ), ZCVIP(IZ)
```

index.inc

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C       All index in x, y, z directions
C
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      COMMON /INDEX/L1,L2,L3,M1,M2,M3,N1,N2,N3,IST,JST,KST
```

refpar.inc

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C     The control parameters of the problem
C
C     Author: Ye Li
C     Date: 1998/01/18
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      COMMON /REFPAR/ UREF, AREF
```

source.inc

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C     The source term in the concentration equation
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

C     In order to avoid underflow, real*8 for conb
C
      REAL*8 CONB

      COMMON /SOURCE/ CONB(IX,IY,IZ)
```

source1.inc

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C     The location and parameters of pollutant sources
C
C     Author: Ye Li
C     Date: 1998/01/19
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C     In order to avoid underflow, real*8 for source
      REAL*8 SOURCE
C

      COMMON/SOURCE1/NUM_OF_SOURCE, SOURCE_RATE, SOURCE(IX,IY,IZ)
```

Appendix B

Fortran source code for WIPDB_E --- Wind-Induced Pollutant Distribution around Buildings by Explicit method

```
**************************************************************************
C       This program predicts the wind-induced concentration field around buildings by
C       an explicit method.
C
C       Author: Ye Li
C       Date of Last Modification: 1997/10/17
**************************************************************************
C
        PROGRAM WIPDB_E
C
        include 'con_a.con'
C
        REAL*8 CR1
        character*60 FNAME, FMDIN, FMDOUT
        character*1 ANS
C
C       READ THE FLOWFIELD
C
        WRITE(*,*) 'ENTER MAIN FILE NAME'
        CALL OLDOPN
        WRITE(*,*) 'READING ... '

        READ(1) IMAX, JMAX, KMAX
        READ(1) (X(I), I=1,IMAX)
        READ(1) (Y(J), J=1,JMAX)
        READ(1) (Z(K), K=1,KMAX)
        DO 100 L=1,4
          READ(1) LPOSE
          DO 100 K=1,KMAX
            READ(1) KPOSE
            DO 100 J=1,JMAX
              READ(1) JPOSE
              READ(1) (U(I,J,K,L), I=1,IMAX)
100     CONTINUE
C
        DO 200 L=5,7
          READ(1) LPOSE
          DO 200 K=1,KMAX
            READ(1) KPOSE
            DO 200 J=1,JMAX
              READ(1) JPOSE
              READ(1) (U(I,J,K,4), I=1,IMAX)
200     CONTINUE
        CLOSE(1)

C       READ THE GRID FILE INCLUDING THE BUILDING LOCATION
C
```

```
      WRITE(*,*) 'ENTER THE GRID LOCATION FILE'
      READ(*, '(20A)') FNAME
      OPEN(2, FILE=FNAME, STATUS='OLD')
      READ(2,*)IIMAX, JJMAX, KKMAX
      READ(2, '(8F10.3)') (XX(I), I=1,IIMAX)
      READ(2, '(8F10.3)') (YY(J), J=1,JJMAX)
      READ(2, '(8F10.3)') (ZZ(K), K=1,KKMAX)
      READ(2,*) (LB(I), I=1,10)
      CLOSE (2, STATUS='KEEP')

      open(3, file='conoutf.dat', status='new', form='formatted')
      WRITE(3,*) IMAX, JMAX, KMAX
      WRITE(3,*) (X(I), I=1,IMAX)
      WRITE(3,*) (Y(J), J=1,JMAX)
      WRITE(3,*) (Z(K), K=1,KMAX)
      WRITE(3,*) (LB(L), L=1,10)
      CLOSE (3, STATUS='SAVE')
C
      WRITE(*,*) 'ENTER THE FILE NAME FOR MIDWAY OUTPUT'
      READ(*,'(60A)') FMDOUT
      WRITE(*,*) 'ENTER THE FILE NAME FOR MIDWAY INPUT'
      READ(*,'(60A)') FMDIN
      WRITE(*,*)' KOP   = INPUT SWITCH'
      WRITE(*,*)
    WRITE(*,*)' MAXIT = MAXIMUM NUMBER OF ITERATIONS'
    WRITE(*,*)
    WRITE(*,*)' RESMAX= MAXIMUM NORMALIZED ERROR FOR CONVERGENCE'
    WRITE(*,*)
    WRITE(*,*)' DIVERG= MINIMUM NORMALIZED ERROR FOR DIVERGENCE'
    WRITE(*,*)
      WRITE(*,*)' DS    = THE MINIMUM GRID SIZE'
      WRITE(*,*)
    WRITE(*,*)' TYPICAL VALUES 3000, 0.000001, 20., 0.0001'
    WRITE(*,*) ' TYPE MAXIT,RESMAX,DIVERG, DS'
    READ(*,*)  KOP, MAXIT,RESMAX,DIVERG,DS
      WRITE(*,*) 'UREF=REFERENCE VELOCITY'
      WRITE(*,*)
      WRITE(*,*) 'AREF=REFERENCE AREA OF THE BUILDING'
      WRITE(*,*)
      WRITE(*,*) 'TYPE UREF, AREF'
      READ(*,*) UREF, AREF

C     CALCULATE THE GRID PARAMETER
      DO 300 I = 2 , IMAX-1
C
C---- HERE XD(I) IS DELTAX AT i+1/2, DX(I) IS DELTAX AT i
C---- PAY ATTENTION THAT THE SCALAR IS NOT AT THE CENTER OF THE
C---- CELL, THE LEFT PART IS EQUAL TO THE RIGHT PART OF THE LEFT
C---- CELL.
C
       XD(I) = X(I+1) - X(I)
       DX(I) = (X(I+1) - X(I-1)) / 2.
300   CONTINUE
```

```
        XD(1) = X(2)-X(1)

        DO 400 J = 2 , JMAX-1
         YD(J) = Y(J+1) - Y(J)
         DY(J) = (Y(J+1) - Y(J-1)) / 2.
400     CONTINUE

        YD(1) = Y(2)-Y(1)

        DO 500 K = 2 , KMAX-1
         ZD(K) = Z(K+1) - Z(K)
         DZ(K) = (Z(K+1) - Z(K-1)) / 2.
500     CONTINUE

        ZD(1) = Z(2)-Z(1)

C       THE SOURCE OF CONCENTRATION

        DO 600 I=1,IMAX
        DO 600 J=1,JMAX
        DO 600 K=1,KMAX
         S(I,J,K)=0
600     CONTINUE

C       READ IN THE SOURCE

        WRITE(*,*) 'TYPE NUMBER OF SOURCES'
        READ(*,*) NUM_OF_SOURCE

        DO 700 L=1, NUM_OF_SOURCE

        WRITE(*,*) 'TYPE THE SOURCE POSITION NX,NY,NZ'
        READ(*,*) NX,NY,NZ
        S(NX,NY,NZ) = SOURCE_RATE/DX(NX)/DY(NY)/DZ(NZ)

700     CONTINUE

C       THE INITIAL CONCENTRATION

        DO 1100 I=1,IMAX
        DO 1100 J=1,JMAX
        DO 1100 K=1,KMAX

        CONC(I,J,K) = S(I,J,K)
        C(I,J,K) = 0.

1100    CONTINUE

C       THE EDDY VISCOSITY

        ZSC = 0.7

        DO 1300 I=1,IMAX-1
        DO 1300 J=1,JMAX
```

```
      DO 1300 K=1,KMAX
       FE = ( X(I+1) - XX(2*I+1) ) / XD(I)
       IF ( ( U(I,J,K,4) .EQ. 0.0) .OR. ( U(I+1,J,K,4) .EQ. 0.0) ) THEN
            DFX(I,J,K) = 0.0
       ELSE
            TEMP = (1-FE)/U(I,J,K,4) + FE/U(I+1,J,K,4)
            DFX(I,J,K) = 1/TEMP/ZSC
       END IF
1300   CONTINUE

      DO 1400 I=1,IMAX
      DO 1400 J=1,JMAX-1
      DO 1400 K=1,KMAX
       FE = ( Y(J+1) - YY(2*J+1) ) / YD(J)
       IF ( ( U(I,J,K,4) .EQ. 0.0) .OR. ( U(I,J+1,K,4) .EQ. 0.0) ) THEN
            DFY(I,J,K) = 0.0
       ELSE
            TEMP = (1-FE)/U(I,J,K,4) + FE/U(I,J+1,K,4)
            DFY(I,J,K) = 1/TEMP/ZSC
       END IF
1400   CONTINUE

      DO 1500 I=1,IMAX
      DO 1500 J=1,JMAX
      DO 1500 K=1,KMAX-1
       FE = ( Z(K+1) - ZZ(2*K+1) ) / ZD(K)
       IF ( ( U(I,J,K,4) .EQ. 0.0) .OR. ( U(I,J,K+1,4) .EQ. 0.0) ) THEN
            DFZ(I,J,K) = 0.0
       ELSE
            TEMP = (1-FE)/U(I,J,K,4) + FE/U(I,J,K+1,4)
            DFZ(I,J,K) = 1/TEMP/ZSC
       END IF
1500   CONTINUE

      WRITE(*,*) 'ENTER THE ERROR FILE NAME'
      READ(*, '(60A)') FNAME
      OPEN(4, FILE = FNAME, ACCESS = 'APPEND', STATUS = 'UNKNOWN',
     1       FORM = 'FORMATTED')

      KSTP = 0

      WRITE(*,*) 'CALCULATE DT?'
            READ(*,'(1A)') ANS
      IF ( ANS .EQ. 'Y') THEN
            DT = 2*DS/UREF
      ELSE IF (KOP .LE. 1) THEN
            WRITE(*,*) 'TYPE DT'
            READ(*,*) DT
            GO TO 1540
      ELSE
            OPEN(6, FILE=FMDIN, STATUS='OLD', FORM='UNFORMATTED')
            READ(6) CONC, C, DT, KSTP
            CLOSE (6)
      END IF
```

```
1540    DT = 2*DT
1550    DT = DT/2
        WRITE(*,*) 'DT=', DT
        KSTP = 0

C       SET BOUNDARY CONDITION

1600    CALL BOUNDSUB

C       THE FLUX IN X DIRECTION
C
        KSTP = KSTP + 1
        IF  (KSTP .LE. 10) THEN
                WRITE(*,199) KSTP, CR1
        ELSE IF ( (KSTP .LE. 100) .AND. ( MOD(KSTP, 10) .EQ. 0) )THEN
                WRITE(*,199) KSTP, CR1
        ELSE IF ( (KSTP .LE. 1000) .AND. ( MOD(KSTP, 100) .EQ. 0) )THEN
                WRITE(*,199) KSTP, CR1
        ELSE IF ( (KSTP .LE. 10000) .AND. ( MOD(KSTP, 1000) .EQ. 0) )THEN
                WRITE(*,199) KSTP, CR1
        ELSE IF ( (KSTP .LE. 100000) .AND. ( MOD(KSTP, 10000) .EQ. 0) )THEN
                WRITE(*,199) KSTP, CR1
        ELSE IF ( MOD(KSTP, 10000) .EQ. 0 ) THEN
                WRITE(*,199) KSTP, CR1
        END IF
C
199     FORMAT(1X,'KSTP=', I6, 5X, 'CR1=', E13.4)

C       CLEARANCE OF C(I,J,K)
        DO 1800 I=1,IMAX
        DO 1800 J=1,JMAX
        DO 1800 K=1,KMAX
         C(I,J,K) = 0.0
1800    CONTINUE

        DO 1900 I=1,IMAX-1
        DO 1900 J=1,JMAX-1
        DO 1900 K=1,KMAX
      AF(I,J,K) = AMAX1(DFX(I,J,K)/XD(I) - U(I+1,J,K,1)/2,
     1            -U(I+1,J,K,1))
         AF(I,J,K) = DMAX1(0.0D0, AF(I,J,K))
         AF(I,J,K) = (CONC(I,J,K) - CONC(I+1,J,K))*AF(I,J,K)

         AF(I,J,K) = U(I+1,J,K,1) * CONC(I,J,K) + AF(I,J,K)
1900    CONTINUE

        DO 2000 I=2,IMAX-1
        DO 2000 J=1,JMAX-1
        DO 2000 K=1,KMAX
         C(I,J,K) = C(I,J,K) - (AF(I,J,K)-AF(I-1,J,K))/DX(I)
2000    CONTINUE

C       THE FLUX IN Y DIRECTION
```

```
      DO 2100 I=1,IMAX
      DO 2100 J=1,JMAX-1
      DO 2100 K=1,KMAX
     AF(I,J,K) = AMAX1(DFY(I,J,K)/YD(J)-U(I,J+1,K,2)/2,
     1          -U(I,J+1,K,2))
       AF(I,J,K) = DMAX1(0.0D0, AF(I,J,K))
       AF(I,J,K) = (CONC(I,J,K)-CONC(I,J+1,K)) * AF(I,J,K)
     AF(I,J,K) = U(I,J+1,K,2)*CONC(I,J,K) + AF(I,J,K)
2100  CONTINUE

      DO 2200 I=1,IMAX
      DO 2200 J=2,JMAX-1
      DO 2200 K=1,KMAX
       C(I,J,K) = C(I,J,K) - (AF(I,J,K)-AF(I,J-1,K))/DY(J)
2200  CONTINUE

C     THE FLUX IN Z DIRECTION

      DO 2300 I=1,IMAX
      DO 2300 J=1,JMAX-1
      DO 2300 K=1,KMAX-1
     AF(I,J,K) = AMAX1(DFZ(I,J,K)/ZD(K)-U(I,J,K+1,3)/2,
     1          -U(I,J,K+1,3))
     AF(I,J,K) = DMAX1(0.0D0,AF(I,J,K))
     AF(I,J,K) = (CONC(I,J,K)-CONC(I,J,K+1))*AF(I,J,K)
     AF(I,J,K) = U(I,J,K+1,3)*CONC(I,J,K) + AF(I,J,K)
2300  CONTINUE

      DO 2400 I=1,IMAX
      DO 2400 J=1,JMAX
      DO 2400 K=2,KMAX-1
       C(I,J,K) = C(I,J,K) - (AF(I,J,K)-AF(I,J,K-1))/DZ(K)
2400  CONTINUE

      DO 2500 I=1,IMAX
      DO 2500 J=1,JMAX
      DO 2500 K=1,KMAX
       C(I,J,K) = C(I,J,K) + S(I,J,K)
2500  CONTINUE

      DO 2600 I=1,IMAX
      DO 2600 J=1,JMAX
      DO 2600 K=1,KMAX
       CONC(I,J,K) = CONC(I,J,K) + DT*C(I,J,K)
      IF ( CONC(I,J,K) .LT. 0. ) then
        WRITE(*,*) 'NEGATIVE CONCENTRATION CONC(I,J,K)-'
        WRITE(*,*) I,J,K,CONC(I,J,K)
        GOTO 1550
      END IF
2600  CONTINUE

      CR1 = 1.D-30
c-    CR2 = 0.0
```

```
c-      CR3 = 0.0
C
        DO 2800 I = 1, IMAX
        DO 2800 J = 1, JMAX
        DO 2800 K = 1, KMAX
         CR1 = DMAX1 (CR1, DABS(C(I,J,K)) )
c-       CR2 = CR2 + ABS(C(I,J,K))
c-       CR3 = CR3 + C(I,J,K)*C(I,J,K)
2800    CONTINUE

        IF ( (KSTP .LE. 10) .OR. ( ( KSTP .LE. 100) .AND.
   1    ( MOD (KSTP, 10) .EQ. 0) ) .OR. (MOD(KSTP, 100) .EQ. 0)) THEN
c-              WRITE(4,*) KSTP, CR1, CR2, CR3
                WRITE(4,*) KSTP, CR1
        END IF

        IF ((CR1 .LE. resmax) .OR. (KSTP .GT. MAXIT))  THEN
         GO TO 2850
        ELSE IF ( MOD(KSTP,1000) .EQ. 0) THEN
         OPEN(5, FILE=FMDOUT, STATUS='UNKNOWN', FORM='UNFORMATTED')
         WRITE(5) CONC,C,DT,KSTP
         CLOSE(5)
         GO TO 1600
        ELSE
         GO TO 1600
        END IF

2850    CONTINUE

c       OPEN(33, FILE='CONOUTF.DAT', STATUS='NEW', FORM='FORMATTED')
        WRITE(*,*) 'ENTER THE OUTPUT FILE NAME'
        READ(*, '(60A)') FNAME
        OPEN(7, FILE=FNAME, STATUS='NEW', FORM='UNFORMATTED')
        WRITE(7) IMAX, JMAX, KMAX
        WRITE(7) (X(I), I=1,IMAX)
        WRITE(7) (Y(J), J=1,JMAX)
        WRITE(7) (Z(K), K=1,KMAX)
c       WRITE(33,*) IMAX, JMAX, KMAX
c       WRITE(33,*) (X(I), I=1,IMAX)
c       WRITE(33,*) (Y(J), J=1,JMAX)
c       WRITE(33,*) (Z(K), K=1,KMAX)
C

        DO 2900 K=1,KMAX
        DO 2900 J=1,JMAX
        DO 2900 I=1,IMAX
         CONC(I,J,K) = CONC(I,J,K)*UREF*AREF/(SOURCE_RATE*NUM_OF_SOURCE)
2900    CONTINUE

C*******************************************************************************
C
C       Output format is changed in order to be compatible with
C       rec.for
C*******************************************************************************
```

```
      DO 3000 K=1,KMAX
       WRITE(7) KPOSE
      DO 3000 J=1,JMAX
       WRITE(7) JPOSE
      DO 3000 I=1,IMAX
       WRITE(7) CONC(I,J,K)
3000  CONTINUE
      CLOSE(7, STATUS='KEEP')
C     close(33, status='keep')
      close(4)


      STOP
      END

      SUBROUTINE OLDOPN
      CHARACTER*60 MANFIL, ANS
      LOGICAL STAT
3100  READ(*, '(A60)') MANFIL
      INQUIRE(FILE=MANFIL, EXIST=STAT)
      IF( .NOT. STAT) THEN
       WRITE(*,*) 'FILE IS NON EXISTANT, REENTER FILENAME.'
       GO TO 3100
      ENDIF

      OPEN(UNIT=1, FILE=MANFIL, STATUS='OLD', FORM='UNFORMATTED')
      REWIND(1)
      RETURN
      END


      SUBROUTINE OLDOPN1
      CHARACTER*60 NAM, ANS
      LOGICAL STAT
3200  READ(*, '(A60)') NAM
      INQUIRE(FILE=NAM, EXIST=STAT)
      IF (.NOT. STAT) THEN
      WRITE(*,*) 'FILE IS NOT EXIST, ENTER NAME AGAIN'
      GOTO 3200
      ENDIF
      OPEN(UNIT=1, FILE=NAM, STATUS='OLD', FORM='FORMATTED')
      REWIND(1)
      RETURN
      END

      SUBROUTINE BOUNDSUB
      include 'con_A.con'

C     SET THE CONCENTRATION INSIDE THE BUILDING

      DO 3300 I = LB(1) , LB(2)-1
      DO 3300 J = LB(5), LB(7)-1
      DO 3300 K = 2, LB(9)-1
        CONC(I,J,K) = 0.0
```

```
3300    CONTINUE

C       SET THE CONCENTRATION SOURCE

C       SET THE INLET AND OUTLET CONCENTRATION IN X DIRECTION
        DO 3500 J=1,JMAX
        DO 3500 K=2,KMAX
         CONC(1,J,K) = CONC(2,J,K)
         CONC(IMAX,J,K) = CONC(IMAX-1,J,K)
3500    CONTINUE

C       SET THE CONCENTRATION BOUNDARY AT FRONT AND LEEWARD WALL
        DO 3600 J = LB(5), LB(7)-1
        DO 3600 K= 2, LB(9)-1
         CONC(LB(1),J,K) = CONC(LB(1)-1, J,K)
         CONC(LB(2)-1,J,K) = CONC(LB(2), J,K)
3600    CONTINUE

C       CONCENTRATION BOUNDARY ALONG SIDE WALL
        DO 3700 I=LB(1), LB(2)-1
        DO 3700 K=2, LB(9)-1
         CONC(I,LB(5),K) = CONC(I,LB(5)-1,K)
         CONC(I,LB(7)-1,K) = CONC(I, LB(7), K)
3700    CONTINUE

C       CONCENTRATION BOUNDARY AT SIDE COMPUTATION DOMAIN
        DO 3800 I=1,IMAX
        DO 3800 K=1,KMAX
         CONC(I,JMAX,K) = CONC(I,JMAX-1,K)
         conc(i,1,k) = conc(i,2,k)
3800    CONTINUE

C       CONCENTRATION BOUNDARY AT GROUND
        DO 3900 I=1,IMAX
        DO 3900 J=1,JMAX
         CONC(I,J,1) = CONC(I,J,2)
3900    CONTINUE

C       CONCENTRATION BOUNDARY AT TOP OF BUILDING
        DO 4000 I= LB(1), LB(2)-1
        DO 4000 J= LB(5), LB(7)-1
         CONC(I,J,LB(9)-1) = CONC(I,J,LB(9))
4000    CONTINUE

C       CONCENTRATION BOUNDARY AT TOP OF COMPUTATION DOMAIN
        DO 4100         I=1,IMAX
        DO 4100 J=1,JMAX
         CONC(I,J,KMAX) = CONC(I,J,KMAX-1)
4100    CONTINUE


        RETURN
        END
```

B.2. The included files

con_a.par

```
      parameter (ix=65, iy=60, iz=40)
```

con_a.con

```
c******* the included files are here ***********************************
      PARAMETER(SOURCE_RATE=12.5)
      PARAMETER(SOURCE_CONC=1.)
      include 'con_a.par'

      REAL*8 AF(IX,IY,IZ)
      COMMON AF

      REAL*8 DT, CONC(IX,IY,IZ), C(IX,IY,IZ)
      COMMON DT, CONC, C

      COMMON        U(IX,IY,IZ,4), X(IX), Y(IY), Z(IZ),
     1             S(IX,IY,IZ),GAMMA(IX,IY,IZ),
     2             DX(IX), XD(IX), DY(IY), YD(IY),
     3          DZ(IZ), ZD(IZ), DFX(IX,IY,IZ), DFY(IX
     4        ,IY,IZ), DFZ(IX,IY,IZ)
C
      COMMON/GRID/ XX(2*IX), YY(2*IY), ZZ(2*IZ)
      COMMON/BUILDING LOCATION/ LB(10)
      COMMON        IMAX,JMAX,KMAX
c********** finish of the included file ****************************
```

Appendix C

Fortran source code for TWIST — Turbulent Wind Simulation Technique — standard k-ε model approach

C.1 The main program

```
Cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
C       This code is a revised version of Twist, which is a code developed at center for building studies in
C       Concordia University by A. Baskaran and Y.S. Zhou, it can be used to calculate the wind flow
C       field around different shapes of building with different wind approaching angles.
C
C       The turbulence model used in this code is the standard k-ε model
C
C       The code is revised by Ye Li on April 4, 1997 to add the following:
C       1.      documentation
C       2.      the convergence criteria of Ferziger
C       3.      the include files are one_afer.inc, two_a.inc, three_afer.inc
C               and four_afer.inc respectively, other than the one_a.inc, two_a.inc,
C               three_a.inc and four_a.inc in the original code
C
C
c<<<<<<<<<<<<<<<<<<<  >>>>>>>>>>>>>>> <><><<<<>>>><><
      program twist
      INCLUDE 'ONE_afer.INC'
      DIMENSION SNORM(6)
      LOGICAL HELP
c
      WRITE(*,*) '!!  *** WELCOME TO TWIST **** !!'

      CALL Module1
c
c Fixing  Convergence Parameters
c
      WRITE(*,*) ' TYPE MAXIT,RESMAX,DIVERG'
        WRITE(*,*)'TYPICAL VALUES 150,0.2,20.'
      READ(*,*) MAXIT,RESMAX,DIVERG
        WRITE(*,*)'THE ROUGHNESS LENGTH FOR GROUND'
        READ(*,*)Zo
      write(*,*) ' Entre the output file name '
      READ(*,'(20a)') FNAME
      OPEN(3,FILE=FNAME,STATUS='NEW',form='unformatted')
C
        WRITE (*,*) 'Enter the output file of LAMDA'
        READ(*,'(20A)') fname
        OPEN(11, FILE=fname, STATUS='NEW', FORM='FORMATTED')

c
        WRITE (*,*) 'Enter the output file of ERRN'
        READ(*,'(20A)') fname
        OPEN(12, FILE=fname, STATUS='NEW', FORM='FORMATTED')
```

```
c
        WRITE (*,*) 'Enter the output file of RESIDUE'
        READ(*,'(20A)') fname
        OPEN(13, FILE=fname, STATUS='NEW',FORM='FORMATTED')
c
        WRITE (*,*) 'Enter the output file of LAMDAM1'
        READ(*,'(20A)') fname
        OPEN(14, FILE=fname, STATUS='NEW', FORM='FORMATTED')
c
        WRITE(11,9010)
        WRITE(12,9020)
        WRITE(13,9030)
        WRITE(14,9040)
c
     WRITE(*,*) 'Thanks for Waiting .....'
C
C Main computation
C
10     NITER = NITER+1
C-------------------------------------------------------
C----- IEQ IS THE INDEX OF THE EQUATIONS
C----- IEQ       1       2       3       4       5       6
C----- EQ        4.11    4.12    4.13    4.25    4.9     4.9
C----- VAR       U       V       W       P'      k       e
C-------------------------------------------------------
C
C-------------------------------------------------------
C-      VA(I,J,K,L) ARE ALL THE MAIN VARIABLES
C- L    1       2       3       4       5       6       7       8
C- VAR U        V       W       P'      k       e       P       vt
C-------------------------------------------------------
C
     DO 30 IEQ = 1,6
c
C Reset the Pressure Coreection Terms ...
c
        IF(IEQ.EQ.4) THEN
        DO 20 K=1,KMAX
        DO 20 J=1,JMAX
        DO 20 I=1,IMAX
20       VA(I,J,K,4)=0.0
        END IF
c
C Iteration loop Starts .....
c
c
30     CALL MODULE2(IEQ,NITER,Zo)
c
C FIND TURBULENT VISCOSITY
C VISCL IS THE LAMINAR EDDY VISCOSITY FOR AIR
C
     DO 40 K=3,KMAX-1
     DO 40 J=1,JMAX-1
     DO 40 I=2,IMAX-1
```

```
40      VA(I,J,K,8)=C(I,J,K,4)*VA(I,J,K,5)**2/VA(I,J,K,6)*URF(8)+
     &  VA(I,J,K,8)*(1.-URF(8))
C
C
C  CALCULATE RESIDUAL ERRORS AND CHECK FOR CONVERGENCE
C
      IF(NITER.NE.1) GO TO 60
      RESID=MAX(RES(1),RES(2),RES(3))
      SNORM(1)=RESID
      SNORM(2)=RESID
      SNORM(3)=RESID
      DO 50 IEQ=4,6
50     SNORM(IEQ) = RES(IEQ)
C---- INITIALIZE THE ARRAY STORE THE OLD VALUE
      DO 55 L=1, 8
        DO 55 K=1, KMAX
        DO 55 J=1, JMAX
        DO 55 I=1, IMAX
         VOLD(I,J,K,L) = 0.0
55      CONTINUE
C
C---- INITIALIZE RMSDN AND RMSDNP1
C---- RMSDN IS THE RMS VALUE OF THE DIFFERENCE BETWEEN STEP N AND N-1
C---- RMSDN = || PHI(N) - PHI(N-1) ||
C---- RMSDNP1 IS THE RMS VALUE OF THE DIFFERENCE BETWEEN STEP N+1 AND N
C---- RMSDNP1 = || PHI(N+1) - PHI(N) ||
C
        DO 57 L = 1, 8
        RMSDN(L) = 1.0
        RMSDNP1(L) = 1.0
57      CONTINUE
C
60     WRITE(*,*) NITER,RES
C
C---- CONVERGENCE CRITERIA DUE TO FERZIGER
C---- CALCULATE RMSDNP1 FOR EACH STEP
C
        DO 59 L=1,8
         TEMPAR(L) = 0.0
59      CONTINUE
C
        DO 62 L=1, 8
        DO 62 K=1, KMAX
        DO 62 J=1, JMAX
        DO 62 I=1, IMAX
         TEMPAR(L) = TEMPAR(L) + (VA(I,J,K,L) - VOLD(I,J,K,L))**2
62      CONTINUE
C
        DO 64 L=1,8
        RMSDNP1(L) = SQRT(TEMPAR(L))
64      CONTINUE
C
C---- ALAMDA IS THE BIGGEST EIGENVALUE OF THE ITERATION MATRIX
C---- ALAMDA = || PHI(N+1) - PHI(N) || / || PHI(N) - PHI(N-1) ||
```

```
C---- THE STEP NOW IS STEP N+1
C
        DO 66 L=1,8
        ALAMDA(L) = RMSDNP1(L) / RMSDN(L)
C---- ERRN IS THE RMS ERROR IN STEP N
C
        ALAMDAM1(L) = ALAMDA(L) - 1.
        IF (ALAMDAM1(L) .LE. 1.E-30) THEN
              ALAMDA(L) = 1.E-10
        END IF
        ERRN(L) = RMSDNP1(L) / ABS(ALAMDAM1(L))
C
C---- ASSIGN RMSDNP1 IN THIS STEP AS RMSDN FOR NEXT STEP
        RMSDN(L) = RMSDNP1(L)
C
 66     CONTINUE

        WRITE(11,9110) NITER, (ALAMDA(L), L=1,8)
        WRITE(12,9120) NITER, (ERRN(L), L=1,8)
        WRITE(14,9140) NITER, (ALAMDAM1(L), L=1,8)
9010    FORMAT(1X, 5HNITER, 6X,9HALAMDA(U), 6X,9HALAMDA(V),
   1            6X, 9HALAMDA(W), 5X, 10HALAMDA(P'),6X, 9HALAMDA(k),
   1            6X, 9HALAMDA(e), 6X, 9HALAMDA(P), 5X, 10HALAMDA(vt) )
9020    FORMAT(1X, 5HNITER, 8X, 7HERRN(U), 8X,7HERRN(V),
   1            8X, 7HERRN(W), 7X, 8HERRN(P'), 8X, 7HERRN(k),
   1            8X, 7HERRN(e), 8X, 7HERRN(P), 7X, 8HERRN(vt) )
9040    FORMAT(1X, 5HNITER, 6X,9HLAMDA1(U), 6X,9HLAMDA1(V),
   1            6X, 9HLAMDA1(W), 5X, 10HLAMDA1(P'),6X, 9HLAMDA1(k),
   1            6X, 9HLAMDA1(e), 6X, 9HLAMDA1(P), 5X, 10HLAMDA1(vt) )
9110    FORMAT(1X, I5, 8(1X, E14.6))
9120    FORMAT(1X, I5, 8(1X, E14.6))
9140    FORMAT(1X, I5, 8(1X, E14.6))
      DO 70 IEQ=1,6
70     RES(IEQ)=RES(IEQ)/SNORM(IEQ)
      RESID = MAX(RES(1),RES(2),RES(3),RES(4),RES(5),RES(6))
      WRITE(*,*) NITER,RES
        WRITE(13, 9130) NITER, (RES(L), L=1,6)
9030    FORMAT(1X, 5HNITER, 11X, 6HRES(U), 11X,6HRES(V),
   1            11X, 6HRES(W), 10X, 7HRES(P'), 11X, 6HRES(k),
   1            11X, 6HRES(e) )
9130    FORMAT(1X, I5, 6(1X, E16.8))
      IF(NITER.LT.MAXIT.AND.RESID.GT.RESMAX.AND.RESID.LT.DIVERG)
     & GO TO 10
      IF(RESID.GE.DIVERG) write(*,*) 'ALGORITHM DIVERGENCE'
C
C Store the output data in a file .....
c
      CALL OUTF
c
            STOP
            END
C
C
C-------------------------------------------
```

```
      SUBROUTINE MODULE1
      PARAMETER(SMALL=1.E-3)
      INCLUDE 'ONE_afer.INC'
       DIMENSION DIAG(60),BELOW(60),ABOV(60),AKINI(60),EPINI(60),
     1   ANUINI(60),AU(60),RHS(60)
       LOGICAL HELP,OUTPUT,ABORT,L1,L2,L3
     CALL GRIDGN(LB)
CCC  PRIMARY INITIALIZATION
C
C       X(I), Y(J), Z(K) IS THE CENTER OF THE CONTROL VOLUME WHERE
C       PRESSURE, K AND EPSILON ARE CALCULATED
C       XX(2*I+1) IS MIDWAY BETWEEN X(I) AND X(I+1)
C
        IMAX=IIMAX/2
        JMAX=JJMAX/2
        KMAX=KKMAX/2
       WRITE(*,*) 'IMAX=',IMAX, ' JMAX=',JMAX, ' KMAX=',KMAX
        DO 310 I = 1,IMAX
310       X(I)=XX(2*I)
       DO 311 J=1,JMAX
311    Y(J)=YY(2*J)
        DO 312 K=1,KMAX
312    Z(K)=ZZ(2*K)
       DO 320 L=1,8
         DO 320 K=1,KMAX
         DO 320 J=1,JMAX
         DO 320 I=1,IMAX
         if(l.Gt.4) go to 319
         c(i,j,k,l)=0.
         IF(L.EQ.4) C(I,J,K,L)=0.09
319        va(I,J,K,L)=0.
320        continue
c
     WRITE(*,*)' DO YOU WANT : '
     write(*,*)
     write(*,*) '   1)CALCULATE THE INITIAL FLOW FIELD
   &
   & 2)READ THE INITIAL FLOW FIELD '
     WRITE(*,*)
     READ(*,*) ICHO
     GO TO (1,2) ICHO
C
1       WRITE(*,*) ' POWER LAW VELOCITY PROFILE'
       WRITE(*,*)
       WRITE(*,*) 'TYPE ALPHA,ZREF,UREF'
       READ(*,*) ALPHA,ZREF,UREF
        WRITE(*,*)'ANG--Attacking angle'
        READ(*,*)ANG
        ANG=ANG*3.1416/180.
        write(*,*)'VELOCITY FORM; 1--POWER LAW; 2--UNIFORM FLOW'
        READ(*,*)ICHOP
        GOTO(41,42)ICHOP
C
C       CALCULATE VELOCITY
```

```
C
41      UREF1=UREF/ZREF**ALPHA
C
C
        DO 610 I=1,IMAX
        DO 610 J=1,JMAX
        DO 610 K=3,KMAX
C
C       UVREF IS THE VELOCITY AT HEIGHT Z(K). IF Z(K) IS BIGGER THAN
C       THE GRADIENT HEIGHT, THEN UVREF TAKES THE GRADIENT VELOCITY
C
        IF ( Z(K) .LE. ZREF) THEN
                UVREF=UREF1*Z(K)**ALPHA
        ELSE
                UVREF=UREF
        END IF
C
C       X-COMPONENT OF ONCOMING VELOCITY
C
        VA(I,J,K,1)=UVREF*COS(ANG)
C
C       Y-COMPONENT OF ONCOMING VELOCITY
C
        VA(I,J,K,2)=UVREF*SIN(ANG)
C
C       Z-COMPONENT OF ONCOMING VELOCITY
C
        VA(I,J,K,3)=SMALL
C
C       CHECK INTITAL VELOCITY
C
610     CONTINUE
        GOTO 611
C
42      DO 511 I=1,IMAX
        DO 511 J=1,JMAX
        DO 511 K=3,KMAX
        VA(I,J,K,1)=UREF*COS(ANG)
        VA(I,J,K,2)=UREF*SIN(ANG)
        VA(I,J,K,3)=SMALL
511     CONTINUE
611     DO 620 I=1,IMAX
        DO 620 J=1,JMAX
        VA(I,J,2,1)=VA(I,J,3,1)*Z(2)/Z(3)
        VA(I,J,2,2)=VA(I,J,3,2)*Z(2)/Z(3)
        VA(I,J,2,3)=SMALL
620     CONTINUE
C
C
        WRITE(*,*)'READ IN FACT1 & FACT2,(.4 &.3)'
        READ(*,*)FACT1,FACT2
C
C  FIND THE TURBULENT KINETIC ENERGY AND ITS DISSIPATION
C
```

```
      WRITE(*,*) 'KMAX=', KMAX
         DO 90 I=1,1
      DO 90 J=1,1
      DO 90 K=2,KMAX
      U = VA(I,J,K,1)
      UU = VA(I+1,J,K,1)
      V = VA(I,J,K,2)
      IF(J.NE.JMAX) THEN
       VV = VA(I,J+1,K,2)
      ELSE
       VV = V
      ENDIF
      W = VA(I,J,K,3)
      IF(K.NE.KMAX) THEN
       WW = VA(I,J,K+1,3)
      ELSE
       WW = W
      ENDIF
      USQ = .25*((U+UU)**2 + (V+VV)**2 + (W+WW)**2)
      AKINI(K-1)= .584*USQ/(Z(K)**ALPHA/ALPHA)**2
      EPINI(K-1) = .391*AKINI(K-1)**1.5*Z(K)**(ALPHA-1.)
      ANUINI(K-1)= C(I,J,K,4)*AKINI(K-1)**2/EPINI(K-1)

C     WRITE(*,*) 'EPINI= ', EPINI(K-1), 'K=', K

90    CONTINUE
      WRITE(*,*) 'Z(3)=', Z(3)
      AKINI(1)=AKINI(2)*(Z(2)/Z(3))**2

      EPINI(1)=2*0.133*AKINI(1)/Z(2)**2
      ANUINI(1)=C(I,J,2,4)*AKINI(1)**2/EPINI(1)
      NN=0.
111   ACMAX=0.
      NN=NN+1
      DIAG(1)=1.
      DIAG(KMAX-1)=1.
      BELOW(1)=0.
      BELOW(KMAX-1)=-1.
      ABOV(1)=-(Z(2)/Z(3))**2
      ABOV(KMAX-1)=0.
      RHS(1)=0.
      RHS(KMAX-1)=0.
      DO 112 K=2,KMAX-2
      DIAG(K)=-2*0.09*(1/(Z(K+2)-Z(K+1))+1/(Z(K+1)-Z(K)))/
     1 (PRTE*(Z(K+2)-Z(K)))-EPINI(K)**2/AKINI(K)**3
      BELOW(K)=-2*0.09*(AKINI(K+1)-AKINI(K-1))/(AKINI(K)*
     1 PRTE*(Z(K+2)-Z(K))**2)+
     1 2*0.09/(PRTE*(Z(K+2)-Z(K))*(Z(K+1)-Z(K)))+0.09*(EPINI(K+1)-
     1 EPINI(K-1))/(PRTE*EPINI(K)*(Z(K+2)-Z(K))**2)
      ABOV(K)=2*0.09*(AKINI(K+1)-AKINI(K-1))/(AKINI(K)*PRTE*(Z(K+2)-
     2 Z(K))**2)+2*0.09/(PRTE*(Z(K+2)-Z(K))*(Z(K+2)-Z(K+1)))
     3 -0.09*(EPINI(K+1)-EPINI(K-1))/(PRTE*EPINI(K)*(Z(K+2)-Z(K))**2)
      RHS(K)=-0.09*(ALPHA*UREF1*Z(K+1)**(ALPHA-1))**2
112   CONTINUE
```

```
      CALL TRID(ABOV,BELOW,DIAG,RHS,AU,KMAX-1)
      DO 102 K=2,KMAX-2
      AC=ABS(AU(K)-AKINI(K))
102   ACMAX=AMAX1(AC,ACMAX)
      DO 103 K=1,KMAX-1
103   AKINI(K)=FACT1*AU(K)+(1.-FACT1)*AKINI(K)
      DIAG(1)=1.
      DIAG(KMAX-1)=1.
      BELOW(1)=0.
      BELOW(KMAX-1)=-1.
      ABOV(1)=0.
      ABOV(KMAX-1)=0.
      RHS(1)=2*0.133*AKINI(2)/Z(3)**2
      RHS(KMAX-1)=0.
      DO 114 K=2,KMAX-2
      DIAG(K)=-2*0.09*AKINI(K)*(1/(Z(K+2)-Z(K+1))+1/(Z(K+1)-Z(K)))/
     1(PRED*(Z(K+2)-Z(K))*EPINI(K))-1.92*EPINI(K)/AKINI(K)**2
      BELOW(K)=-2*0.09*(AKINI(K+1)-AKINI(K-1))/(PRED*
     2EPINI(K)*(Z(K+2)-Z(K))**2)+
     1 2*0.09*AKINI(K)/(PRED*EPINI(K)*(Z(K+2)-Z(K))*(Z(K+1)-Z(K)))
     3 +0.09*AKINI(K)*(EPINI(K+1)-EPINI(K-1))/(PRED*(EPINI(K)*(Z(K+2)-
     4 Z(K)))**2)
      ABOV(K)=2*0.09*(AKINI(K+1)-AKINI(K-1))/(PRED*EPINI(K)*
     2(Z(K+2)-Z(K))**2)+2*0.09*AKINI(K)/
     2(PRED*EPINI(K)*(Z(K+2)-Z(K))*(Z(K+2)-Z(K+1)))-0.09*AKINI(K)*(
     3 EPINI(K+1)-EPINI(K-1))/(PRED*(EPINI(K)*(Z(K+2)-Z(K)))**2)
      RHS(K)=-1.44*0.09*(ALPHA*UREF1*Z(K+1)**(ALPHA-1))**2
114   CONTINUE
      CALL TRID(ABOV,BELOW,DIAG,RHS,AU,KMAX-1)
      DO 105 K=2,KMAX-2
      AC=ABS(AU(K)-EPINI(K))
105   ACMAX=AMAX1(AC,ACMAX)
      DO 106 K=1,KMAX-1
106   EPINI(K)=FACT2*ABS(AU(K))+(1.-FACT2)*EPINI(K)
      IF(NN.GE.1000)WRITE(*,*)'CHOOSE ANO SET OF FACT1 &
     1 FACT2,TRY AGAIN'
      IF(ACMAX.GE.0.1) GOTO 111
      DO 107 K=1,KMAX-1
107   ANUINI(K)=0.09*AKINI(K)**2/EPINI(K)
C
C     CHECK INITIAL VELOCITY FIELD
C
      WRITE (*,*) 'Enter the output file of initial velocity field'
      READ(*,'(20A)') fname
      OPEN(91, FILE=fname, STATUS='NEW', FORM='FORMATTED')
      DO 108 K=1,KMAX
       WRITE(91,*) VA(1,1,K,1), z(k)
108   CONTINUE
      CLOSE(91)
C
C
C     CHECK INITIAL EPSILON and K
C
      WRITE (*,*) 'Enter the output file of initial k and epsilon field'
```

```
      READ(*,'(20A)') fname
      OPEN(91, FILE=fname, STATUS='NEW', FORM='FORMATTED')
      DO 109 K=1, KMAX
     WRITE(91,*)  AKINI(K), EPINI(K), Z(K)
109   CONTINUE
      CLOSE(91)
C
C
C-- SET INITIAL TURBULENCE PROPERTIES
      DO 212 I=1,IMAX
      DO 212 J=1,JMAX
      DO 212 K=2,KMAX
      VA(I,J,K,5)=AKINI(K-1)
      VA(I,J,K,6)=EPINI(K-1)
212   VA(I,J,K,8)=ANUINI(K-1)
     DO 80 I = LB(1),LB(2)
     DO 80 J = LB(5),LB(7)
     DO 80 K = 2,LB(9)
    IF(J.NE.LB(7).AND.K.NE.LB(9))va(I,J,K,1) = SMALL
    IF(I.NE.LB(2).AND.K.NE.LB(9))va(I,J,K,2) = SMALL
      IF(I.NE.LB(2).AND.J.NE.LB(7))VA(I,J,K,3)=SMALL
80    CONTINUE
      DO 81 I=LB(3),LB(4)
      DO 81 J=LB(6),LB(8)
      DO 81 K=1,LB(10)
      IF(J.NE.LB(8).AND.K.NE.LB(10))VA(I,J,K,1)=SMALL
      IF(I.NE.LB(4).AND.K.NE.LB(10))VA(I,J,K,2)=SMALL
      IF(I.NE.LB(4).AND.J.NE.LB(8))VA(I,J,K,3)=SMALL
81    CONTINUE
C
C CONTINUITY MUST BE SATISFIED GLOBALLY
        SUM=0.
      DO 110 J=2,JMAX-1
        YJ=.5*(Y(J+1)-Y(J-1))
      DO 110 K=2,KMAX-1
        ZK = .5*(Z(K+1)-Z(K-1))
110       SUM=SUM+(va(IMAX,J,K,1)-va(2,J,K,1))*YJ*ZK
      DO 130 I=2,IMAX-1
        XI=.5*(X(I+1)-X(I-1))
      DO 120 K=2,KMAX-1
        ZK=.5*(Z(K+1)-Z(K-1))
120       SUM=SUM+(va(I,JMAX,K,2)-VA(I,2,K,2))*XI*ZK
      DO 130 J=2,JMAX-1
        YJ=.5*(Y(J+1)-Y(J-1))
130       SUM=SUM+va(I,J,KMAX,3)*XI*YJ
C
C  ADJUST VERTICAL VELOCITY AR TOP OF GRID
     AREAX=(XX(IMAX-1)-XX(3))*(X(IMAX-1)-XX(3))*(YY(JJMAX-1)
   1    -YY(3))
      DO 140 I=1,IMAX
        DELTA=SUM*(X(I)-XX(3))/AREAX
      DO 140 J=1,JMAX
140       va(I,J,KMAX,3)=va(I,J,KMAX,3)-DELTA
C
```

```
C  OUTPUT INITIAL CONDITIONS TO FILE IF REQUIRED
C
      CALL HELPER('DO YOU WANT TO STORE INITIAL FLOW FIELD',OUTPUT)
         IF(.NOT.OUTPUT) GO TO 233
C
       WRITE(*,*)'ENTRE THE FILE NAME'
       READ(*,'(20A)') FNAME
       OPEN(2,FILE=FNAME,STATUS='NEW')
       WRITE(2,*)IMAX,JMAX,KMAX
       WRITE(2,'(8F10.3)') (X(I),I=1,IMAX)
       WRITE(2,'(8F10.3)')  (Y(J), J=1,JMAX)
       WRITE(2,'(8F10.3)')  (Z(K),K=1,KMAX)
          DO 200 L=1,8
          IF(L.EQ.4.OR.L.EQ.7) GO TO 200
          WRITE(2,*) l
          DO 190 K=1,KMAX
          write(2,*) k
          DO 190 J=1,JMAX
          write(2,*) j
          WRITE(2,'(8F10.3)') (VA(I,J,K,L),I=1,IMAX)
190         CONTINUE
200         CONTINUE
          CLOSE(2,STATUS='KEEP')
             go to 233
C
2       WRITE(*,*)' Entre the file name'
       READ(*,'(20A)')  FNAME
       OPEN(2,FILE=FNAME,STATUS='old')
       read(2,*)IMAX,JMAX,KMAX
       read(2,'(8F10.3)') (X(I),I=1,IMAX)
       read(2,'(8F10.3)') (Y(J), J=1,JMAX)
       read(2,'(8F10.3)') (Z(K),K=1,KMAX)
          DO 202 L=1,8
          IF(L.EQ.4.OR.L.EQ.7) GO TO 202
          read(2,*) lpos
          DO 191 K=1,KMAX
          read(2,*) kpos
          DO 191 J=1,JMAX
          read(2,*) jpos
          read(2,'(8F10.3)') (VA(I,J,K,L),I=1,IMAX)
191         CONTINUE
202         CONTINUE
         CLOSE(2,STATUS='KEEP')
C
C  SET UP UNDERRELAXION FACTORS
233          URF(1)=.5
          URF(2)=.5
          urf(3)=.5
          URF(4)=1.0
          URF(5)=.7
          URF(6)=.7
          URF(7)=.3
          URF(8)=.7
                RETURN
```

```
            END
C-------------------------------
C--       SOLVE TRIDIAGONAL MATRIX
          SUBROUTINE TRID(ABOV,BELO,DIAG,RHS,U,N)
          DIMENSION ABOV(N),BELO(N),DIAG(N),RHS(N),U(N)
          DO 10 I=2,N
          RATIO=BELO(I)/DIAG(I-1)
          DIAG(I)=DIAG(I)-RATIO*ABOV(I-1)
10        RHS(I)=RHS(I)-RATIO*RHS(I-1)
          RHS(N)=RHS(N)/DIAG(N)
          DO 20 I=2,N
          J=N-I+1
20        RHS(J)=(RHS(J)-ABOV(J)*RHS(J+1))/DIAG(J)
          DO 21 I=1,N
21        U(I)=RHS(I)
          RETURN
          END
C_____________________________________________________________
           SUBROUTINE GRIDGN(LB)
           INCLUDE 'TWO_A.INC'
          REAL LX1,LX2,MD
CCCC READ BUILDING PARAMETERS
       WRITE(*,*)'H1,THE HEIGHT OF FRONT BLDG'
          WRITE(*,*)'H2,THE HEIGHT OF BACK BLDG'
       READ(*,*) H1,H2
C
C         VERIFY
C
          WRITE(*,*) 'H1= ', H1
          WRITE(*,*) 'H2= ', H2
C
          WRITE(*,*)'DO YOU WANT TO:'
       write(*,*)  '1) use the grid generation routine
     &  2) read the grid locations from a file'
       read(*,*) icho
       go to (1,2)icho
c
1         IF(H1.LE.H2)THEN
          HZ1=H1
          HZ2=H2-H1
          GOTO 101
          ENDIF
          HZ1=H2
          HZ2=H1-H2
101       WRITE(*,*)'NHZ1=?,NHZ2=? (IF H1=H2,NHZ2=0)'
          READ(*,*)NHZ1,NHZ2
C
C         VERIFY
C
          WRITE(*,*) 'NHZ1= ', NHZ1
          WRITE(*,*) 'NHZ2= ', NHZ2
C
          WRITE(*,*)'UD,LX1,LX2,DD,DS1,W1,W2(NON 0.),W3,DS2=?'
          READ(*,*)UD,LX1,LX2,DD,DS1,W1,W2,W3,DS2
```

```
C
C       VERIFY THE INPUT
        WRITE(*,*) 'UD= ', UD
        WRITE(*,*) 'LX1= ', LX1
        WRITE(*,*) 'LX2= ', LX2
        WRITE(*,*) 'DD= ', DD
        WRITE(*,*) 'DS1= ', DS1
        WRITE(*,*) 'W1= ', W1
        WRITE(*,*) 'W2= ', W2
        WRITE(*,*) 'W3= ', W3
        WRITE(*,*) 'DS2= ', DS2
C
C
        WRITE(*,*)'NUD,NLX1,NLX2,NDD,NDS1,NW1,NW2,NW3,NDS2=?'
        READ(*,*)NUD,NLX1,NLX2,NDD,NDS1,NW1,NW2,NW3,NDS2
C
C       VERIFY THE INPUT
C
        WRITE(*,*) 'NUD= ', NUD
        WRITE(*,*) 'NLX1= ', NLX1
        WRITE(*,*) 'NLX2= ', NLX2
        WRITE(*,*) 'NDD= ', NDD
        WRITE(*,*) 'NDS1= ', NDS1
        WRITE(*,*) 'NW1= ', NW1
        WRITE(*,*) 'NW2= ', NW2
        WRITE(*,*) 'NW3= ', NW3
        WRITE(*,*) 'NDS2= ', NDS2
C
C
        WRITE(*,*)'DT,NDT-LENGTH & GRID NO. OVER TOP'
        READ(*,*)DT,NDT
C
C       VERIFY THE INPUT
C
        WRITE(*,*) 'DT= ', DT
        WRITE(*,*) 'NDT= ',NDT
C
        WRITE(*,*)'THE DISTANCE & GRID No. BETWEEN TWO BLDGS'
        WRITE(*,*)'MD, NMD=?'
        READ(*,*)MD,NMD
C
C       VERIFY THE INPUT
C
        WRITE(*,*) 'MD= ', MD
        WRITE(*,*) 'NMD= ', NMD
C
        WRITE(*,*)'THE INITIAL STEP SIZES FOR GRID'
        WRITE(*,*)'IN THE ORDER OF (DGF,DGB,DGS,DGG,DGT)'
        READ(*,*) DGF,DGB,DGS,DGG, DGT
C
C       VERIFY THE INPUT
C
        WRITE(*,*) 'DGF= ', DGF
        WRITE(*,*) 'DGB= ', DGB
```

```
        WRITE(*,*) 'DGS= ', DGS
        WRITE(*,*) 'DGG= ', DGG
        WRITE(*,*) 'DGT= ', DGT
C
C     GRID GENERATING ROUTINE STARTS HERE
C
10      IIMAX=2*(NUD+NLX1+NMD+NLX2+NDD)+1
       JJMAX=2*(NDS1+NW1+NW2+NW3+NDS2)+1
       KKMAX=2*(NHZ1+NHZ2+NDT)+3
        WRITE(*,*) 'IIMAX=',IIMAX, ' JJMAX=',JJMAX, ' KKMAX=', KKMAX
         XX(2*NUD+1) =0.
         XX(2*(NUD+NLX1)+1)=Lx1
        XX(2*(NUD+NLX1+NMD)+1)=LX1+MD
        XX(2*(NUD+NLX1+NMD+NLX2)+1)=LX1+LX2+MD
         YY(2*NDS1+1)=-W1
        YY(2*(NDS1+NW1)+1)=0.
        YY(2*(NDS1+NW1+NW2)+1)=W2
        YY(2*(NDS1+NW1+NW2+NW3)+1)=W2+W3
      ZZ(3)=0.
        ZZ(2)=-0.5
        ZZ(1)=-1.0
       ZZ(2*NHZ1+3)=HZ1
        ZZ(2*(NHZ1+NHZ2)+3)=HZ1+HZ2
C
C   computes the building locations
C
C
C       first grid inside the front wall
C
        LB(1)=NUD+1
C
C       first grid outside of the realwall of the first building
C
        LB(2)=NUD+1+NLX1
C
C       first grid inside the front wall of the second building
C
        LB(3)=LB(2)+NMD
C
C       first grid outside the real wall of the second building
C
        LB(4)=LB(3)+NLX2
C
C       first gridinside side wall 1 of the first building
C
        LB(5)=NDS1+1
C
C       first grid inside side wall 1 of the first building
C
        LB(6)=NDS1+1+NW1
C
C       first grid outside side wall 2 of the first building
C
        LB(7)=NDS1+1+NW1+NW2
```

```
C
C       first grid outside side wall 2 of the second building
C
        LB(8)=LB(7)+NW3
C
        IF(H1.LE.H2)THEN
C
C       first grid outside roof of first building
C
        LB(9)=NHZ1+2
C
C       first grid outside roof of second building
C
        LB(10)=NHZ1+NHZ2+2
C
        GOTO 82
        ENDIF
        LB(9)=NHZ1+NHZ2+2
        LB(10)=NHZ1+2
c for X direction
82     CALL FINDB(NUD,UD,DGF,B)
        WRITE(*,*)'BUD='
        WRITE(*,*)B
     CALL SETGRD(XX,iimax,2*NUD+1,1,DGF,B,1)
        IF((NLX1/2)*2.EQ.NLX1)THEN
        NLX11=NLX1/2
        NLX12=NLX1/2
        GOTO 112
        ENDIF
        NLX11=(NLX1+1)/2
        NLX12=NLX1/2
112      CALL FINDB(NLX11,Lx1/2,DGF,B)
        WRITE(*,*)'B2='
        WRITE(*,*)B
     CALL SETGRD(XX,iimax,2*NUD+1,2*NUD+1+2*NLX11,DGF,B,2)
        CALL FINDB(NLX12,LX1/2,DGB,B)
        WRITE(*,*)'BLX1='
        WRITE(*,*)B
     CALL SETGRD(XX,iimax,2*NUD+1+2*NLX1,2*NUD+1+2*NLX11,DGB,B,1)
        IF(NMD/2*2.EQ.NMD)THEN
        NMD1=NMD/2
        NMD2=NMD1
        GOTO 120
        ENDIF
        NMD1=(NMD+1)/2
        NMD2=NMD/2
120     CALL FINDB(NMD1,MD/2,DGF,B)
        WRITE(*,*)'BMD=(<1.5)'
        WRITE(*,*)B
        CALL SETGRD(XX,IIMAX,2*(NUD+NLX1)+1,2*(NUD+NLX1+NMD1)+1,DGF,
   1    B,2)
        CALL FINDB(NMD2,MD/2,DGB,B)
        CALL SETGRD(XX,IIMAX,2*(NUD+NLX1+NMD)+1,2*(NUD+NLX1+NMD1)+1,
   1    DGB,B,1)
```

```
        IF((NLX2/2)*2.EQ.NLX2)THEN
        NLX21=NLX2/2
        NLX22=NLX21
        GOTO 113
        ENDIF
        NLX21=(NLX2+1)/2
        NLX22=NLX2/2
113   CALL FINDB(NLX21,LX2/2,DGF,B)
        WRITE(*,*)'BLX2='
        WRITE(*,*)B
     CALL SETGRD(XX,iimax,2*(NUD+NLX1+NMD)+1,2*(NUD+NLX1+NLX21+NMD)
    1   +1,DGF,B,2)
        CALL FINDB(NLX22,LX2/2,DGB,B)
        LN2=2*(NUD+NLX1+NLX2+NMD)+1
        CALL SETGRD(XX,IIMAX,LN2,2*(NUD+NLX1+NMD+NLX21)+1,DGB,B,1)
        CALL FINDB(NDD,DD,DGB,B)
        WRITE(*,*)'BDD='
        WRITE(*,*)B
        CALL SETGRD(XX,IIMAX,LN2,IIMAX,DGB,B,2)
CCC
Cc for Y direction
CCC
        IF(ABS(W1).LT.0.01)GOTO 114
        IF((NW1/2)*2.EQ.NW1) THEN
        NW11=NW1/2
        NW12=NW11
        GOTO 121
        ENDIF
        NW11=(NW1+1)/2
        NW12=NW1/2
121     CALL FINDB(NW11,W1/2,DGS,B)
        WRITE(*,*)'BW1='
        WRITE(*,*)B
        CALL SETGRD(YY,JJMAX,2*NDS1+1,2*(NDS1+NW11)+1,DGS,B,2)
        CALL FINDB(NW12,W1/2,DGS,B)
        CALL SETGRD(YY,JJMAX,2*(NDS1+NW1)+1,2*(NDS1+NW11)+1,DGS,B,1)
114     IF((NW2/2)*2.EQ.NW2) THEN
        NW21=NW2/2
        NW22=NW21
        GOTO 122
        ENDIF
        NW21=(NW2+1)/2
        NW22=NW2/2
122     CALL FINDB(NW21,W2/2,DGS,B)
        LN3=2*(NDS1+NW1)+1
        CALL SETGRD(YY,JJMAX,LN3,LN3+2*NW21,DGS,B,2)
        CALL FINDB(NW22,W2/2,DGS,B)
        WRITE(*,*)'BW2='
        WRITE(*,*)B
        CALL SETGRD(YY,JJMAX,LN3+2*NW2,LN3+2*NW21,DGS,B,1)
    CALL FINDB(NDS1,DS1,DGS,B)
        WRITE(*,*)'BDS1='
        WRITE(*,*)B
     CALL SETGRD(YY,jjmax,2*NDS1+1,1,DGS,B,1)
```

```
      IF(ABS(W3).LT.0.01) GOTO 115
      IF((NW3/2)*2.EQ.NW3) THEN
      NW31=NW3/2
      NW32=NW31
      GOTO 123
      ENDIF
      NW31=(NW3+1)/2
      NW32=NW3/2
123   CALL FINDB(NW31,W3/2,DGS,B)
      WRITE(*,*)'BW3='
      WRITE(*,*)B
      CALL SETGRD(YY,JJMAX,LN3+2*NW2,LN3+2*(NW2+NW31),
     1 DGS,B,2)
      CALL FINDB(NW32,W3/2,DGS,B)
      CALL SETGRD(YY,JJMAX,LN3+2*NW2+2*NW3,LN3+2*NW2+2*NW31,DGS,
     1 B,1)
115   CALL FINDB(NDS2,DS2,DGS,B)
      WRITE(*,*)'BDS2='
      WRITE(*,*)B
      CALL SETGRD(YY,JJMAX,LN3+2*(NW2+NW3),JJMAX,DGS,B,2)
CCC
c for Z direction
CCC
      IF((NHZ1/2)*2.EQ.NHZ1)THEN
      NHZ11=NHZ1/2
      NHZ12=NHZ11
      GOTO 116
      ENDIF
      NHZ11=(NHZ1+1)/2
      NHZ12=NHZ1/2
116   CALL FINDB(NHZ11,HZ1*DGG/(DGG+DGT),DGG,B)
      CALL SETGRD(ZZ,KKMAX,3,2*NHZ11+3,DGG,B,2)
      CALL FINDB(NHZ12,HZ1*DGT/(DGG+DGT),DGT,B)
      WRITE(*,*)'B12='
      WRITE(*,*)B
      CALL SETGRD(ZZ,KKMAX,2*NHZ1+3,2*NHZ11+3,DGT,B,1)
      IF(ABS(HZ2).LT.0.01)GOTO 117
      IF((NHZ2/2)*2.EQ.NHZ2)THEN
      NHZ21=NHZ2/2
      NHZ22=NHZ21
      GOTO 118
      ENDIF
      NHZ21=(NHZ2+1)/2
      NHZ22=NHZ2/2
118   CALL FINDB(NHZ21,HZ2/2,DGT,B)
      WRITE(*,*)'B13='
      WRITE(*,*)B
      CALL SETGRD(ZZ,KKMAX,2*NHZ1+3,2*(NHZ1+NHZ21)+3,DGT,B,2)
      CALL FINDB(NHZ22,HZ2/2,DGT,B)
      WRITE(*,*)'B15='
      WRITE(*,*)B
      CALL SETGRD(ZZ,KKMAX,KKMAX-2*NDT,2*(NHZ1+NHZ21)+3,DGT,B,1)
117   CALL FINDB(NDT,DT,DGT,B)
      CALL SETGRD(ZZ,KKMAX,KKMAX-2*NDT,KKMAX,DGT,B,2)
```

```
CCc
C  OUTPUT TO FILE IF REQUIRED
CCC
      CALL HELPER(' DO YOU WANT STORE GRID DETAILS ?',OUTPUT )
       IF(.NOT.OUTPUT) return
       WRITE(*,*) ' TYPE THE FILE NAME '
       READ(*,'(20a)') FNAME
       OPEN(1,FILE =FNAME,STATUS='NEW')
       WRITE(1,*) IIMAX,JJMAX,KKMAX
       WRITE(1,'(8F10.3)') (XX(I),I=1,IIMAX)
       WRITE(1,'(8F10.3)') (YY(j),j=1,JJMAX)
       WRITE(1,'(8F10.3)') (ZZ(k),k=1,KKMAX)
       write(1,'(9I4)')   (lb(i),i=1,10)
       CLOSE(1,STATUS='KEEP')
       return
2      write(*,*) 'entre the grid location file'
       read(*,'(20a)') fname
       OPEN(1,FILE =FNAME,STATUS='old')
       read(1,*) IIMAX,JJMAX,KKMAX
       read(1,'(8F10.3)') (XX(I),I=1,IIMAX)
       read(1,'(8F10.3)') (YY(j),j=1,JJMAX)
       read(1,'(8F10.3)') (ZZ(k),k=1,KKMAX)
       READ(1,*) (LB(I),I=1,10)
       CLOSE(1,STATUS='KEEP')
          RETURN
          END
C
C-----------------------------------------------------

      SUBROUTINE SETGRD (X,NX,N1,N2,D,B,isweep)
C
C   SET PART OF THE COORDIATE VECTOR X
        DIMENSION X(NX)
       IF(N1.EQ.N2) RETURN
        N=N1
        A=D
        IF(isweep.EQ.2) GO TO 20
C
C   WORKING BACKWARDS THROUGH VECTOR
C
10        N=N-2
       X(N+1)=X(N+2)-A
       IF(N.LT.N2) RETURN
       A=A*B
       X(N)=X(N+1)-A
       IF(N.GT.N2) GO TO 10
       RETURN
C
C   WORKING FORWARDS THROUGH VECTOR
C
20      N=N+2
       X(N-1)=X(N-2)+A
       IF(N.GT.N2) RETURN
       A=A*B
```

```
          X(N)=X(N-1)+A
          IF(N.LT.N2) GO TO 20
           RETURN
           END
C
C-------------------------------------------------
C
          SUBROUTINE FINDB(NG,G,DO,B)
C
C  FINDS EXPANSION FACTOR FROM GEOMETRIC PROGRESSION
          IF(ABS(G).LE.0.00001.OR.NG.LE.0)RETURN
          AL1=G/DO
          B=1
          J=1
11        F=1
          J=J+1
          IF(J.GT.200) WRITE(*,*)B,AL1,AL2
          IF(NG.EQ.1) THEN
          B=1
          RETURN
          ENDIF
          DO 20 I=1,NG-1
20        F=F+B**I
30        AL2=F*(B+1)
          AL3=0.1*AL1
          IF(ABS(AL2-AL1).GT.AL3.AND.AL2.LT.AL1)B=B+0.02*B
          IF(ABS(AL2-AL1).GT.AL3.AND.AL2.GT.AL1)B=B-0.02*B
          IF(ABS(AL2-AL1).GT.AL3)GOTO 11
           RETURN
           END
C-------------------------------------------------------
       SUBROUTINE MODULE2(IEQ,NITER,Zo)
         INCLUDE 'FOUR_afer.INC'
         DIMENSION tridi(4,100),OV(100)
C
         I1=2
         I2=IMAX-1
         J1=2
         J2=JMAX-1
         K1=2
         K2=KMAX-1
         IF(IEQ.EQ.1) I1=3
         IF(IEQ.EQ.2) J1=3
         IF(IEQ.EQ.3) K1=3
CCCc  ........  ........  ......  ......
C SWEEP IN X DIRECTION
CCCC
          ISWEEP=1
          LEN=K2-K1+1
         DO 20 I=i1,i2
         DO 20 J=J1,J2
         DO 10 K=K1,K2
10      CALL assembler(IEQ,I,J,K,I1,J1,K1,TRIDI,ISWEEP,LEN,NITER,Zo)
          TO=va(I,J,K1-1,IEQ)
```

```
          T1=va(I,J,K2+1,IEQ)
          CALL tms(TRIDI,OV,LEN,TO,T1)
         DO 20 L=1,LEN
20         va(I,J,K1+L-1,IEQ) = OV(L)
c
C SWEEP IN Y DIRECTION
C
          ISWEEP=2
          LEN=I2-I1+1
         DO 70 J=J1,J2
         DO 70 K=K1,K2
         DO 60 I=I1,I2
60     CALL assembler(IEQ,I,J,K,I1,J1,K1,TRIDI,ISWEEP,LEN,NITER,Zo)
          TO=va(I1-1,J,K,IEQ)
          T1=va(I2+1,J,K,IEQ)
          CALL Tms(TRIDI,OV,LEN,TO,T1)
         DO 70 L=1,LEN
70         va(I1+L-1,J,K,IEQ)=OV(L)
C
C SWEEP IN Z DIRECTION

        isweep=3
        RES(IEQ)=0.
        LEN=J2-J1+1
       DO 40 K=K1,K2
       DO 40 I=I1,I2
       DO 30 J=J1,J2
30     CALL assembler(IEQ,I,J,K,I1,J1,K1,tridi,isweep,LEN,NITER,Zo)
        TO=va(I,J1-1,K,IEQ)
        T1=va(I,J2+1,K,IEQ)
        CALL Tms(TRIDI,OV,LEN,TO,T1)
       DO 40 L=1,LEN
        va(I,J1+L-1,K,IEQ)=OV(L)
40      continue
C
C
C UPDATE VELOCITIES AND PRESSURE
C
        IF(IEQ.LT.4) RETURN
C
        IF(IEQ.EQ.4) THEN
          DO 50 I=I1,I2
          DO 50 J=J1,J2
          DO 50 K=K1,K2
      IF(I.NE.2) VA(I,J,K,1)=VA(I,J,K,1)+C(I,J,K,1)*(va(I-1,J,K,4)-
     &  va(I,J,K,4))
      IF(J.NE.2) VA(I,J,K,2)=VA(I,J,K,2)+C(I,J,K,2)*(va(I,J-1,K,4)-
     &  va(I,J,K,4))
      IF(K.NE.2) VA(I,J,K,3)=VA(I,J,K,3)+C(I,J,K,3)*(va(I,J,K-1,4)
     &  -va(I,J,K,4))
50     VA(I,J,K,7)=VA(I,J,K,7)+URF(7)*va(I,J,K,4)
        ENDIF
C
       IF(IEQ.EQ.5) THEN
```

```
        DO 80 I = LB(1),LB(2)-1
        DO 80 J = LB(5),LB(7)-1
        DO 80 K = 2,LB(9)-1
        va(I,J,K,5) = TINY
80       CONTINUE
        DO 81 I=LB(3),LB(4)-1
        DO 81 J=LB(6),LB(8)-1
        DO 81 K=2,LB(10)-1
        VA(I,J,K,5)=TINY
81      CONTINUE
      ENDIF
      IF(IEQ.EQ.6) THEN
        DO 8 I = LB(1),LB(2)-1
        DO 8 J = LB(5),LB(7)-1
        DO 8 K = 2,LB(9)-1
        va(I,J,K,6) = TINY
8       CONTINUE
        DO 9 I=LB(3),LB(4)-1
        DO 9 J=LB(6),LB(8)-1
        DO 9 K=2,LB(10)-1
        VA(I,J,K,6)=TINY
9       CONTINUE
      ENDIF
C
         RETURN
         END
C
C-------------------------------------------------
      SUBROUTINE assembler(IEQ,I,J,K,I1,J1,K1,TRIDI,ISWEEP,
     &  LEN,NITER,Zo)
      INCLUDE 'FOUR_afer.INC'
      DIMENSION F(6),D(6),tridi(4,len)
C
C  GENERAL EQUATION FIRST
C
         VISC=va(I,J,K,8)
C        PRED=CAPPA*CAPPA/(C2-C1)/(C(I,J,K,4)**.5)
       IF(IEQ.GE.5) THEN
         VIS(1)=0.5*(VISC+VA(I+1,J,K,8))
         VIS(2)=0.5*(VISC+VA(I-1,J,K,8))
         VIS(3)=0.5*(VISC+VA(I,J+1,K,8))
         VIS(4)=0.5*(VISC+VA(I,J-1,K,8))
         VIS(5)=0.5*(VISC+VA(I,J,K+1,8))
         VIS(6)=0.5*(VISC+VA(I,J,K-1,8))
       IF(IEQ.EQ.5) THEN
       DO 40 L=1,6
40      - VIS(L)=VIS(L)/PRTE
         VISC=VISC/PRTE
       ELSE
       DO 50 L=1,6
50        VIS(L)=VIS(L)/PRED
         VISC=VISC/PRED
       END IF
       END IF
```

```
C
          DXE=X(I+1)-X(I)
          DXW=X(I)-X(I-1)
          DX=0.5*(DXE+DXW)
          DYN=Y(J+1)-Y(J)
          DYS=Y(J)-Y(J-1)
          DY= 0.5*(DYN+DYS)
          DZT=Z(K+1)-Z(K)
          DZB=Z(K)-Z(K-1)
          DZ=0.5*(DZT+DZB)
C
          UE=va(I+1,J,K,1)
          UW=va(I,J,K,1)
          VN=va(I,J+1,K,2)
          VS=va(I,J,K,2)
          WT=va(I,J,K+1,3)
          WB=va(I,J,K,3)
C
         if(ieq.GE.4) go to 30
C
C  X DIRECTION MOMENTUM EQUATION
C       control volume face is at grid point. Velocity u is at the center
C       of control volume (x-direction)
C
C
      IF(IEQ.NE.1) GO TO 10
        DXE=DX
        DX=DXW
        DXW=0.5*(X(I)-X(I-2))
        VIS(1)=VISC
        VIS(2)=va(I-1,J,K,8)
        VISC=VISC+VIS(2)
        VIS(3)=0.25*(VISC+va(I,J+1,K,8)+va(I-1,J+1,K,8))
        VIS(4)=0.25*(VISC+va(I,J-1,K,8)+va(I-1,J-1,K,8))
        VIS(5)=0.25*(VISC+va(I,J,K+1,8)+va(I-1,J,K+1,8))
        VIS(6)=0.25*(VISC+va(I,J,K-1,8)+va(I-1,J,K-1,8))
        VISC=0.5*VISC
          VN=0.5*(VN+va(I-1,J+1,K,2))
          VS=0.5*(VS+va(I-1,J,K,2))
          WT=0.5*(WT+va(I-1,J,K+1,3))
          WB=0.5*(WB+va(I-1,J,K,3))
          AA=0.5*DX/DXE
          UE=AA*UE+(1.-AA)*UW
          AA=0.5*DX/DXW
          UW=AA*va(I-1,J,K,1)+(1.-AA)*UW
      GO TO 30
C
C Y DIRECTION MOMETUM EQUATION
C
10     IF(IEQ.NE.2) GO TO 20
        DYN=DY
        DY=DYS
        DYS=0.5*(Y(J)-Y(J-2))
      VIS(3) = VISC
```

```
          VIS(4)=va(I,J-1,K,8)
          VISC=VISC+VIS(4)
          VIS(1)=0.25*(VISC+va(I+1,J,K,8)+va(I+1,J-1,K,8))
          VIS(2)=0.25*(VISC+va(I-1,J,K,8)+va(I-1,J-1,K,8))
          VIS(5)=0.25*(VISC+va(I,J,K+1,8)+va(I,J-1,K+1,8))
          VIS(6)=0.25*(VISC+va(I,J,K-1,8)+va(I,J-1,K-1,8))
          VISC=0.5*VISC
            UE=0.5*(UE+va(I+1,J-1,K,1))
            UW=0.5*(UW+va(I,J-1,K,1))
            WT=0.5*(WT+va(I,J-1,K+1,3))
            WB=0.5*(WB+va(I,J-1,K,3))
            AA=0.5*DY/DYN
            VN=AA*VN+(1.-AA)*VS
            AA=0.5*DY/DYS
            VS=AA*va(I,J-1,K,2)+(1.-AA)*VS
        GO TO 30
C
C  ZDIRECTION MOMETUM EQUATION
20          DZT=DZ
           DZ= DZB
           DZB=0.5*(Z(K)-Z(K-2))
           VIS(5)=VISC
           VIS(6)=va(I,J,K-1,8)
           VISC=VISC+VIS(6)
           VIS(1)=0.25*(VISC+va(I+1,J,K,8)+va(I-1,J,K-1,8))
           VIS(2)=0.25*(VISC+va(I-1,J,K,8)+va(I-1,J,K-1,8))
           VIS(3)=0.25*(VISC+va(I,J+1,K,8)+va(I,J+1,K-1,8))
           VIS(4)=0.25*(VISC+va(I,J-1,K,8)+va(I,J-1,K-1,8))
           VISC=0.5*VISC
            UE=0.5*(UE+va(I+1,J,K-1,1))
            UW=0.5*(UW+va(I,J,K-1,1))
            VN=0.5*(VN+va(I,J+1,K-1,2))
            VS=0.5*(VS+va(I,J,K-1,2))
            AA=0.5*DZ/DZT
            WT=AA*WT+(1.-AA)*WB
            AA=0.5*DZ/DZB
            WB=AA*va(I,J,K-1,3)+(1.-AA)*WB
c
30          AREAX=DY*DZ
           AREAY=DZ*DX
           AREAZ=DX*DY
           VOL=AREAZ*DZ
C
C  CONVECTION COEFFICIENTS
       IF(IEQ.ne.4) GO TO 11
C
       A(1)=AREAX*c(I+1,J,K,1)
       A(2)=AREAX*c(I,J,K,1)
       A(3)=AREAY*c(I,J+1,K,2)
       A(4)=AREAY*c(I,J,K,2)
       A(5)=AREAZ*c(I,J,K+1,3)
       A(6)=AREAZ*c(I,J,K,3)
       DO 5 II=1,6
5       IF(A(II).LE.TINY) A(II)=0.
```

```
C
11          F(1)=UE*AREAX
            F(2)=UW*AREAX
            F(3)=VN*AREAY
            F(4)=VS*AREAY
            F(5)=WT*AREAZ
            F(6)=WB*AREAZ
C
C SOURCE COEFFICIENTS
C
            SMP=F(1)-F(2)+F(3)-F(4)+F(5)-F(6)
            SP=0.
            SC= -SMP
      IF(IEQ.EQ.4) GO TO 107
            CP=MAX(0.,SMP)
C           CP=0.
            SP=-CP
      IF(IEQ.EQ.1)SC=CP*VA(I,J,K,1)+AREAX*(VA(I-1,J,K,7)-
     1      VA(I,J,K,7))
     1      +VOL*((VA(I,J,K,8)-VA(I-1,J,K,8))*(VA(I+1,J,K,1)-VA(I-1,J,K,1))
     2      /(2*DX**2)+(VA(I-1,J+1,K,8)+VA(I,J+1,K,8)-VA(I-1,J-1,K,8)-
     1      VA(I,J-1,K,8))*(VA(I,J+1,K,2)+VA(I,J,K,2)-VA(I-1,J+1,K,2)-VA(
     1      I-1,J,K,2))/(8*DX*DY)+(VA(I-1,J,K+1,8)+VA(I,J,K+1,8)-
     3      VA(I-1,J,K-1,8)-VA(I,J,K-1,8))*(VA(I,J,K+1,3)+VA(I,J,K,3)
     1      -VA(I-1,J,K+1,3)-VA(I-1,J,K,3))/(8*DZ*DX))
      IF(IEQ.EQ.2)SC=CP*VA(I,J,K,2)+AREAY*(VA(I,J-1,K,7)-
     1      VA(I,J,K,7))
     1+VOL*((VA(I+1,J-1,K,8)+VA(I+1,J,K,8)-VA(I-1,J-1,K,8)-VA(I-1,
     2J,K,8))*(VA(I,J,K,1)+VA(I+1,J,K,1)-VA(I,J-1,K,1)-VA(I+1,J-1,K,1))
     1/(8*DX*DY)+(VA(I,J,K,8)-VA(I,J-1,K,8))*(VA(I,J+1,K,2)-VA(I,J-1,K,
     2      2))/(2*DY**2)+(VA(I,J,K+1,8)+VA(I,J-1,K+1,8)-VA(I,J,K-1,8)-VA(I,
     1J-1,K-1,8))*(VA(I,J,K,3)+VA(I,J,K+1,3)-VA(I,J-1,K,3)-VA(I,J-1,K+1
     2      ,3))/(8*DY*DZ))
      IF(IEQ.EQ.3)SC=CP*VA(I,J,K,3)+AREAZ*(
     1      VA(I,J,K-1,7)-VA(I,J,K,7))
     1+VOL*((VA(I+1,J,K,8)+VA(I+1,J,K-1,8)-VA(I-1,J,K,8)-VA(I-1,J,K-1,
     1      8))*(VA(I,J,K,1)+VA(I+1,J,K,1)-VA(I,J,K-1,1)-VA(I+1,J,K-1,1))/
     2      (8*DZ*DX)+(VA(I,J+1,K-1,8)+VA(I,J+1,K,8)-VA(I,J-1,K-1,8)-
     3      VA(I,J-1,K,8))*(VA(I,J+1,K,2)+VA(I,J,K,2)-VA(I,J+1,K-1,2)-
     1      VA(I,J,K-1,2))/(8*DY*DZ)+(VA(I,J,K,8)-VA(I,J,K-1,8))*(VA(I,J,
     2      K+1,8)-VA(I,J,K-1,3))/(2*DZ**2))
      IF(IEQ.LE.3) GO TO 106
c
C Modification details on the standrad k-C Turbulence Model ....
c
      dudx = (UE-UW)/DX
      dvdy = (VN-VS)/DY
       DWDZ = (WT-WB)/DZ
      DUDY=.25*(VA(I+1,J+1,K,1)+VA(I,J+1,K,1)-VA(I+1,J-1,K,1)
     &   -VA(I,J-1,K,1))/DY
      DUDZ=.25*(VA(I+1,J,K+1,1)+VA(I,J,K+1,1)-VA(I+1,J,K-1,1)
     &   -VA(I,J,K-1,1))/DZ
      DVDX=0.25*(VA(I+1,J+1,K,2)+VA(I+1,J,K,2)-VA(I-1,J+1,K,2)
     &   -VA(I-1,J,K,2))/DX
```

```
      DVDZ=.25*(VA(I,J+1,K+1,2)+VA(I,J,K+1,2)-VA(I,J+1,K-1,2)
     &  -VA(I,J,K-1,2))/DZ
      DWDX=0.25*(VA(I+1,J,K+1,3)+VA(I+1,J,K,3)-VA(I-1,J,K+1,3)
     &  -VA(I-1,J,K,3))/DX
      DWDY=0.25*(VA(I,J+1,K+1,3)+VA(I,J+1,K,3)-VA(I,J-1,K+1,3)
     &  -VA(I,J-1,K,3))/DY
      GEN=2.*(DUDX**2+DVDY**2+DWDZ**2)+
     & (DUDZ+DWDX)**2+(DUDY+DVDX)**2+(DVDZ+DWDY)**2
c
c-- NO changes in cmu
c
        C(I,J,K,4)=0.09
C
      IF(IEQ.EQ.6) GO TO 202
        SC=CP*VA(I,J,K,5)+GEN*VOL*VA(I,J,K,8)
        SP=SP-C(I,J,K,4)*VA(I,J,K,5)*VOL/VA(I,J,K,8)
      GO TO 106
c
202     SC = CP*VA(I,J,K,6)+C1*C(I,J,K,4)*GEN*VA(I,J,K,5)*VOL
        sp = sp-(c2*c(i,j,k,4)*va(i,j,k,5)/va(i,j,k,8))*vol
cc
C DIFFUSION COEFFICIENTS
C
106     D(1)=VIS(1)*AREAX/DXE
        D(2)=VIS(2)*AREAX/DXW
        D(3)=VIS(3)*AREAY/DYN
        D(4)=VIS(4)*AREAY/DYS
        D(5)=VIS(5)*AREAZ/DZT
        D(6)=VIS(6)*AREAZ/DZB
C-- COEFFICIENTS OF VARIABLES
        A(1)=MAX(ABS(.5*F(1)),D(1))-.5*F(1)
        A(2)=MAX(ABS(.5*F(2)),D(2))+.5*F(2)
        A(3)=MAX(ABS(.5*F(3)),D(3))-.5*F(3)
        A(4)=MAX(ABS(.5*F(4)),D(4))+.5*F(4)
        A(5)=MAX(ABS(.5*F(5)),D(5))-.5*F(5)
        A(6)=MAX(ABS(.5*F(6)),D(6))+.5*F(6)
        DO 310 IMM=1,6
        IF(ABS(.5*F(IMM)).GT.D(IMM)) A(IMM)=A(IMM)+D(IMM)
310     CONTINUE
107     CALL BOUNDS(A,SP,SC,I,J,K,IEQ,ICOUNT,Zo)
c Logical Counter is acted for VSL fixing .....
        AP=(A(1)+A(2)+A(3)+A(4)+A(5)+A(6)-SP)/urf(ieq)
        IF(ABS(AP).LT.TINY) AP=1.
        SC=SC+(1.-URF(IEQ))*AP*VA(I,J,K,IEQ)
        IF(IEQ.EQ.1) C(I,J,K,1)=AREAX/AP
        IF(IEQ.EQ.2) C(I,J,K,2)=AREAY/AP
        IF(IEQ.EQ.3) C(I,J,K,3) = AREAZ/AP
C
C ASSEMBLE COEFFICIENTS INTO  TRIDIAGONAL MATRIX

C FOR SWEEP ALONG X DIRECTION

101     IF(isweep.NE.1) GO TO 70
        L=K-K1+1
```

```
          TRIDI(1,L) = AP
          TRIDI(2,L)=A(5)
          TRIDI(3,L)=A(6)
          TRIDI(4,L)=SC+A(1)*va(I+1,J,K,IEQ)+A(2)*va(I-1,J,K,IEQ)+
     &      A(3)*va(I,J+1,K,IEQ)+A(4)*va(I,J-1,K,IEQ)
        GOTO 111
c
C FOR SWEEP ALONG Y DIRECTION
c
70      IF(ISWEEP.ne.2) GO TO 90
          L=I-I1+1
          TRIDI(1,L)=AP
          TRIDI(2,L)=A(1)
          TRIDI(3,L)=A(2)
          TRIDI(4,L)=SC+A(3)*va(I,J+1,K,IEQ)+A(4)*va(I,J-1,K,IEQ)+
     &      A(5)*va(I,J,K+1,IEQ)+A(6)*va(I,J,K-1,IEQ)
        GOTO 111
c
C FOR SWEEP ALONG Z DIRECTION
c
90        L=J-J1+1
          TRIDI(1,L)=AP
          TRIDI(2,L)=A(3)
          TRIDI(3,L)=A(4)
          TRIDI(4,L)=SC+A(1)*va(I+1,J,K,IEQ)+A(2)*va(I-1,J,K,IEQ)+
     &      A(5)*va(I,J,K+1,IEQ)+A(6)*va(I,J,K-1,IEQ)
c
C ERROR COMPUTATION
c
          IF(IEQ.NE.4) GO TO 80
          RES(4)= RES(4)+ABS(SMP)
          GOTO 111
80        TEMP=ABS(TRIDI(4,L)+A(4)*va(I,J-1,K,IEQ)+A(3)*va(I,J+1,K,IEQ)
     &    -AP*VA(I,J,K,IEQ))
          IF(-Sp.GT.HUGE/10.) TEMP=TEMP/HUGE
          RES(IEQ)=RES(IEQ)+TEMP
C
111         RETURN
            END
c-------------------------------------------------------
          SUBROUTINE BOUNDS(A,SP,sc,I,J,K,IEQ,ICOUNT,Zo)
C
           INCLUDE 'FOUR_afer.INC'
           VA(I,J,K,5)=ABS(VA(I,J,K,5))
C
          GO TO (1,2,3,4,5,6) IEQ
C
C .....STREAMWISE VELOCITY ...U  .... U ...U
C++++GROUND AND ROOF
1     IF(K.EQ.2.OR.(K.EQ.lb(9).AND.I.GE.LB(1).AND.I.Le.LB(2).AND.
     & J.GE.LB(5).AND.J.LT.LB(7)).OR.(K.EQ.LB(10).AND.I.GE.LB(3)
     2    .AND.I.LE.LB(4).AND.J.GE.LB(6).AND.J.LT.LB(8))) THEN
C         A(6)=0.0
          DZ=Z(K)-ZZ(2*K-1)
```

```
        CSP=SQRT(SQRT(C(I,J,K,4))*0.5*(va(I,J,K,5)+va(I-1,J,K,5)))
        YPLUSU = CSP*DZ/VISCL
       IF(YPLUSU.LT.YPLUS) TEMP= VISCL/DZ
       IF(YPLUSU.GE.YPLUS) TEMP=CSP*CAPPA/ALOG(EPLUS*YPLUSU)
          if(K.EQ.2)TEMP=CSP*CAPPA/ALOG(DZ/Zo)
          SP=SP-TEMP*AREAZ
        END IF
 C SIDE WALL2
      IF((J.EQ.LB(7).AND.I.GE.LB(1).AND.I.LE.LB(2).AND.K.Lt.LB(9)).OR.
     1 (J.EQ.LB(8).AND.I.GE.LB(3).AND.I.LE.LB(4).AND.K.LT.LB(10)))THEN
C         A(4)=0.0
       CSP = SQRT(SQRT(C(I,J,K,4))*0.5*(va(I,J,K,5)+va(I-1,J,K,5)))
          DY=Y(J)-YY(2*J-1)
       YPLUSU = CSP*DY/VISCL
       IF(YPLUSU.LT.YPLUS)TEMP= VISCL/DY
       IF(YPLUSU.GE.YPLUS)TEMP=CSP*CAPPA/ALOG(EPLUS*YPLUSU)
          SP=SP-TEMP*AREAY
C--       SIDE1
       ELSE IF((J.EQ.LB(5)-1.AND.I.GE.LB(1).AND.I.LE.LB(2).AND.K.LT.
     1    LB(9)).OR.
     1    (J.EQ.LB(6)-1.AND.I.GE.LB(3).AND.I.LE.LB(4).AND.K.LT.LB(10)))
     2    THEN
C         A(3)=0.
          DY=YY(2*J+1)-Y(J)
          CSP=SQRT(SQRT(C(I,J,K,4))*0.5*(VA(I,J,K,5)+VA(I-1,J,K,5)))
          YPLUSU=CSP*DY/VISCL
          IF(YPLUSU.LT.YPLUS)TEMP=VISCL/DY
          IF(YPLUSU.GE.YPLUS)TEMP=CSP*CAPPA/ALOG(EPLUS*YPLUSU)
          SP=SP-TEMP*AREAY
          END IF
          IF((I.GE.LB(1).AND.I.LE.LB(2).AND.J.GE.LB(5).AND.J.LT.LB(7).AND.
     1    K.LT.LB(9)).OR.(I.GE.LB(3).AND.I.LE.LB(4).AND.J.GE.LB(6).AND.J.
     2    LT.LB(8).AND.K.LT.LB(10))) THEN
          SP=-HUGE
          SC=HUGE*VA(I,J,K,1)
       ENDIF
            RETURN
C
C.....CROSS-STREAM VELOCITY....V .... V ....V
C GROUND AND ROOF
2      IF(K.EQ.2.OR.(K.EQ.LB(9).AND.I.GE.LB(1).AND.I.LT.LB(2).
     &  AND.J.GE.LB(5).AND.J.Le.LB(7)).OR.(K.EQ.LB(10).AND.I.GE.
     2    LB(3).AND.I.LT.LB(4).AND.J.GE.LB(6).AND.J.LE.LB(8))) THEN
C         A(6)=0.0
          DZ=Z(K)-ZZ(2*K-1)
         CSP=SQRT(SQRT(C(I,J,K,4))*0.5*(va(I,J,K,5)+va(I,J-1,K,5)))
         YPLUSV = CSP*DZ/VISCL
       IF(YPLUSV.LT.YPLUS)TEMP= VISCL/DZ
       IF(YPLUSV.GE.YPLUS)TEMP=CSP*CAPPA/ALOG(EPLUS*YPLUSV)
          if(k.eq.2)TEMP=CSP*CAPPA/ALOG(DZ/Zo)
          SP=SP-TEMP*AREAZ
      ENDIF
C FRONT  WALL
       IF((I.EQ.LB(1)-1.AND.J.Ge.LB(5).AND.J.LE.LB(7).AND.K.LT.LB(9)).
```

```
     1     OR.(I.EQ.LB(3)-1.AND.J.GE.LB(6).AND.J.LE.LB(8).AND.K.LT.LB(10))
     2     ) THEN
C          A(1)=0.0
           DX=XX(2*I+1)-X(I)
         CSP=SQRT(SQRT(C(I,J,K,4))*0.5*(va(I,J,K,5)+va(I,J-1,K,5)))
         YPLUSV = CSP*DX/VISCL
       IF(YPLUSV.LT.YPLUS) TEMP=VISCL/DX
       IF(YPLUSV.GE.YPLUS)TEMP= CSP*CAPPA/ALOG(EPLUS*YPLUSV)
           SP=SP-TEMP*AREAX
C BACK WALL
        ELSE IF((I.EQ.LB(2).AND.J.Ge.LB(5).AND.J.LE.LB(7).AND.K.LT.
     1     LB(9)).OR.(I.EQ.LB(4).AND.J.GE.LB(6).AND.J.LE.LB(8).AND.
     2     K.LT.LB(10))) THEN
C          A(2)=0.0
           DX=X(I)-XX(2*I-1)
           CSP=SQRT(SQRT(C(I,J,K,4))*0.5*(va(I,J,K,5)+va(I,J-1,K,5)))
           YPLUSV = CSP*DX/VISCL
       IF(YPLUSV.LT.YPLUS) TEMP=VISCL/DX
       IF(YPLUSV.GE.YPLUS)TEMP=CSP*CAPPA/ALOG(EPLUS*YPLUSV)
           SP=SP-TEMP*AREAX
        ENDIF
C  SIDE WALL AND INSIDE BUILDING
        IF((I.GE.LB(1).AND.I.LT.LB(2).AND.J.GE.LB(5).AND.J.LE.LB(7).AND
     1     .K.LT.LB(9)).OR.(I.GE.LB(3).AND.I.LT.LB(4).AND.J.GE.LB(6).AND.
     2     J.LE.LB(8).AND.K.LT.LB(10))) THEN
         SP = -HUGE
         SC = va(I,J,K,2)*HUGE
       ENDIF
            RETURN

C.....VERTICAL VELOCITY....W .....W...W
C FRONT WALL
3      IF((I.EQ.LB(1)-1.AND.J.GE.LB(5).AND.J.LT.LB(7).AND
     1     .K.LE.LB(9))
     1     .OR.(I.EQ.LB(3)-1.AND.J.GE.LB(6).AND.J.LT.LB(8).AND.K.
     2     LE.LB(10))) THEN
C          A(1)=0.0
           DX=XX(2*I+1)-X(I)
         CSP=SQRT(SQRT(C(I,J,K,4))*0.5*(va(I,J,K,5)+va(I,J,K-1,5)))
         YPLUSW = CSP*DX/VISCL
       IF(YPLUSW.LT.YPLUS) TEMP= VISCL/DX
       IF(YPLUSW.GE.YPLUS) TEMP=CSP*CAPPA/ALOG(EPLUS*YPLUSW)
           SP=SP-TEMP*AREAX
C BACK WALL
        ELSE IF((I.EQ.LB(2).AND.J.GE.LB(5).AND.J.LT.LB(7).AND.K.LE.
     1     LB(9)).OR.(I.EQ.LB(4).AND.J.GE.LB(6).AND.J.LT.LB(8)
     2     .AND.K.LE.LB(10))) THEN
C          A(2)=0.0
           DX=X(I)-XX(2*I-1)
         CSP=SQRT(SQRT(C(I,J,K,4))*0.5*(va(I,J,K,5)+va(I,J,K-1,5)))
         YPLUSW = CSP*DX/VISCL
       IF(YPLUSW.LT.YPLUS) TEMP= VISCL/DX
       IF(YPLUSW.GE.YPLUS) TEMP=CSP*CAPPA/ALOG(EPLUS*YPLUSW)
           SP=SP-TEMP*AREAX
```

```
      ENDIF
 C SIDE WALL
        IF((J.EQ.LB(5)-1.AND.I.GE.LB(1).AND.I.LT.LB(2).AND.K.LE.LB(9))
   1    .OR.(J.EQ.LB(6)-1.AND.I.GE.LB(3).AND.I.LT.LB(4).AND.K.LE.LB(10)
   2    )) THEN
 C       A(3)=0.
        DY=YY(2*J+1)-Y(J)
        CSP=SQRT(SQRT(C(I,J,K,4))*0.5*(VA(I,J,K,5)+VA(I,J,K-1,5)))
        YPLUSW=CSP*DY/VISCL
        IF(YPLUSW.LT.YPLUS)TEMP=VISCL/DY
        IF(YPLUSW.GE.YPLUS)TEMP=CSP*CAPPA/ALOG(EPLUS*YPLUSW)
        SP=SP-TEMP*AREAY
        ENDIF
      IF((J.EQ.LB(7).AND.I.GE.LB(1).AND.I.LT.LB(2).AND.K.LE.LB(9))
   1    .OR.(J.EQ.LB(8).AND.I.GE.LB(3).AND.I.LT.LB(4).AND.K.LE.LB(10)
   2    ))THEN
        DY=Y(J)-YY(2*J-1)
      CSP=SQRT(SQRT(C(I,J,K,4))*0.5*(va(I,J,K,5)+va(I,J,K-1,5)))
       YPLUSW = CSP*DY/VISCL
      IF(YPLUSW.LT.YPLUS) TEMP=VISCL/DY
      IF(YPLUSW.GE.YPLUS)TEMP=CSP*CAPPA/ALOG(EPLUS*YPLUSW)
        SP=SP-TEMP*AREAY
      ENDIF
        IF(K.EQ.2.OR.(I.GE.LB(1).AND.I.LT.LB(2).AND.J.GE.LB(5).AND.J.LT.
   1    LB(7).AND.K.LE.LB(9)).OR.(I.GE.LB(3).AND.I.LT.LB(4).AND.J.GE.
   2    LB(6).AND.J.LT.LB(8).AND.K.LE.LB(10))) THEN
        SP=-HUGE
        SC=HUGE*VA(I,J,K,3)
        ENDIF
          return
c .....PRESSURE .....p.....p....p
C  GROUND AND ROOF
4       IF(K.EQ.2.OR.(K.EQ.LB(9).AND.I.GE.LB(1).AND.I.LT.LB(2).
   &  AND.J.GE.LB(5).AND.J.LT.LB(7)).OR.(K.EQ.LB(10).AND.I.GE.
   2    LB(3).AND.I.LT.LB(4).AND.J.GE.LB(6).AND.J.LT.LB(8)
   3    )) THEN
        A(6)=0.
      ENDIF
C  FRONT WALL
      IF((I.EQ.LB(1)-1.AND.J.GE.LB(5).AND.J.LT.LB(7).AND.K.LT.
   1    LB(9)).
   1    OR.(I.EQ.LB(3)-1.AND.J.GE.LB(6).AND.J.LT.LB(8).AND.K.LT.LB(10)
   2    )) THEN
        A(1)=0.
C  BACK WALL
      ELSE IF((I.EQ.LB(2).AND.J.GE.LB(5).AND.J.LT.LB(7).AND.K.LT.
   1    LB(9)).OR.(I.EQ.LB(4).AND.J.GE.LB(6).AND.J.LT.LB(8).AND.K.LT.
   2    LB(10))) THEN
        A(2)=0.
     ENDIF
C SIDE WALL
        IF((J.EQ.LB(5)-1.AND.I.GE.LB(1).AND.I.LT.LB(2).AND.K.LT.LB(9))
   1    .OR.(J.EQ.LB(6)-1.AND.I.GE.LB(3).AND.I.LT.LB(4).AND.K.
   2    LT.LB(10))) THEN
```

```
          A(3)=0.
          ELSE IF((J.EQ.LB(7).AND.I.GE.LB(1).AND.I.LT.LB(2).AND.K.LT.
     1    LB(9)).OR.(J.EQ.LB(8).AND.I.GE.LB(3).AND.I.LT.LB(4).AND.
     2    K.LT.LB(10))) THEN
          A(4)=0.
        ENDIF
          RETURN
C
C..... TURBULENT KINETIC ENERGY.....k .....k ....k
C  GROUND AND ROOF
5       IF(K.EQ.2.OR.(K.EQ.LB(9).AND.I.GE.LB(1).AND.I.LT.LB(2).
     &  AND.J.GE.LB(5).AND.J.LT.LB(7)).OR.(K.EQ.LB(10).AND.I.GE.
     2    LB(3).AND.I.LT.LB(4).AND.J.GE.LB(6).AND.J.LT.LB(8)
     3    )) THEN
        TEMP=va(i,j,K+1,5)*((z(K)-ZZ(2*K-1))/(z(K+1)-ZZ(2*K
     1  -1)))**2
          va(i,j,k,5)=temp
          SP=-HUGE
          SC=HUGE*TEMP
        ENDIF
C  FRONT WALL
        IF((I.EQ.LB(1)-1.AND.J.GE.LB(5).AND.J.LT.LB(7).AND.K.LT.
     1    LB(9)).
     1    OR.(I.EQ.LB(3)-1.AND.J.GE.LB(6).AND.J.LT.LB(8).AND.K.LT.LB(10)
     2    )) THEN
        TEMP=va(i-1,j,K,5)*((X(I)-XX(2*I+1))/(X(I-1)-XX(2*I
     1  +1)))**2
          VA(I,J,K,5)=TEMP
          SP=-HUGE
          SC=HUGE*TEMP
C  BACK WALL
        ELSE IF((I.EQ.LB(2).AND.J.GE.LB(5).AND.J.LT.LB(7).AND.K.LT.
     1    LB(9)).OR.(I.EQ.LB(4).AND.J.GE.LB(6).AND.J.LT.LB(8).AND.K.LT.
     2    LB(10))) THEN
        TEMP=va(i+1,j,K,5)*((X(I)-XX(2*I-1))/(X(I+1)-XX(2*I
     1    -1)))**2
          VA(I,J,K,5)=TEMP
          SP=-HUGE
          SC=HUGE*TEMP
       ENDIF
C SIDE WALL
          IF((J.EQ.LB(5)-1.AND.I.GE.LB(1).AND.I.LT.LB(2).AND.K.LT.LB(9))
     1    .OR.(J.EQ.LB(6)-1.AND.I.GE.LB(3).AND.I.LT.LB(4).AND.K.
     2    LT.LB(10))) THEN
          TEMP=VA(I,J-1,K,5)*((YY(2*J+1)-Y(J))/(YY(2*J+1)-Y(J-1)))
     1    **2
          VA(I,J,K,5)=TEMP
          SP=-HUGE
          SC=HUGE*TEMP
          ELSE IF((J.EQ.LB(7).AND.I.GE.LB(1).AND.I.LT.LB(2).AND.K.LT.
     1    LB(9)).OR.(J.EQ.LB(8).AND.I.GE.LB(3).AND.I.LT.LB(4).AND.
     2    K.LT.LB(10))) THEN
        TEMP= va(i,j+1,K,5)*((Y(J)-YY(2*J-1))/(Y(J+1)-YY(2*J
     1    -1)))**2
```

```
          VA(I,J,K,5)=TEMP
          SP=-HUGE
          SC=HUGE*TEMP
        ENDIF
C  INSIDE BUILDING
      IF((I.GE.LB(1).AND.I.LT.LB(2).AND.J.GE.LB(5).AND.J.LT.LB(7).
     2     AND.
     1     K.LT.LB(9)).OR.(I.GE.LB(3).AND.I.LT.LB(4).AND.J.GE.LB(6).AND.
     2     J.LT.LB(8).AND.K.LT.LB(10))) THEN
        SP=-HUGE
        SC=HUGE*VA(I,J,K,5)
        ENDIF
              RETURN
C
C ..... .....TURBULENT ENERGY DISSIPATION ....c ..... c ...c
C
C  GROUND AND ROOF
6       IF(K.EQ.2.OR.(K.EQ.LB(9).AND.I.GE.LB(1).AND.I.LT.LB(2).
     &  AND.J.GE.LB(5).AND.J.LT.LB(7)).OR.(K.EQ.LB(10).AND.I.GE.
     2     LB(3)
     1     .AND.I.LT.LB(4).AND.J.GE.LB(6).AND.J.LT.LB(8))) THEN
        TEMP=C(I,J,K,4)**0.75*VA(I,J,K,5)**1.5/(CAPPA*
     1     (z(K)-ZZ(2*K-1)))
C         *ALOG(EPLUS*2*(Z(K)-ZZ(2*K-1))*
C    2    C(I,J,K,4)**.25*VA(I,J,K,5)**.5/VISCL)
          VA(I,J,K,6)=TEMP
          SP=-HUGE
          SC=HUGE*TEMP
      ENDIF
C FRONT WALL
      IF((I.EQ.LB(1)-1.AND.J.GE.LB(5).AND.J.LT.LB(7).AND.K.LT.LB(9)
     2     ).OR.(I.EQ.LB(3)-1.AND.J.GE.LB(6).AND.J.LT.LB(8).AND.K.LT.
     1     LB(10))) THEN
          TEMP= C(I,J,K,4)**0.75*va(i,j,K,5)**1.5/
     2     (CAPPA*(XX(2*I+1)-X(I)))
C         *ALOG(EPLUS*2.*(XX(2*I+1)-X(I))*
C    1    C(I,J,K,4)**.25*VA(I,J,K,5)**.5/VISCL)
          VA(I,J,K,6)=TEMP
          SP=-HUGE
          SC=HUGE*TEMP
C BACK WALL
      ELSE IF((I.EQ.LB(2).AND.J.GE.LB(5).AND.J.LT.LB(7).AND.K.LT.
     1     LB(9)).OR.(I.EQ.LB(4).AND.J.GE.LB(6).AND.J.LT.LB(8).AND.K.LT
     2     .LB(10))) THEN
          TEMP=C(I,J,K,4)**0.75*va(i,j,K,5)**1.5/(CAPPA*(X(I)-
     1     XX(2*I-1)))
C         *ALOG(EPLUS*2.*(X(I)-XX(2*I-1))*C(I,J,K,4)**.25*
C    2    VA(I,J,K,5)**.5/VISCL)
          VA(I,J,K,6)=TEMP
          SP=-HUGE
          SC=HUGE*TEMP
      ENDIF
C SIDE WALL
          IF((J.EQ.LB(5)-1.AND.I.GE.LB(1).AND.I.LT.LB(2).AND.K.LT.LB(9))
```

```
     1     .OR.(J.EQ.LB(6)-1.AND.I.GE.LB(3).AND.I.LT.LB(4).AND
     2     .K.LT.LB(10))) THEN
           TEMP=C(I,J,K,4)**0.75*VA(I,J,K,5)**1.5/(CAPPA*(YY(2*J+1)-
     1     Y(J)))
C          *ALOG(EPLUS*2.*(YY(2*J+1)-Y(J))*C(I,J,K,4)**.25*
C    2     VA(I,J,K,5)**.5/VISCL)
           VA(I,J,K,6)=TEMP
           SP=-HUGE
           SC=HUGE*TEMP
      ELSE IF((J.EQ.LB(7).AND.I.GE.LB(1).AND.I.LT.LB(2).AND.K.LT.LB(9
     1     )).OR.(J.EQ.LB(8).AND.I.GE.LB(3).AND.I.LT.LB(4).AND.K.LT.LB(10)
     2     )) THEN
           TEMP=C(I,J,K,4)**.75*va(i,j,K,5)**1.5/(CAPPA*(Y(J)-YY(2*J-1)))
C    1     ALOG(EPLUS*2.*(Y(J)-YY(2*J-1))*C(I,J,K,4)**.25*VA(I,J,K,5)**.5/
C    2     VISCL)
           VA(I,J,K,6)=TEMP
           SP=-HUGE
           SC=HUGE*TEMP
      ENDIF
C INSIDE BUILDING
      IF((I.GE.LB(1).AND.I.LT.LB(2).AND.J.GE.LB(5).AND.J.LT.LB(7).AND.
     1     K.LT.LB(9)).OR.(I.GE.LB(3).AND.I.LT.LB(4).AND.J.GE.LB(6).AND.
     2     J.LT.LB(8).AND.K.LT.LB(10))) THEN
        SP=-HUGE
        SC=HUGE*VA(I,J,K,6)
        ENDIF
                RETURN
                END

C-------------------------------------------
        SUBROUTINE Tms(TRIDI,OV,LEN,TO,T1)
        DIMENSION P(100),Q(100),OV(LEN),TRIDI(4,LEN)
c
C  TRIDIAGONAL MATRIX SOLVER
C
          P(1)=TRIDI(2,1)/TRIDI(1,1)
          Q(1) =(TRIDI(4,1)+TRIDI(3,1)*TO)/TRIDI(1,1)
        DO 10 I=2,LEN
          TEMP=TRIDI(1,I)-TRIDI(3,I)*P(I-1)
          P(I)=TRIDI(2,I)/TEMP
10         Q(I)=(TRIDI(4,I)+TRIDI(3,I)*Q(I-1))/TEMP
          OV(LEN)=Q(LEN)+P(LEN)*T1
        DO 20 I=LEN-1,1,-1
20         OV(I)=P(I)*OV(I+1)+Q(I)
          RETURN
          END
C-------------------------------------------
            SUBROUTINE outf
            INCLUDE 'THREE_afer.INC'
          DO 1 I=1,IMAX
          DO 1 J=1,JMAX
          VA(I,J,1,5)=VA(I,J,2,5)
1         VA(I,J,1,6)=VA(I,J,2,6)*2
      DO 3 K=2,KMAX-1
```

```
          DO 3 J=2,JMAX-1
            va(1,J,K,7)=va(2,J,K,7)
3           va(IMAX,J,K,7)=va(IMAX-1,J,K,7)
        DO 5 I=1,IMAX
          DO 4 K=2,KMAX-1
            va(I,1,K,7)=va(I,2,K,7)
4           va(I,JMAX,K,7)=va(I,JMAX-1,K,7)
          DO 5 J=2,JMAX-1
            va(I,J,1,7)=va(I,J,2,7)
            va(I,J,KMAX,7)=va(I,J,KMAX-1,7)
5         CONTINUE
          DO 2 I=1,IMAX
           va(I,1,1,7)=va(I,2,2,7)
           va(I,JMAX,1,7)=va(I,JMAX-1,2,7)
           va(I,1,KMAX,7)=va(I,2,KMAX-1,7)
2       va(I,JMAX,KMAX,7)=va(I,JMAX-1,KMAX-1,7)
C
C         P/rou = ( P/rou + (2/3)k ) - (2/3)k
C
        DO 6 J=1,JMAX
        DO 6 I=1,IMAX
          DO 6 K=1,KMAX
           VA(I,J,K,7)=VA(I,J,K,7)-(2./3.)*VA(I,J,K,5)
6         CONTINUE
C
C         PO  is the reference pressure
C
          PO=VA(2,JMAX-2,KMAX-2,7)
C
          DO 8 I=1,IMAX
          DO 8 J=1,JMAX
          DO 8 K=1,KMAX
           VA(I,J,K,7)=VA(I,J,K,7)-PO
8         CONTINUE
C
C PRINT MAIN VARIABLES
        WRITE(3) IMAX,JMAX,KMAX
        WRITE(3) (X(I),I=1,IMAX)
        WRITE(3) (Y(J),J=1,JMAX)
        WRITE(3) (Z(K),K=1,KMAX)
C
         DO 20 L=1,8
           IF(L.EQ.4) GO TO 20
           WRITE(3) L
         DO 20 K=1,KMAX
           write(3) k
         DO 20 J=1,JMAX
           write(3) j
          WRITE(3) (va(I,J,K,L),I=1,IMAX)
20          CONTINUE
           CLOSE(3,STATUS='KEEP')
           RETURN
           end
C-------------------------------------------
```

```
C
      SUBROUTINE HELPER(TXT,REPLY)
C  ASK YES OR NO OF TEXT
      CHARACTER*2 ANS
      CHARACTER*40 TXT
      LOGICAL REPLY
      WRITE(*,'(A40)') TXT
      READ(*,'(A1)') ANS
      REPLY=.FALSE.
      IF(ANS.EQ.'Y') REPLY = .TRUE.
           RETURN
           END
c_____________________________________________________________
```

C.2 The included files

one_afer.inc

```
      PARAMETER (VISCL=.133, PRED=1.3, PRTE=1.0 )
      include 'angle3.par'
      common  /varv/ va(IX,IY,IZ,8), vold(IX,IY,IZ,8)
      common  /varc/ C(IX,IY,IZ,4)
      COMMON  /ploc/ XX(2*IX),YY(2*IY),ZZ(2*IZ),IIMAX,JJMAX,KKMAX
      COMMON  /vloc/ X(IX),Y(IY),Z(IZ),imax,jmax,kmax
      common  /othe/ LB(10),URF(8),RES(6),VIS(6)
      dimension alamda(8), alamdam1(8), errn(8), rmsdnp1(8), rmsdn(8),
     1          tempar(8)
      CHARACTER*60 FNAME
```

two_a.inc

```
      include 'angle3.par'
      COMMON /ploc/ XX(2*IX),YY(2*IY),ZZ(2*IZ),IIMAX,JJMAX,KKMAX
      DIMENSION LB(10)
      CHARACTER*60 FNAME
      LOGICAL HELP,OUTPUT
```

three_afer.inc

```
      PARAMETER(C1=1.44,C2=1.92,CAPPA=.4,HUGE=1.E24,TINY=1.E-9)
     PARAMETER(PRTE=1.0,PRED=1.32,SMALL=1.E-3)
     parameter(eplus=9.0,yplus=11.65,viscl=0.133)
      include 'angle3.par'
     common /varv/ va(IX,IY,IZ,8), VOLD(IX,IY,IZ,8)
     common /varc/ c(IX,IY,IZ,4)
     COMMON /vloc/ X(IX),Y(IY),Z(IZ),imax,jmax,kmax
     COMMON /PLOC/ XX(2*IX),YY(2*IY),ZZ(2*IZ),IIMAX,JJMAX,KKMAX
     common /othe/ lb(10),urf(8),res(6),vis(6)
     COMMON /GRID/ VISC,DX,DXE,DXW,DY,DYN,DYS,DZ,DZT,
    1    DZB,UE,UW,VN,VS,WT,WB
     COMMON /AREA/ VOL,AREAX,AREAY,AREAZ
```

```
      COMMON /DU/ DUDY,DUDZ,DVDX,DVDZ,DWDX,DWDY,GEN
      DIMENSION A(6), alamda(8), alamdam1(8), errn(8), rmsdnp1(8),
     1     rmsdn(8), tempar(8)
```

four_afer.inc

```
      PARAMETER(C1=1.44,C2=1.92,CAPPA=.4,HUGE=1.E24,TINY=1.E-9)
      PARAMETER(PRTE=1.0,PRED=1.32,SMALL=1.E-3)
      parameter(eplus=9.0,yplus=11.65,visc=0.133)
          include 'angle3.par'
      common /varv/ va(IX,IY,IZ,8), VOLD(IX,IY,IZ,8)
      common /varc/ c(IX,IY,IZ,4)
      COMMON /vloc/ X(IX),Y(IY),Z(IZ),imax,jmax,kmax
      COMMON /PLOC/ XX(2*IX),YY(2*IY),ZZ(2*IZ),IIMAX,JJMAX,KKMAX
      common /othe/ Ib(10),urf(8),res(6),vis(6)
      COMMON /GRID/ VISC,DX,DXE,DXW,DY,DYN,DYS,DZ,DZT,
     1     DZB,UE,UW,VN,VS,WT,WB
      COMMON /AREA/ VOL,AREAX,AREAY,AREAZ
      COMMON /DU/ DUDY,DUDZ,DVDX,DVDZ,DWDX,DWDY,GEN
      DIMENSION A(6), alamda(8), alamdam1(8), errn(8), rmsdnp1(8),
     1     rmsdn(8), tempar(8)
```

Appendix D

Fortran source code for TWIST --- Turbulent Wind Simulation Technique --- two-layer approach

D.1 The main program

```
Ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
C       This code is a revised version of Twist, which is a code developed at center for building studies in
C       Concordia University by A.Baskaran and Y.S. Zhou, it can be used to calculate the wind flow
C       field around different shapes of building with different wind approaching angles.
C
C       Two-layer approach has been used for the turbulence modeling
C
c<<<<<<<<<<<<<<<<<<  >>>>>>>>>>>>>>> <><><<<<>>>><<>><

c<<<<<<<<<<<<<<<<<<  >>>>>>>>>>>>>>> <><><<<<>>>><<>><
      program twist
      INCLUDE 'FOUR.INC'
        CHARACTER*60 FNAME
      DIMENSION SNORM(6)
      LOGICAL HELP
c
     WRITE(*,*) '!!  *** WELCOME TO TWIST **** !!'

     CALL Module1
c
c Fixing  Convergence Parameters
c
     WRITE(*,*) ' TYPE MAXIT,RESMAX,DIVERG'
        WRITE(*,*)'TYPICAL VALUES 150,0.2,20.'
     READ(*,*) MAXIT,RESMAX,DIVERG
        WRITE(*,*)'THE ROUGHNESS LENGTH FOR GROUND'
        READ(*,*)Zo
     write(*,*) ' Entre the output file name '
     READ(*,'(60a)') FNAME
     OPEN(3,FILE=FNAME,STATUS='NEW',form='unformatted')
     WRITE(*,*) 'Thanks for Waiting .....'
C
C  Main computation
C
10    NITER = NITER+1
     DO 30 IEQ = 1,6
c
C  Reset the Pressure Coreection Terms ...
c
      IF(IEQ.EQ.4) THEN
      DO 20 K=1,KMAX
      DO 20 J=1,JMAX
      DO 20 I=1,IMAX
20      VA(I,J,K,4)=0.0
      END IF
```

```
c
C  Iteration loop Starts .....
c
c
30      CALL MODULE2(IEQ,NITER,Zo)
c
C  FIND TURBULENT VISCOSITY
c
      DO 40 K=3,KMAX-1
      DO 40 J=1,JMAX-1
      DO 40 I=2,IMAX-1
40      VA(I,J,K,8)=(C(I,J,K,4)*VA(I,J,K,5)**2/VA(I,J,K,6)+VISCL)
     1     *URF(8)+
     &   VA(I,J,K,8)*(1.-URF(8))
C--EXTRAPOLATION OF NU T
C       DO 240 K=3,KMAX-1
C       DO 240 J=1,JMAX-1
C       DO 240 I=2,IMAX-1
C      IF(I.GE.LB(2).AND.I.LT.LB(3).AND.J.GE.LB(6).AND.J.LT.LB(7).AND.
C     2   K.LT
C     1    .LB(9)) THEN
C          IF(I.EQ.LB(2))THEN
C          VA(I,J,K,8)=-VA(I-1,J,K,8)
C          ELSE IF(I.EQ.LB(3)-1)THEN
C          VA(I,J,K,8)=-VA(I+1,J,K,8)
C          ENDIF
C          IF(J.EQ.LB(6))THEN
C          VA(I,J,K,8)=-VA(I,J-1,K,8)
C          ELSE IF(I.EQ.LB(7)-1)THEN
C          VA(I,J,K,8)=-VA(I,J+1,K,8)
C          ENDIF
C          IF(K.EQ.LB(9)-1)THEN
C          VA(I,J,K,8)=-VA(I,J,K+1,8)
C          ENDIF
C        ENDIF
C240     CONTINUE
C=================================
C  CALCULATE RESIDUAL ERRORS AND CHECK FOR CONVERGENCE
C
      IF(NITER.NE.1) GO TO 60
      RESID=MAX(RES(1),RES(2),RES(3))
      SNORM(1)=RESID
      SNORM(2)=RESID
      SNORM(3)=RESID
      DO 50 IEQ=4,6
50      SNORM(IEQ) = RES(IEQ)
60      WRITE(*,*) NITER,RES
      DO 70 IEQ=1,6
70      RES(IEQ)=RES(IEQ)/SNORM(IEQ)
      RESID = MAX(RES(1),RES(2),RES(3),RES(4),RES(5),RES(6))
      WRITE(*,*) NITER,RES
      IF(NITER.LT.MAXIT.AND.RESID.GT.RESMAX.AND.RESID.LT.DIVERG)
     &   GO TO 10
      IF(RESID.GE.DIVERG) write(*,*) 'ALGORITHM DIVERGENCE'
```

```
C
C Store the output data in a file .....
c
      CALL OUTF
c
            STOP
            END
C
C
C-------------------------------------------------------
      SUBROUTINE MODULE1
      INCLUDE 'FOUR.INC'
       CHARACTER*60 FNAME
       DIMENSION DIAG(60),BELOW(60),ABOV(60),AKINI(60),EPINI(60),
     1  ANUINI(60),AU(60),RHS(60)
       LOGICAL HELP,OUTPUT,ABORT,L1,L2,L3
     CALL GRIDGN
CCC  PRIMARY INITIALIZATION
        IMAX=IIMAX/2
        JMAX=JJMAX/2
        KMAX=KKMAX/2
        DO 310 I = 1,IMAX
310       X(I)=XX(2*I)
       DO 311 J=1,JMAX
311    Y(J)=YY(2*J)
        DO 312 K=1,KMAX
312    Z(K)=ZZ(2*K)
       DO 320 L=1,8
        DO 320 K=1,KMAX
        DO 320 J=1,JMAX
        DO 320 I=1,IMAX
        if(l.Gt.4) go to 319
        c(i,j,k,l)=0.
        IF(L.EQ.4) C(I,J,K,L)=0.09
319       va(I,J,K,L)=0.
320       continue
c
     WRITE(*,*)' DO YOU WANT : '
     write(*,*)
     write(*,*) '   1)CALCULATE THE INITIAL FLOW FIELD
   &
   &  2)READ THE INITIAL FLOW FIELD '
     WRITE(*,*)
     READ(*,*) ICHO
     GO TO (1,2) ICHO
C
1       WRITE(*,*) ' POWER LAW VELOCITY PROFILE'
       WRITE(*,*)
       WRITE(*,*) 'TYPE ALPHA,ZREF,UREF'
       READ(*,*) ALPHA,ZREF,UREF
        WRITE(*,*)'ANG--Attacking angle'
        READ(*,*)ANG
        ANG=ANG*3.1416/180.
       UREF1=UREF/ZREF**ALPHA
```

```
      DO 610 I=1,IMAX
      DO 610 J=1,JMAX
      DO 610 K=3,KMAX
      UVREF=UREF1*Z(K)**ALPHA
      VA(I,J,K,1)=UVREF*COS(ANG)
      VA(I,J,K,2)=UVREF*SIN(ANG)
      VA(I,J,K,3)=SMALL
610   CONTINUE
      DO 620 I=1,IMAX
      DO 620 J=1,JMAX
      VA(I,J,2,1)=VA(I,J,3,1)*Z(2)/Z(3)
      VA(I,J,2,2)=VA(I,J,3,2)*Z(2)/Z(3)
      VA(I,J,2,3)=SMALL
620   CONTINUE

      WRITE(*,*)'READ IN FACT1 & FACT2,(.4 &.3)'
      READ(*,*)FACT1,FACT2
C
C  FIND THE TURBULENT KINETIC ENERGY AND ITS DISSIPATION
C
      DO 90 I=1,1
      DO 90 J=1,1
      DO 90 K=2,KMAX
      U = VA(I,J,K,1)
      UU = VA(I+1,J,K,1)
      V = VA(I,J,K,2)
      IF(J.NE.JMAX) THEN
       VV = VA(I,J+1,K,2)
      ELSE
       VV = V
      ENDIF
      W = VA(I,J,K,3)
      IF(K.NE.KMAX) THEN
       WW = VA(I,J,K+1,3)
      ELSE
       WW = W
      ENDIF
      USQ = .25*((U+UU)**2 + (V+VV)**2 + (W+WW)**2)
      AKINI(K-1)= .584*USQ/(Z(K)**ALPHA/ALPHA)**2
      EPINI(K-1) = .391*AKINI(K-1)**1.5*Z(K)**(ALPHA-1.)
      ANUINI(K-1)= C(I,J,K,4)*AKINI(K-1)**2/EPINI(K-1)
90    CONTINUE
      AKINI(1)=AKINI(2)*(Z(2)/Z(3))**2
      EPINI(1)=2*0.133*AKINI(1)/Z(2)**2
      ANUINI(1)=C(I,J,2,4)*AKINI(1)**2/EPINI(1)
      NN=0.
111   ACMAX=0.
      ACMAX1=0.
      NN=NN+1
      DIAG(1)=1.
      DIAG(KMAX-1)=1.
      BELOW(1)=0.
      BELOW(KMAX-1)=-1.
      ABOV(1)=-(Z(2)/Z(3))**2
```

```
      ABOV(KMAX-1)=0.
      RHS(1)=0.
      RHS(KMAX-1)=0.
      DO 112 K=2,KMAX-2
      ANU1=0.5*(ANUINI(K+1)+ANUINI(K))/((ZZ(2*K+3)-ZZ(2*K+1))*
     1 (Z(K+2)-Z(K+1)))
      ANU2=0.5*(ANUINI(K)+ANUINI(K-1))/((ZZ(2*K+3)-ZZ(2*K+1))*
     1 (Z(K+1)-Z(K)))
      DIAG(K)=-(ANU1+ANU2)/PRTE-EPINI(K)/AKINI(K)
      BELOW(K)=ANU2/PRTE
      ABOV(K)=ANU1/PRTE
      RHS(K)=-0.09*AKINI(K)**2/EPINI(K)*(ALPHA*UREF1
     1 *Z(K+1)**(ALPHA-1))**2
  112 CONTINUE
      CALL TRID(ABOV,BELOW,DIAG,RHS,AU,KMAX-1)
      DO 102 K=2,KMAX-2
      ACMAX1=AMAX1(ACMAX1,ABS(AU(K)))
      AC=ABS(AU(K)-AKINI(K))
  102 ACMAX=AMAX1(AC,ACMAX)
      DO 103 K=1,KMAX-1
  103 AKINI(K)=FACT1*AU(K)+(1.-FACT1)*AKINI(K)
      DIAG(1)=1.
      DIAG(KMAX-1)=1.
      BELOW(1)=0.
      BELOW(KMAX-1)=-1.
      ABOV(1)=0.
      ABOV(KMAX-1)=0.
      RHS(1)=2*0.133*AKINI(2)/Z(3)**2
      RHS(KMAX-1)=0.
      DO 114 K=2,KMAX-2
      ANU1=0.5*(ANUINI(K+1)+ANUINI(K))/((ZZ(2*K+3)-ZZ(2*K+1))*
     1 (Z(K+2)-Z(K+1)))
      ANU2=0.5*(ANUINI(K)+ANUINI(K-1))/((ZZ(2*K+3)-ZZ(2*K+1))*
     1 (Z(K+1)-Z(K)))
      DIAG(K)=-1.92*EPINI(K)/AKINI(K)-(ANU1+ANU2)/PRED
      BELOW(K)=ANU2/PRED
      ABOV(K)=ANU1/PRED
      RHS(K)= -1.44*0.09*AKINI(K)*(ALPHA*UREF1*Z(K+1)**(ALPHA-1))**2
  114 CONTINUE
      CALL TRID(ABOV,BELOW,DIAG,RHS,AU,KMAX-1)
      DO 105 K=2,KMAX-2
      ACMAX1=AMAX1(ACMAX1,ABS(AU(K)))
      AC=ABS(AU(K)-EPINI(K))
  105 ACMAX=AMAX1(AC,ACMAX)
      ACCC=ACMAX/ACMAX1
      DO 106 K=1,KMAX-1
  106 EPINI(K)=FACT2*ABS(AU(K))+(1.-FACT2)*EPINI(K)
      DO 107 K=1,KMAX-1
  107 ANUINI(K)=0.09*AKINI(K)**2/EPINI(K)
      IF(NN.GE.1000)WRITE(*,*)'CHOOSE ANO SET OF FACT1 &
     1 FACT2,TRY AGAIN'
      IF(ACCC.GE.0.01) GOTO 111
C-- SET INITIAL TURBULENCE PROPERTIES
      DO 212 I=1,IMAX
```

```
          DO 212 J=1,JMAX
          DO 212 K=2,KMAX
          VA(I,J,K,5)=AKINI(K-1)
          VA(I,J,K,6)=EPINI(K-1)
          VA(I,J,K,7)=2./3.*VA(I,J,K,5)
212       VA(I,J,K,8)=ANUINI(K-1)+VISCL
        DO 80 I = LB(2),LB(3)
       DO 80 J = LB(6),LB(7)
       DO 80 K = 2,LB(9)
      IF(J.NE.LB(7).AND.K.NE.LB(9)) va(I,J,K,1) = SMALL
      IF(I.NE.LB(3).AND.K.NE.LB(9))va(I,J,K,2) = SMALL
         IF(I.NE.LB(3).AND.J.NE.LB(7))VA(I,J,K,3)=SMALL
80        CONTINUE
C
C CONTINUITY MUST BE SATISFIED GLOBALLY
            SUM=0.
          DO 110 J=2,JMAX-1
            YJ=.5*(Y(J+1)-Y(J-1))
          DO 110 K=2,KMAX-1
            ZK = .5*(Z(K+1)-Z(K-1))
110           SUM=SUM+(va(IMAX,J,K,1)-va(2,J,K,1))*YJ*ZK
          DO 130 I=2,IMAX-1
            XI=.5*(X(I+1)-X(I-1))
          DO 120 K=2,KMAX-1
            ZK=.5*(Z(K+1)-Z(K-1))
120           SUM=SUM+(va(I,JMAX,K,2)-VA(I,2,K,2))*XI*ZK
          DO 130 J=2,JMAX-1
            YJ=.5*(Y(J+1)-Y(J-1))
130           SUM=SUM+va(I,J,KMAX,3)*XI*YJ
C
C  ADJUST VERTICAL VELOCITY AR TOP OF GRID
        AREAXYZ=(XX(IMAX-1)-XX(3))*(X(IMAX-1)-XX(3))*(YY(JJMAX-1)
   1     -YY(3))
          DO 140 I=1,IMAX
            DELTA=SUM*(X(I)-XX(3))/AREAXYZ
          DO 140 J=1,JMAX
140           va(I,J,KMAX,3)=va(I,J,KMAX,3)-DELTA
C
C  OUTPUT INITIAL CONDITIONS TO FILE IF REQUIRED
C
     CALL HELPER('DO YOU WANT TO STORE INITIAL FLOW FIELD',OUTPUT)
         IF(.NOT.OUTPUT) GO TO 233
C
       WRITE(*,*)'ENTRE THE FILE NAME'
       READ(*,'(60A)') FNAME
       OPEN(2,FILE=FNAME,STATUS='NEW')
       WRITE(2,*)IMAX,JMAX,KMAX
       WRITE(2,'(8F10.3)') (X(I),I=1,IMAX)
       WRITE(2,'(8F10.3)')  (Y(J), J=1,JMAX)
       WRITE(2,'(8F10.3)')  (Z(K),K=1,KMAX)
         DO 200 L=1,8
         IF(L.EQ.4.OR.L.EQ.7) GO TO 200
         WRITE(2,*) l
         DO 190 K=1,KMAX
```

```
          write(2,*) k
          DO 190 J=1,JMAX
          write(2,*) j
          WRITE(2,'(8F10.3)') (VA(I,J,K,L),I=1,IMAX)
190         CONTINUE
200         CONTINUE
          CLOSE(2,STATUS='KEEP')
              go to 233
C
2       WRITE(*,*)' Entre the file name'
        READ(*,'(60A)')  FNAME
        OPEN(2,FILE=FNAME,STATUS='old')
        read(2,*)IMAX,JMAX,KMAX
        read(2,'(8F10.3)') (X(I),I=1,IMAX)
        read(2,'(8F10.3)') (Y(J), J=1,JMAX)
        read(2,'(8F10.3)') (Z(K),K=1,KMAX)
          DO 202 L=1,8
          IF(L.EQ.4.OR.L.EQ.7) GO TO 202
          read(2,*) lpos
          DO 191 K=1,KMAX
          read(2,*) kpos
          DO 191 J=1,JMAX
          read(2,*) jpos
          read(2,'(8F10.3)') (VA(I,J,K,L),I=1,IMAX)
191        CONTINUE
202        CONTINUE
        CLOSE(2,STATUS='KEEP')
C
C  SET UP UNDERRELAXION FACTORS
233         URF(1)=0.9
          URF(2)=.9
          urf(3)=0.9
          URF(4)=1.0
          URF(5)=.45
          URF(6)=.45
          URF(7)=.15
          URF(8)=.7
                RETURN
              END
C-------------------------------
C--       SOLVE TRIDIAGONAL MATRIX
          SUBROUTINE TRID(ABOV,BELO,DIAG,RHS,U,N)
          DIMENSION ABOV(N),BELO(N),DIAG(N),RHS(N),U(N)
          DO 10 I=2,N
          RATIO=BELO(I)/DIAG(I-1)
          DIAG(I)=DIAG(I)-RATIO*ABOV(I-1)
10        RHS(I)=RHS(I)-RATIO*RHS(I-1)
          RHS(N)=RHS(N)/DIAG(N)
          DO 20 I=2,N
          J=N-I+1
20        RHS(J)=(RHS(J)-ABOV(J)*RHS(J+1))/DIAG(J)
          DO 21 I=1,N
21        U(I)=RHS(I)
          RETURN
```

```
      END
C_____________________________________________________________
      SUBROUTINE GRIDGN
      INCLUDE 'FOUR.INC'
      CHARACTER*60 FNAME
      LOGICAL HELP, OUTPUT
      REAL LX
CCCC READ BUILDING PARAMETERS
      WRITE(*,*)'H,THE HEIGHT OF BLDG'
      READ(*,*) H
        WRITE(*,*)'DO YOU WANT TO:'
      write(*,*)  '1) USE the grid generation routine
     &  2) READ the grid locations from a file'
      read(*,*) icho
      go to (1,2)icho
c
1       WRITE(*,*)'NH=?'
        READ(*,*)NH
        WRITE(*,*)'UD,LX,DD,DS1,W(NON 0.),DS2=?'
        READ(*,*)UD,LX,DD,DS1,W,DS2
        WRITE(*,*)'NUD,NLX,NDD,NDS1,NW,NDS2=?'
        READ(*,*)NUD,NLX,NDD,NDS1,NW,NDS2
        WRITE(*,*)'DT,NDT-LENGTH & GRID NO. OVER TOP'
        READ(*,*)DT,NDT
        WRITE(*,*)'THE INITIAL STEP SIZES FOR GRID'
        WRITE(*,*)'(DGF,DGF1,DGB,DGB1,DGS,DGS1,DGG,DGT,DGT1)'
        WRITE(*,*)'DGF,DGT,DGB,DGS -- REMAIN TO BE REFINED'
        READ(*,*) DGF,DGF1,DGB,DGB1,DGS,DGS1,DGG, DGT,DGT1
        WRITE(*,*)'NDGF-FOR FINER GRID NEAR FRONT WALL'
        WRITE(*,*)'NDGT-FOR FINER GRID OVER ROOF TOP'
        WRITE(*,*)'NDGB-FOR FINER GRID NEAR FRONT WALL'
        WRITE(*,*)'NDGS-FOR FINER GRID OVER ROOF TOP'
        READ (*,*)NDGF,NDGT,NDGB,NDGS
        WRITE(*,*)'DGFREF,DGTREF,DGBREF,DGSREF--REFINER GRID
   1    STEP FOR DGF,DGT,DGB,DGS'
        READ (*,*)DGFREF,DGTREF,DGBREF,DGSREF
CCC
C     GRID GENERATING ROUTINE STARTS HERE
CCC
10      IIMAX=2*(NUD+NDGF+NDGB+NLX+NDD)+1
       JJMAX=2*(NDS1+NW+NDS2+2*NDGS)+1
       KKMAX=2*(NH+NDGT+NDT)+3
     XX(2*NUD+1) =-DGF
     XX(2*(NUD+NDGF)+1)=0.
        XX(2*(NUD+NDGF+NLX)+1)=LX
        XX(2*(NUD+NDGF+NLX+NDGB)+1)=LX+DGB
        YY(2*NDS1+1)=-DGS
        YY(2*(NDS1+NDGS)+1)=0.
        YY(2*(NDS1+NDGS+NW)+1)=W
        YY(2*(NDS1+NDGS+NW+NDGS)+1)=W+DGS
      ZZ(3)=0.
        ZZ(2)=-DGG
        ZZ(1)=-2*DGG
       ZZ(2*NH+3)=H
```

```
        ZZ(2*(NH+NDGT)+3)=H+DGT
CCCCc
CCc   computes the building locations
CCCCc
        LB(1)=NUD+1
        LB(2)=NUD+1+NDGF
        LB(3)=LB(2)+NLX
        LB(4)=LB(3)+NDGB
        LB(5)=NDS1+1
        LB(6)=LB(5)+NDGS
        LB(7)=LB(6)+NW
        LB(8)=LB(7)+NDGS
        LB(9)=NH+2
        LB(10)=NH+NDGT+2
C---------
c for X direction
C---------
        CALL FINDB(NDGF,DGF,DGFREF,B)
        WRITE(*,*)'BDGF=' , B
        CALL SETGRD(XX,IIMAX,2*(NUD+NDGF)+1,2*NUD+1,DGFREF,B,1)
        DGFN=XX(2*NUD+2)-XX(2*NUD+1)
        CALL FINDB(NUD,UD,DGFN,B)
        WRITE(*,*)'BUD', B
    CALL SETGRD(XX,iimax,2*NUD+1,1,DGFN,B,1)
        IF((NLX/2)*2.EQ.NLX)THEN
        NLX11=NLX/2
        NLX12=NLX/2
        GOTO 112
        ENDIF
        NLX11=(NLX+1)/2
        NLX12=NLX/2
112      CALL FINDB(NLX11,LX/2,DGF1,B)
        WRITE(*,*)'BLX1=',B
    CALL SETGRD(XX,iimax,2*LB(2)-1,2*LB(2)-1+2*NLX11,DGF1,B,2)
        CALL FINDB(NLX12,LX/2,DGB1,B)
        WRITE(*,*)'BLX2=', B
    CALL SETGRD(XX,iimax,2*LB(3)-1,2*LB(2)-1+2*NLX11,DGB1,B,1)
        CALL FINDB(NDGB,DGB,DGBREF,B)
        WRITE(*,*)'BDGB=' , B
        CALL SETGRD(XX,IIMAX,2*LB(3)-1,2*LB(4)-1,DGBREF,B,2)
        DGBN=XX(2*LB(4)-1)-XX(2*LB(4)-2)
        CALL FINDB(NDD,DD,DGBN,B)
        WRITE(*,*)'BDD=',B
        CALL SETGRD(XX,IIMAX,2*LB(4)-1,IIMAX,DGBN,B,2)
CCC----------
Cc for Y direction-----
CCC----------
        IF((NW/2)*2.EQ.NW) THEN
        NW21=NW/2
        NW22=NW21
        GOTO 122
        ENDIF
        NW21=(NW+1)/2
        NW22=NW/2
```

```
122     CALL FINDB(NW21,W/2,DGS1,B)
        CALL SETGRD(YY,JJMAX,2*LB(6)-1,2*LB(6)-1+2*NW21,DGS1,B,2)
        CALL FINDB(NW22,W/2,DGS1,B)
        WRITE(*,*)'BW=', B
        CALL SETGRD(YY,JJMAX,2*LB(7)-1,2*LB(6)-1+2*NW21,DGS1,B,1)
        CALL FINDB(NDGS,DGS,DGSREF,B)
        WRITE(*,*)'BDGS=' , B
        CALL SETGRD(YY,JJMAX,2*LB(6)-1,2*LB(5)-1,DGSREF,B,1)
        DGSN=YY(2*LB(5))-YY(2*LB(5)-1)
        CALL FINDB(NDS1,DS1,DGSN,B)
        WRITE(*,*)'BDS1=', B
    CALL SETGRD(YY,jjmax,2*NDS1+1,1,DGSN,B,1)
        CALL FINDB(NDGS,DGS,DGSREF,B)
        WRITE(*,*)'BDGS=' , B
        CALL SETGRD(YY,JJMAX,2*LB(7)-1,2*LB(8)-1,DGSREF,B,2)
        DGSN=YY(2*LB(8)-1)-YY(2*LB(8)-2)
        CALL FINDB(NDS2,DS2,DGSN,B)
        WRITE(*,*)'BDS2=', B
        CALL SETGRD(YY,JJMAX,2*LB(8)-1,JJMAX,DGSN,B,2)
CCC
c  for Z direction
CCC
        IF((NH/2)*2.EQ.NH)THEN
        NHZ11=NH/2
        NHZ12=NHZ11
        GOTO 116
        ENDIF
        NHZ11=(NH+1)/2
        NHZ12=NH/2
116     CALL FINDB(NHZ11,H*DGG/(DGG+DGT1),DGG,B)
        CALL SETGRD(ZZ,KKMAX,3,2*NHZ11+3,DGG,B,2)
        CALL FINDB(NHZ12,H*DGT1/(DGG+DGT1),DGT1,B)
        WRITE(*,*)'BH2='
        WRITE(*,*)B
        CALL SETGRD(ZZ,KKMAX,2*NH+3,2*NHZ11+3,DGT1,B,1)
        CALL FINDB(NDGT,DGT,DGTREF,B)
        WRITE(*,*)'BDGT=',B
        CALL SETGRD(ZZ,KKMAX,2*LB(9)-1,2*LB(10)-1,DGTREF,B,2)
        DGTN=ZZ(2*LB(10)-1)-ZZ(2*LB(10)-2)
        CALL FINDB(NDT,DT,DGTN,B)
        WRITE(*,*)'BDT=',B
        CALL SETGRD(ZZ,KKMAX,KKMAX-2*NDT,KKMAX,DGTN,B,2)
CCc
C  OUTPUT TO FILE IF REQUIRED
CCC
      CALL HELPER(' DO YOU WANT STORE GRID DETAILS ?',OUTPUT )
       IF(.NOT.OUTPUT) return
       WRITE(*,*) ' TYPE THE FILE NAME '
       READ(*,'(60a)') FNAME
       OPEN(1,FILE =FNAME,STATUS='NEW')
       WRITE(1,*) IIMAX,JJMAX,KKMAX
       WRITE(1,'(8F10.3)') (XX(I),I=1,IIMAX)
       WRITE(1,'(8F10.3)') (YY(J),J=1,JJMAX)
       WRITE(1,'(8F10.3)') (ZZ(K),K=1,KKMAX)
```

```
      write(1,'(9I4)')  (LB(I),I=1,10)
      CLOSE(1,STATUS='KEEP')
      return
2     write(*,*) 'entre the grid location file'
      read(*,'(60a)') fname
      OPEN(1,FILE =FNAME,STATUS='old')
      read(1,*) IIMAX,JJMAX,KKMAX
      read(1,'(8F10.3)') (XX(I),I=1,IIMAX)
      read(1,'(8F10.3)') (YY(j),j=1,JJMAX)
      read(1,'(8F10.3)') (ZZ(k),k=1,KKMAX)
      READ(1,*) (LB(I),I=1,10)
      CLOSE(1,STATUS='KEEP')
        RETURN
        END
C
C-----------------------------------------------

      SUBROUTINE SETGRD (X,NX,N1,N2,D,B,isweep)
C
C   SET PART OF THE COORDIATE VECTOR X
       DIMENSION X(NX)
      IF(N1.EQ.N2) RETURN
       N=N1
       A=D
       IF(isweep.EQ.2) GO TO 20
C
C   WORKING BACKWARDS THROUGH VECTOR
C
10      N=N-2
       X(N+1)=X(N+2)-A
       IF(N.LT.N2) RETURN
       A=A*B
       X(N)=X(N+1)-A
       IF(N.GT.N2) GO TO 10
       RETURN
C
C   WORKING FORWARDS THROUGH VECTOR
C
20      N=N+2
       X(N-1)=X(N-2)+A
       IF(N.GT.N2) RETURN
       A=A*B
       X(N)=X(N-1)+A
       IF(N.LT.N2) GO TO 20
        RETURN
        END
C
C-----------------------------------------------
C
      SUBROUTINE FINDB(NG,G,DO,B)
C
C FINDS EXPANSION FACTOR FROM GEOMETRIC PROGRESSION
      IF(ABS(G).LE.0.00001.OR.NG.LE.0)RETURN
      AL1=G/DO
```

```
          B=1
          J=1
 11       F=1
          J=J+1
          IF(J.GT.200) WRITE(*,*)B,AL1,AL2
          IF(NG.EQ.1) THEN
          B=1
          RETURN
          ENDIF
          DO 20 I=1,NG-1
 20       F=F+B**I
 30       AL2=F*(B+1)
          AL3=0.1*AL1
          IF(ABS(AL2-AL1).GT.AL3.AND.AL2.LT.AL1)B=B+0.02*B
          IF(ABS(AL2-AL1).GT.AL3.AND.AL2.GT.AL1)B=B-0.02*B
          IF(ABS(AL2-AL1).GT.AL3)GOTO 11
           RETURN
           END
C-------------------------------------------------------
       SUBROUTINE MODULE2(IEQ,NITER,Zo)
        INCLUDE 'FOUR.INC'
        DIMENSION tridi(4,100),OV(100)
          DOUBLE PRECISION TRIDI
C
        I1=2
        I2=IMAX-1
        J1=2
        J2=JMAX-1
        K1=2
        K2=KMAX-1
        IF(IEQ.EQ.1) I1=3
        IF(IEQ.EQ.2) J1=3
        IF(IEQ.EQ.3) K1=3
CCCc  ........  ........  ......  ......
C SWEEP IN X DIRECTION
CCCC
          ISWEEP=1
          LEN=K2-K1+1
        DO 20 I=i1,i2
        DO 20 J=J1,J2
        DO 10 K=K1,K2
 10     CALL assembler(IEQ,TRIDI,ISWEEP,LEN,NITER,Zo)
          TO=va(I,J,K1-1,IEQ)
          T1=va(I,J,K2+1,IEQ)
          CALL tms(TRIDI,OV,LEN,TO,T1)
        DO 20 L=1,LEN
 20       va(I,J,K1+L-1,IEQ) = OV(L)
c
C SWEEP IN Y DIRECTION
C
          ISWEEP=2
          LEN=I2-I1+1
        DO 70 J=J1,J2
        DO 70 K=K1,K2
```

```
        DO 60 I=I1,I2
60     CALL assembler(IEQ,TRIDI,ISWEEP,LEN,NITER,Zo)
         TO=va(I1-1,J,K,IEQ)
         T1=va(I2+1,J,K,IEQ)
         CALL Tms(TRIDI,OV,LEN,TO,T1)
        DO 70 L=1,LEN
70       va(I1+L-1,J,K,IEQ)=OV(L)
C
C SWEEP IN Z DIRECTION

        isweep=3
        RES(IEQ)=0.
        LEN=J2-J1+1
      DO 40 K=K1,K2
      DO 40 I=I1,I2
      DO 30 J=J1,J2
30    CALL assembler(IEQ,tridi,isweep,LEN,NITER,Zo)
        TO=va(I,J1-1,K,IEQ)
        T1=va(I,J2+1,K,IEQ)
        CALL Tms(TRIDI,OV,LEN,TO,T1)
      DO 40 L=1,LEN
       va(I,J1+L-1,K,IEQ)=OV(L)
40      continue
C
C
C UPDATE VELOCITIES AND PRESSURE
C
        IF(IEQ.LT.4) RETURN
C
        IF(IEQ.EQ.4) THEN
          DO 50 I=I1,I2
          DO 50 J=J1,J2
          DO 50 K=K1,K2
      IF(I.NE.2) VA(I,J,K,1)=VA(I,J,K,1)+C(I,J,K,1)*(va(I-1,J,K,4)-
     & va(I,J,K,4))
      IF(J.NE.2) VA(I,J,K,2)=VA(I,J,K,2)+C(I,J,K,2)*(va(I,J-1,K,4)-
     & va(I,J,K,4))
      IF(K.NE.2) VA(I,J,K,3)=VA(I,J,K,3)+C(I,J,K,3)*(va(I,J,K-1,4)
     & -va(I,J,K,4))
50     VA(I,J,K,7)=VA(I,J,K,7)+URF(7)*va(I,J,K,4)
        ENDIF
C
       IF(IEQ.EQ.5) THEN
         DO 80 I = LB(2),LB(3)-1
         DO 80 J = LB(6),LB(7)-1
         DO 80 K = 2,LB(9)-1
         va(I,J,K,5) = TINY
80        CONTINUE
         DO 83 I=1,IMAX
         DO 83 J=1,JMAX
         DO 83 K=2,KMAX
         VA(I,J,K,5)=ABS(VA(I,J,K,5))
83       CONTINUE
       ENDIF
```

```
      IF(IEQ.EQ.6) THEN
        DO 8 I = LB(2),LB(3)-1
        DO 8 J = LB(6),LB(7)-1
        DO 8 K = 2,LB(9)-1
        va(I,J,K,6) = TINY
8       CONTINUE
       ENDIF
         RETURN
         END
C
C-------------------------------------------------
      SUBROUTINE assembler(IEQ,TRIDI,ISWEEP,
     & LEN,NITER,Zo)
c-      IMPLICIT DOUBLE PRECISION A
      INCLUDE 'FOUR.INC'
      DIMENSION F(6),D(6),tridi(4,len)
        DOUBLE PRECISION TRIDI
c--define F's
C----     SET MODIFIED COIFFICIENTS
C
C  GENERAL EQUATION FIRST
C
        DXE=X(I+1)-X(I)
        DXW=X(I)-X(I-1)
         DX=XX(2*I+1)-XX(2*I-1)
         DYN=Y(J+1)-Y(J)
          DYS=Y(J)-Y(J-1)
         DY= YY(2*J+1)-YY(2*J-1)
        DZT=Z(K+1)-Z(K)
       DZB=Z(K)-Z(K-1)
      DZ=ZZ(2*K+1)-ZZ(2*K-1)
C
C  X DIRECTION MOMENTUM EQUATION
C
      IF(IEQ.NE.1) GO TO 10
        DXE=XX(2*I+1)-XX(2*I-1)
        DX=X(I)-X(I-1)
        DXW=XX(2*I-1)-XX(2*I-3)
        VIS(1)=VA(I,J,K,8)
        VIS(2)=va(I-1,J,K,8)
        AA=(X(I)-XX(2*I-1))/(X(I)-X(I-1))
        VISC1=(1.0-AA)*VA(I,J,K,8)+AA*VA(I-1,J,K,8)
        VISC2=(1.0-AA)*VA(I,J+1,K,8)+AA*VA(I-1,J+1,K,8)
        BB=(YY(2*J+1)-Y(J))/(Y(J+1)-Y(J))
        VIS(3)=(1.-BB)*VISC1+BB*VISC2
        VISC2=(1.0-AA)*VA(I,J-1,K,8)+AA*VA(I-1,J-1,K,8)
        BB=(Y(J)-YY(2*J-1))/(Y(J)-Y(J-1))
        VIS(4)=(1.-BB)*VISC1+BB*VISC2
        VISC2=(1.0-AA)*VA(I,J,K+1,8)+AA*VA(I-1,J,K+1,8)
        BB=(ZZ(2*K+1)-Z(K))/(Z(K+1)-Z(K))
        VIS(5)=(1.-BB)*VISC1+BB*VISC2
        VISC2=(1.0-AA)*VA(I,J,K-1,8)+AA*VA(I-1,J,K-1,8)
        BB=(Z(K)-ZZ(2*K-1))/(Z(K)-Z(K-1))
        VIS(6)=(1.-BB)*VISC1+BB*VISC2
```

```
          VISC=VISC1
            VN=(1.0-AA)*VA(I,J+1,K,2)+AA*va(I-1,J+1,K,2)
            VS=(1.0-AA)*VA(I,I,K,2)+AA*va(I-1,J,K,2)
            WT=(1.0-AA)*VA(I,J,K+1,3)+AA*va(I-1,J,K+1,3)
            WB=(1.0-AA)*VA(I,J,K,3)+AA*va(I-1,J,K,3)
            AA=(X(I)-XX(2*I-1))/(XX(2*I+1)-XX(2*I-1))
            UE=(1.0-AA)*VA(I,J,K,1)+AA*VA(I+1,J,K,1)
            AA=(XX(2*I-1)-X(I-1))/(XX(2*I-1)-XX(2*I-3))
            UW=(1.0-AA)*VA(I,J,K,1)+AA*va(I-1,J,K,1)
        GO TO 30
C
C Y DIRECTION MOMETUM EQUATION
C
10      IF(IEQ.NE.2) GO TO 20
         DYN=YY(2*J+1)-YY(2*J-1)
         DY=Y(J)-Y(J-1)
         DYS=YY(2*J-1)-YY(2*J-3)
        VIS(3) =VA(I,J,K,8)
        VIS(4)=va(I,J-1,K,8)
          AA=(Y(J)-YY(2*J-1))/(Y(J)-Y(J-1))
           VISC1=(1.0-AA)*VA(I,J,K,8)+AA*VA(I,J-1,K,8)
         VISC2=(1.0-AA)*VA(I+1,J,K,8)+AA*VA(I+1,J-1,K,8)
          BB=(XX(2*I+1)-X(I))/(X(I+1)-X(I))
         VIS(1)=(1.-BB)*VISC1+BB*VISC2
         VISC2=(1.0-AA)*VA(I-1,J,K,8)+AA*VA(I-1,J-1,K,8)
          BB=(X(I)-XX(2*I-1))/(X(I)-X(I-1))
         VIS(2)=(1.-BB)*VISC1+BB*VISC2
         VISC2=(1.0-AA)*VA(I,J,K+1,8)+AA*VA(I,J-1,K+1,8)
          BB=(ZZ(2*K+1)-Z(K))/(Z(K+1)-Z(K))
         VIS(5)=(1.-BB)*VISC1+BB*VISC2
         VISC2=(1.0-AA)*VA(I,J,K-1,8)+AA*VA(I,J-1,K-1,8)
          BB=(Z(K)-ZZ(2*K-1))/(Z(K)-Z(K-1))
         VIS(6)=(1.-BB)*VISC1+BB*VISC2
        VISC=VISC1
         UE=(1.0-AA)*VA(I+1,J,K,1)+AA*va(I+1,J-1,K,1)
         UW=(1.0-AA)*VA(I,J,K,1)+AA*va(I,J-1,K,1)
         WT=(1.0-AA)*VA(I,J,K+1,3)+AA*VA(I,J-1,K+1,3)
         WB=(1.0-AA)*VA(I,J,K,3)+AA*VA(I,J-1,K,3)
         AA=(Y(J)-YY(2*J-1))/(YY(2*J+1)-YY(2*J-1))
         VN=(1.-AA)*VA(I,J,K,2)+AA*VA(I,J+1,K,2)
         AA=(YY(2*J-1)-Y(J-1))/(YY(2*J-1)-YY(2*J-3))
         VS=AA*va(I,J-1,K,2)+(1.-AA)*VA(I,J,K,2)
     GO TO 30
C----------
C       Z-DIRECTION MOMENTUM
20      IF(IEQ.NE.3) GO TO 25
         DZT=ZZ(2*K+1)-ZZ(2*K-1)
         DZ=Z(K)-Z(K-1)
         DZB=ZZ(2*K-1)-ZZ(2*K-3)
        VIS(5) = VA(I,J,K,8)
        VIS(6)=va(I,J,K-1,8)
          AA=(Z(K)-ZZ(2*K-1))/(Z(K)-Z(K-1))
           VISC1=(1.0-AA)*VA(I,J,K,8)+AA*VA(I,J,K-1,8)
         VISC2=(1.0-AA)*VA(I+1,J,K,8)+AA*VA(I+1,J,K-1,8)
```

```
          BB=(XX(2*I+1)-X(I))/(X(I+1)-X(I))
          VIS(1)=(1.-BB)*VISC1+BB*VISC2
          VISC2=(1.0-AA)*VA(I-1,J,K,8)+AA*VA(I-1,J,K-1,8)
          BB=(X(I)-XX(2*I-1))/(X(I)-X(I-1))
          VIS(2)=(1.-BB)*VISC1+BB*VISC2
          VISC2=(1.-AA)*VA(I,J+1,K,8)+AA*VA(I,J+1,K-1,8)
          BB=(YY(2*J+1)-Y(J))/(Y(J+1)-Y(J))
          VIS(3)=(1.-BB)*VISC1+BB*VISC2
          VISC2=(1.-AA)*VA(I,J-1,K,8)+AA*VA(I,J-1,K-1,8)
          BB=(Y(J)-YY(2*J-1))/(Y(J)-Y(J-1))
          VIS(4)=(1.-BB)*VISC1+BB*VISC2
        VISC=VISC1
          UE=(1.-AA)*VA(I+1,J,K,1)+AA*va(I+1,J,K-1,1)
          UW=(1.-AA)*VA(I,J,K,1)+AA*va(I,J,K-1,1)
          VN=(1.-AA)*VA(I,J+1,K,2)+AA*VA(I,J+1,K-1,2)
          VS=(1.-AA)*VA(I,J,K,2)+AA*VA(I,J,K-1,2)
          AA=(Z(K)-ZZ(2*K-1))/(Z(2*K+1)-ZZ(2*K-1))
          WT=(1.-AA)*VA(I,J,K,3)+AA*VA(I,J,K+1,3)
          AA=(ZZ(2*K-1)-Z(K-1))/(ZZ(2*K-1)-ZZ(2*K-3))
          WB=AA*va(I,J,K-1,3)+(1.-AA)*VA(I,J,K,3)
      GO TO 30
C--------FOR IEQ=4,5,6
25      VISC=va(I,J,K,8)
        IF(IEQ.GE.5) THEN
         AA=(XX(2*I+1)-X(I))/(X(I+1)-X(I))
          VIS(1)=(1.-AA)*VISC+AA*VA(I+1,J,K,8)
         AA=(X(I)-XX(2*I-1))/(X(I)-X(I-1))
          VIS(2)=(1.-AA)*VISC+AA*VA(I-1,J,K,8)
         AA=(YY(2*J+1)-Y(J))/(Y(J+1)-Y(J))
         VIS(3)=(1.-AA)*VISC+AA*VA(I,J+1,K,8)
         AA=(Y(J)-YY(2*J-1))/(Y(J)-Y(J-1))
          VIS(4)=(1.-AA)*VISC+AA*VA(I,J-1,K,8)
         AA=(ZZ(2*K+1)-Z(K))/(Z(K+1)-Z(K))
          VIS(5)=(1.-AA)*VISC+AA*VA(I,J,K+1,8)
         AA=(Z(K)-ZZ(2*K-1))/(Z(K)-Z(K-1))
          VIS(6)=(1.-AA)*VISC+AA*VA(I,J,K-1,8)
        IF(IEQ.EQ.5) THEN
        DO 40 L=1,6
40        VIS(L)=VIS(L)/PRTE
          VISC=VISC/PRTE
        ELSE
        DO 50 L=1,6
50        VIS(L)=(VIS(L)-VISCL)/PRED+VISCL
          VISC=(VISC-VISCL)/PRED+VISCL
        END IF
        END IF
C
C
          UE=va(I+1,J,K,1)
          UW=va(I,J,K,1)
          VN=va(I,J+1,K,2)
          VS=va(I,J,K,2)
          WT=va(I,J,K+1,3)
          WB=va(I,J,K,3)
```

```
C
30        AREAX=DY*DZ
          AREAY=DZ*DX
          AREAZ=DX*DY
          VOL=AREAZ*DZ
C
C  CONVECTION COEFFICIENTS FOR PRESSURE CORRECTIONS
        IF(IEQ.ne.4) GO TO 11
C
        A(1)=AREAX*c(I+1,J,K,1)
        A(2)=AREAX*c(I,J,K,1)
        A(3)=AREAY*c(I,J+1,K,2)
        A(4)=AREAY*c(I,J,K,2)
        A(5)=AREAZ*c(I,J,K+1,3)
        A(6)=AREAZ*c(I,J,K,3)
C-----COEFFICIENTS ALL
11          F(1)=UE*AREAX
          F(2)=UW*AREAX
          F(3)=VN*AREAY
          F(4)=VS*AREAY
          F(5)=WT*AREAZ
          F(6)=WB*AREAZ
C
C SOURCE COEFFICIENTS
C
          SMP=F(1)-F(2)+F(3)-F(4)+F(5)-F(6)
          SP=0.
          SC= -SMP
     IF(IEQ.EQ.4) GO TO 107
          CP=MAX(0.,SMP)
C         CP=0.
          SP=-CP
          BADJUST=1.0
     IF(IEQ.EQ.1)SC=CP*VA(I,J,K,1)+AREAX*(VA(I-1,J,K,7)-
    1     VA(I,J,K,7))
    1     +BADJUST*VOL*((VIS(1)-VIS(2))*(VA(I+1,J,K,1)-
    1     VA(I-1,J,K,1))
    2     /(DX*(DXE+DXW))+(VIS(3)-
    1     VIS(4))*(VA(I,J+1,K,2)+VA(I,J,K,2)-VA(I-1,J+1,K,2)-VA(
    1     I-1,J,K,2))/(DX*(DYN+DYS))+(VIS(5)-
    3     VIS(6))*(VA(I,J,K+1,3)+VA(I,J,K,3)
    1     -VA(I-1,J,K+1,3)-VA(I-1,J,K,3))/((DZT+DZB)*DX))
     IF(IEQ.EQ.2)SC=CP*VA(I,J,K,2)+AREAY*(VA(I,J-1,K,7)-
    1     VA(I,J,K,7))
    1+BADJUST*VOL*((VIS(1)-VIS(2)
    2     )*(VA(I,J,K,1)+VA(I+1,J,K,1)-VA(I,J-1,K,1)-VA(I+1,J-1,K,1))
    1     /((DXE+DXW)*DY)+(VIS(3)-VIS(4))*(VA(I,J+1,K,2)-VA(I,J-1,K,
    2     2))/(DY*(DYN+DYS))+(VIS(5)-VIS(6)
    1     )*(VA(I,J,K,3)+VA(I,J,K+1,3)-VA(I,J-1,K,3)-VA(I,J-1,K+1
    2     ,3))/(DY*(DZT+DZB)))
     IF(IEQ.EQ.3)SC=CP*VA(I,J,K,3)+AREAZ*(
    1     VA(I,J,K-1,7)-VA(I,J,K,7))
    1     +BADJUST*VOL*((VIS(1)-VIS(2)
    1     )*(VA(I,J,K,1)+VA(I+1,J,K,1)-VA(I,J,K-1,1)-VA(I+1,J,K-1,1))/
```

```
     2     (DZ*(DXE+DXW))+(VIS(3)-VIS(4)
     3     )*(VA(I,J+1,K,2)+VA(I,J,K,2)-VA(I,J+1,K-1,2)-
     1     VA(I,J,K-1,2))/((DYN+DYS)*DZ)+(VA(I,J,K,8)-VA(I,J,K-1,8))*(
     1     VA(I,J,K+1,3)-VA(I,J,K-1,3))/(DZ*(DZT+DZB)))
      IF(IEQ.LE.3) GO TO 106
c
C Modification details on the standrad k-C Turbulence Model ....
c
        dudx = (UE-UW)/DX
        dvdy = (VN-VS)/DY
         DWDZ = (WT-WB)/DZ
           AA=(XX(2*I+1)-X(I))/DX
          DUDY=(((1.-AA)*VA(I,J+1,K,1)+AA*VA(I+1,J+1,K,1))-((1.-AA)*
     1     VA(I,J-1,K,1)+AA*VA(I+1,J-1,K,1)))/(Y(J+1)-Y(J-1))
          DUDZ=(((1.-AA)*VA(I,J,K+1,1)+AA*VA(I+1,J,K+1,1))-((1.-AA)*
     1     VA(I,J,K-1,1)+AA*VA(I+1,J,K-1,1)))/(Z(K+1)-Z(K-1))
           AA=(YY(2*J+1)-Y(J))/DY
          DVDX=(((1.-AA)*VA(I+1,J,K,2)+AA*VA(I+1,J+1,K,2))-((1.-AA)*
     1     VA(I-1,J,K,2)+AA*VA(I-1,J+1,K,2)))/(X(I+1)-X(I-1))
          DVDZ=(((1.-AA)*VA(I,J,K+1,2)+AA*VA(I,J+1,K+1,2))-((1.-AA)*
     1     VA(I,J,K-1,2)+AA*VA(I,J+1,K-1,2)))/(Z(K+1)-Z(K-1))
           AA=(ZZ(2*K+1)-Z(K))/DZ
          DWDX=(((1.-AA)*VA(I+1,J,K,3)+AA*VA(I+1,J,K+1,3))-((1.-AA)*
     1     VA(I-1,J,K,3)+AA*VA(I-1,J,K+1,3)))/(X(I+1)-X(I-1))
          DWDY=(((1.-AA)*VA(I,J+1,K,3)+AA*VA(I,J+1,K+1,3))-((1.-AA)*
     1     VA(I,J-1,K,3)+AA*VA(I,J-1,K+1,3)))/(Y(J+1)-Y(J-1))
       GEN=2.*(DUDX**2+DVDY**2+DWDZ**2)+
     &  (DUDZ+DWDX)**2+(DUDY+DVDX)**2+(DVDZ+DWDY)**2
c-----------SOURCE TERMS FOR K & EP
        IF(IEQ.EQ.6) GO TO 202
          SC=CP*VA(I,J,K,5)+GEN*VOL*(VA(I,J,K,8)-VISCL)
          SP=SP-VOL*VA(I,J,K,6)/VA(I,J,K,5)
        GO TO 106
c-------------
202       SC = CP*VA(I,J,K,6)+C1*C(I,J,K,4)*GEN*VA(I,J,K,5)*VOL
          sp = sp-(c2*va(i,j,k,6)/va(i,j,k,5))*vol
cc
C DIFFUSION COEFFICIENTS
C
106       D(1)=VIS(1)*AREAX/DXE
          D(2)=VIS(2)*AREAX/DXW
          D(3)=VIS(3)*AREAY/DYN
          D(4)=VIS(4)*AREAY/DYS
          D(5)=VIS(5)*AREAZ/DZT
          D(6)=VIS(6)*AREAZ/DZB
C-- COEFFICIENTS OF VARIABLES
          A(1)=MAX(ABS(.5*F(1)),D(1))-.5*F(1)
          A(2)=MAX(ABS(.5*F(2)),D(2))+.5*F(2)
          A(3)=MAX(ABS(.5*F(3)),D(3))-.5*F(3)
          A(4)=MAX(ABS(.5*F(4)),D(4))+.5*F(4)
          A(5)=MAX(ABS(.5*F(5)),D(5))-.5*F(5)
          A(6)=MAX(ABS(.5*F(6)),D(6))+.5*F(6)
C=================================================
          DO 310 IMM=1,6
```

```
          IF(ABS(.5*F(IMM)).GT.D(IMM))A(IMM)=A(IMM)+D(IMM)
310       CONTINUE
C=============================================================
107       CALL BOUNDS(A,SP,SC,IEQ,ICOUNT,Zo)
c  Logical Counter is acted  for VSL fixing .....
          AP=(A(1)+A(2)+A(3)+A(4)+A(5)+A(6)-SP)/urf(ieq)
C-------------------------------
c          IF(ABS(AP).LT.TINY**2.AND.IEQ.NE.4) AP=TINY
C-------------------------------
          SC=SC+(1.-URF(IEQ))*AP*VA(I,J,K,IEQ)
          IF(IEQ.EQ.1) C(I,J,K,1)=AREAX/AP
          IF(IEQ.EQ.2) C(I,J,K,2)=AREAY/AP
          IF(IEQ.EQ.3) C(I,J,K,3)=AREAZ/AP
C
C ASSEMBLE COEFFICIENTS INTO  TRIDIAGONAL MATRIX

C FOR SWEEP ALONG X DIRECTION

      IF(isweep.NE.1) GO TO 70
          L=K-K1+1
          TRIDI(1,L) = AP
          TRIDI(2,L)=A(5)
          TRIDI(3,L)=A(6)
          TRIDI(4,L)=SC+A(1)*va(I+1,J,K,IEQ)+A(2)*va(I-1,J,K,IEQ)+
     &     A(3)*va(I,J+1,K,IEQ)+A(4)*va(I,J-1,K,IEQ)
        GOTO 111
c
C FOR SWEEP ALONG Y DIRECTION
c
70      IF(ISWEEP.ne.2) GO TO 90
          L=I-I1+1
          TRIDI(1,L)=AP
          TRIDI(2,L)=A(1)
          TRIDI(3,L)=A(2)
          TRIDI(4,L)=SC+A(3)*va(I,J+1,K,IEQ)+A(4)*va(I,J-1,K,IEQ)+
     &     A(5)*va(I,J,K+1,IEQ)+A(6)*va(I,J,K-1,IEQ)
        GOTO 111
c
C FOR SWEEP ALONG Z DIRECTION
c
90        L=J-J1+1
          TRIDI(1,L)=AP
          TRIDI(2,L)=A(3)
          TRIDI(3,L)=A(4)
          TRIDI(4,L)=SC+A(1)*va(I+1,J,K,IEQ)+A(2)*va(I-1,J,K,IEQ)+
     &     A(5)*va(I,J,K+1,IEQ)+A(6)*va(I,J,K-1,IEQ)
c
C ERROR COMPUTATION
c
          IF(IEQ.NE.4) GO TO 80
          RES(4)= RES(4)+ABS(SMP)
          GOTO 111
80        TEMP=ABS(TRIDI(4,L)+A(4)*va(I,J-1,K,IEQ)+A(3)*va(I,J+1,K,IEQ)
     &    -AP*VA(I,J,K,IEQ))
```

```
      IF(-Sp.GT.HUGE/10.) TEMP=TEMP/HUGE
      RES(IEQ)=RES(IEQ)+TEMP
C
111       RETURN
         END
c-------------------------------------------
C-------------------------------------------
      SUBROUTINE BOUNDS(A,SP,sc,IEQ,ICOUNT,Zo)
C
       INCLUDE 'FOUR.INC'
C
      GO TO (1,2,3,4,5,6) IEQ
C
C .....STREAMWISE VELOCITY ...U .... U ...U
C++++GROUND AND ROOF
1     IF(K.EQ.2) THEN
        A(6)=0.0
        DZ1=Z(K)-ZZ(2*K-1)
     CSP=SQRT(SQRT(C(I,J,K,4))*0.5*(va(I,J,K,5)+va(I-1,J,K,5)))
        TEMP=CSP*CAPPA/ALOG((DZ1+Zo)/Zo)
        SP=SP-TEMP*AREAZ
      END IF
C----NON SLIP FOR U ON TOP
      IF(K.EQ.LB(9).AND.I.GE.LB(2).AND.I.LE.LB(3).AND.
   & J.GE.LB(6).AND.J.LT.LB(7))THEN
      A(6)=0.
      DZ1=Z(K)-ZZ(2*K-1)
C     CSP=SQRT(SQRT(C(I,J,K,4))*0.5*(va(I,J,K,5)+va(I-1,J,K,5)))
C     YPLUSU=CSP*DZ1/VISCL
C     IF(YPLUSU.LT.YPLUS)TEMP=VISCL/DZ1
C     IF(YPLUSU.GE.YPLUS)TEMP=CSP*CAPPA/ALOG(EPLUS*YPLUSU)
      TEMP=VISCL/DZ1
      SP=SP-TEMP*AREAZ
      ENDIF
C SIDE WALL2
    IF(J.EQ.LB(7).AND.I.GE.LB(2).AND.I.LE.LB(3).AND.K.LtLB(9))THEN
      A(4)=0.0
C     CSP = SQRT(SQRT(C(I,J,K,4))*0.5*(va(I,J,K,5)+va(I-1,J,K,5)))
      DY1=Y(J)-YY(2*J-1)
C     YPLUSU = CSP*DY1/VISCL
C     IF(YPLUSU.LT.YPLUS)TEMP= VISCL/DY1
C     IF(YPLUSU.GE.YPLUS)TEMP=CSP*CAPPA/ALOG(EPLUS*YPLUSU)
      TEMP=VISCL/DY1
      SP=SP-TEMP*AREAY
C--     SIDE1
    ELSE IF(J.EQ.LB(6)-1.AND.I.GE.LB(2).AND.I.LE.LB(3).AND.K.LT.
   1    LB(9)) THEN
    A(3)=0.
      DY1=YY(2*J+1)-Y(J)
C     CSP=SQRT(SQRT(C(I,J,K,4))*0.5*(VA(I,J,K,5)+VA(I-1,J,K,5)))
C     YPLUSU=CSP*DY1/VISCL
C     IF(YPLUSU.LT.YPLUS)TEMP=VISCL/DY1
C     IF(YPLUSU.GE.YPLUS)TEMP=CSP*CAPPA/ALOG(EPLUS*YPLUSU)
      TEMP=VISCL/DY1
```

```
          SP=SP-TEMP*AREAY
          END IF
C----INSIDE BLDG
          IF(I.GE.LB(2).AND.I.LE.LB(3).AND.J.GE.LB(6).AND.J.LT.LB(7).
     2    AND.
     1    K.LT.LB(9)) THEN
          VA(I,J,K,1)=SMALL
          SP=-HUGE
          SC=HUGE*VA(I,J,K,1)
       ENDIF
            RETURN
C
C.....CROSS-STREAM VELOCITY....V ....  V  ....V
C GROUND AND ROOF
2       IF(K.EQ.2) THEN
          A(6)=0.0
          DZ1=Z(K)-ZZ(2*K-1)
          CSP=SQRT(SQRT(C(I,J,K,4))*0.5*(va(I,J,K,5)+va(I,J-1,K,5)))
          TEMP=CSP*CAPPA/ALOG((DZ1+Zo)/Zo)
          SP=SP-TEMP*AREAZ
       ENDIF
C-----NON-SLIP ON ROOF FOR V
          IF(K.EQ.LB(9).AND.I.GE.LB(2).AND.I.LT.LB(3).
     &  AND.J.GE.LB(6).AND.J.LE.LB(7))THEN
          A(6)=0.
          DZ1=Z(K)-ZZ(2*K-1)
C         CSP=SQRT(SQRT(C(I,J,K,4))*0.5*(va(I,J,K,5)+va(I,J-1,K,5)))
C         YPLUSV=CSP*DZ1/VISCL
C         IF(YPLUSV.LT.YPLUS)TEMP=VISCL/DZ1
C         IF(YPLUSV.GE.YPLUS)TEMP=CSP*CAPPA/ALOG(EPLUS*YPLUSV)
          TEMP=VISCL/DZ1
          SP=SP-TEMP*AREAZ
          ENDIF
C FRONT  WALL
       IF(I.EQ.LB(2)-1.AND.J.GE.LB(6).AND.J.LE.LB(7).AND.K.LT.LB(9))
     2    THEN
          A(1)=0.
          DX1=XX(2*I+1)-X(I)
C        CSP=SQRT(SQRT(C(I,J,K,4))*0.5*(va(I,J,K,5)+va(I,J-1,K,5)))
C         YPLUSV=CSP*DX1/VISCL
C         IF(YPLUSV.LT.YPLUS)TEMP=VISCL/DX1
C         IF(YPLUSV.GE.YPLUS)TEMP=CSP*CAPPA/ALOG(EPLUS*YPLUSV)
          TEMP=VISCL/DX1
          SP=SP-TEMP*AREAX
C BACK WALL
       ELSE IF(I.EQ.LB(3).AND.J.Ge.LB(6).AND.J.LE.LB(7).AND.K.LT.
     1    LB(9)) THEN
          A(2)=0.0
          DX1=X(I)-XX(2*I-1)
C         CSP=SQRT(SQRT(C(I,J,K,4))*0.5*(va(I,J,K,5)+va(I,J-1,K,5)))
C         YPLUSV = CSP*DX1/VISCL
C     IF(YPLUSV.LT.YPLUS) TEMP=VISCL/DX1
C     IF(YPLUSV.GE.YPLUS)TEMP=CSP*CAPPA/ALOG(EPLUS*YPLUSV)
          TEMP=VISCL/DX1
```

```
          SP=SP-TEMP*AREAX
        ENDIF
C  SIDE WALL AND INSIDE BUILDING
       IF(I.GE.LB(2).AND.I.LT.LB(3).AND.J.GE.LB(6).AND.J.LE.LB(7).AND
     1      .K.LT.LB(9)) THEN
          VA(I,J,K,2)=SMALL
         SP = -HUGE
         SC = va(I,J,K,2)*HUGE
       ENDIF
            RETURN
C.....VERTICAL VELOCITY....W .....W...W
C FRONT WALL, NON-SLIP FOR W
3      IF(I.EQ.LB(2)-1.AND.J.GE.LB(6).AND.J.LT.LB(7).AND
     1      .K.LE.LB(9)) THEN
          A(1)=0.
          DX1=XX(2*I+1)-X(I)
C       CSP=SQRT(SQRT(C(I,J,K,4))*0.5*(va(I,J,K,5)+va(I,J,K-1,5)))
C         YPLUSW=CSP*DX1/VISCL
C         IF(YPLUSW.LT.YPLUS)TEMP=VISCL/DX1
C         IF(YPLUSW.GE.YPLUS)TEMP=CSP*CAPPA/ALOG(EPLUS*YPLUSW)
          TEMP=VISCL/DX1
          SP=SP-TEMP*AREAX
C BACK WALL
       ELSE IF(I.EQ.LB(3).AND.J.GE.LB(6).AND.J.LT.LB(7).AND.K.LE.
     1      LB(9)) THEN
          A(2)=0.0
          DX1=X(I)-XX(2*I-1)
C        CSP=SQRT(SQRT(C(I,J,K,4))*0.5*(va(I,J,K,5)+va(I,J,K-1,5)))
C        YPLUSW = CSP*DX1/VISCL
C       IF(YPLUSW.LT.YPLUS) TEMP= VISCL/DX1
C       IF(YPLUSW.GE.YPLUS) TEMP=CSP*CAPPA/ALOG(EPLUS*YPLUSW)
          TEMP=VISCL/DX1
          SP=SP-TEMP*AREAX
       ENDIF
C SIDE WALL
          IF(J.EQ.LB(6)-1.AND.I.GE.LB(2).AND.I.LT.LB(3).AND.K.LE.LB(9))
     1      THEN
          A(3)=0.
          DY1=YY(2*J+1)-Y(J)
C         CSP=SQRT(SQRT(C(I,J,K,4))*0.5*(VA(I,J,K,5)+VA(I,J,K-1,5)))
C         YPLUSW=CSP*DY1/VISCL
C         IF(YPLUSW.LT.YPLUS)TEMP=VISCL/DY1
C         IF(YPLUSW.GE.YPLUS)TEMP=CSP*CAPPA/ALOG(EPLUS*YPLUSW)
          TEMP=VISCL/DY1
          SP=SP-TEMP*AREAY
          ENDIF
      IF(J.EQ.LB(7).AND.I.GE.LB(2).AND.I.LT.LB(3).AND.K.LE.LB(9))THEN
          A(4)=0.
          DY1=Y(J)-YY(2*J-1)
C       CSP=SQRT(SQRT(C(I,J,K,4))*0.5*(va(I,J,K,5)+va(I,J,K-1,5)))
C        YPLUSW = CSP*DY1/VISCL
C       IF(YPLUSW.LT.YPLUS) TEMP=VISCL/DY1
C       IF(YPLUSW.GE.YPLUS)TEMP=CSP*CAPPA/ALOG(EPLUS*YPLUSW)
          TEMP=VISCL/DY1
```

```
          SP=SP-TEMP*AREAY
        ENDIF
C--------W INSIDE BLDG
          IF(K.EQ.2.OR.(I.GE.LB(2).AND.I.LT.LB(3).AND.J.GE.LB(6).AND.J.LT.
     1    LB(7).AND.K.LE.LB(9))) THEN
          VA(I,J,K,3)=SMALL
          SP=-HUGE
          SC=HUGE*VA(I,J,K,3)
          ENDIF
             return
c .....PRESSURE .....p.....p....p
4         RETURN
C
C..... TURBULENT KINETIC ENERGY.....k .....k ....k
C  GROUND AND ROOF
5       IF(K.EQ.2.OR.(K.EQ.LB(9).AND.I.GE.LB(2).AND.I.LT.LB(3).
     &  AND.J.GE.LB(6).AND.J.LT.LB(7))) THEN
        TEMP=va(i,j,K+1,5)*((z(K)-ZZ(2*K-1))/(z(K+1)-ZZ(2*K
     1  -1)))**2
          va(i,j,k,5)=temp
          SP=-HUGE
          SC=HUGE*TEMP
        ENDIF
C  FRONT WALL
        IF(I.EQ.LB(2)-1.AND.J.GE.LB(6).AND.J.LT.LB(7).AND.K.LT.
     1    LB(9)) THEN
        TEMP=va(i-1,j,K,5)*((X(I)-XX(2*I+1))/(X(I-1)-XX(2*I
     1  +1)))**2
          VA(I,J,K,5)=TEMP
          SP=-HUGE
          SC=HUGE*TEMP
C  BACK WALL
        ELSE IF(I.EQ.LB(3).AND.J.GE.LB(6).AND.J.LT.LB(7).AND.K.LT.
     1    LB(9)) THEN
        TEMP=va(i+1,j,K,5)*((X(I)-XX(2*I-1))/(X(I+1)-XX(2*I
     1    -1)))**2
          VA(I,J,K,5)=TEMP
          SP=-HUGE
          SC=HUGE*TEMP
      ENDIF
C SIDE WALL
          IF(J.EQ.LB(6)-1.AND.I.GE.LB(2).AND.I.LT.LB(3).AND.K.LT.LB(9))
     1    THEN
          TEMP=VA(I,J-1,K,5)*((YY(2*J+1)-Y(J))/(YY(2*J+1)-Y(J-1)))
     1    **2
          VA(I,J,K,5)=TEMP
          SP=-HUGE
          SC=HUGE*TEMP
          ELSE IF(J.EQ.LB(7).AND.I.GE.LB(2).AND.I.LT.LB(3).AND.K.LT.
     1    LB(9)) THEN
        TEMP= va(i,j+1,K,5)*((Y(J)-YY(2*J-1))/(Y(J+1)-YY(2*J
     1    -1)))**2
          VA(I,J,K,5)=TEMP
          SP=-HUGE
```

```
          SC=HUGE*TEMP
        ENDIF
C  INSIDE BUILDING
       IF(I.GE.LB(2).AND.I.LT.LB(3).AND.J.GE.LB(6).AND.J.LT.LB(7).AND.
     2     K.LT
     1     .LB(9)) THEN
        SP=-HUGE
        SC=HUGE*VA(I,J,K,5)
        ENDIF
              RETURN
C------  ========
C-------------------
C ..... .....TURBULENT ENERGY DISSIPATION ....c ..... c ...c
C---------------
C  GROUND AND ROOF
6         IF(K.EQ.2) THEN
          TEMP=2*VISCL*VA(I,J,K,5)/(Z(K)-ZZ(2*K-1))**2
          va(i,j,k,6)=temp
          SP=-HUGE
          SC=HUGE*TEMP
        ENDIF
C-- DEFINE N-R COEFFICIENTS
C=========================
C--- ZONE1
C-- FRONT WALL
C=====================
          IF(I.GE.LB(1).AND.I.LT.LB(2).AND.J.GE.LB(6).AND.J.LT.LB(7).
     1    AND.K.GT.2.AND.K.LT.LB(9)) THEN
          IE=LB(2)
          RK=SQRT(VA(I,J,K,5))*(XX(2*IE-1)-X(I))/VISCL
          FMU=(1-EXP(-0.0198*RK))
          ALL=6.41*CAPPA*(XX(2*IE-1)-X(I))
          RL=SQRT(VA(I,J,K,5))*ALL/VISCL
          C(I,J,K,4)=0.085*FMU*(1+13.2/RL)
          VA(I,J,K,6)=VA(I,J,K,5)**1.5/ALL*(1+13.2/RL)
          SP=-HUGE
          SC=HUGE*VA(I,J,K,6)
          ENDIF
C============
C- ZONE2
C---ROOF
C=================
          IF(I.GE.LB(2).AND.I.LT.LB(3).AND.J.GE.LB(6).AND.J.LT.LB(7).
     1    AND.K.GE.LB(9).AND.K.LT.LB(10)) THEN
          KE=LB(9)
          RK=SQRT(VA(I,J,K,5))*(Z(K)-ZZ(2*KE-1))/VISCL
          FMU=1-EXP(-0.0198*RK)
          ALL=6.41*CAPPA*(Z(K)-ZZ(2*KE-1))
          RL=SQRT(VA(I,J,K,5))*ALL/VISCL
          C(I,J,K,4)=0.085*FMU*(1+13.2/RL)
          VA(I,J,K,6)=VA(I,J,K,5)**1.5/ALL*(1+13.2/RL)
          SP=-HUGE
          SC=HUGE*VA(I,J,K,6)
          ENDIF
```

```
C=================
C-- ZONE3
C------- BACK WALL
C=================
      IF(I.GE.LB(3).AND.I.LT.LB(4).AND.J.GE.LB(6).AND.J.LT.LB(7).AND
     1  .K.GT.2.AND.K.LT.LB(9)) THEN
      IE=LB(3)
      RK=SQRT(VA(I,J,K,5))*(X(I)-XX(2*IE-1))/VISCL
      FMU=(1-EXP(-0.0198*RK))
      ALL=6.41*CAPPA*(X(I)-XX(2*IE-1))
      RL=SQRT(VA(I,J,K,5))*ALL/VISCL
      C(I,J,K,4)=0.085*FMU*(1+13.2/RL)
      VA(I,J,K,6)=VA(I,J,K,5)**1.5/ALL*(1+13.2/RL)
      SP=-HUGE
      SC=HUGE*VA(I,J,K,6)
    ENDIF
C=================
C---ZONE 4
C SIDE WALL 1
C=================
      IF(J.GE.LB(5).AND.J.LT.LB(6).AND.I.GE.LB(2).AND.I.LT.LB(3).
     1  AND.K.GT.2.AND.K.LT.LB(9)) THEN
      JE=LB(6)
      RK=SQRT(VA(I,J,K,5))*(YY(2*JE-1)-Y(J))/VISCL
      FMU=(1-EXP(-0.0198*RK))
      ALL=6.41*CAPPA*(YY(2*JE-1)-Y(J))
      RL=SQRT(VA(I,J,K,5))*ALL/VISCL
      C(I,J,K,4)=0.085*FMU*(1+13.2/RL)
      VA(I,J,K,6)=VA(I,J,K,5)**1.5/ALL*(1+13.2/RL)
      SP=-HUGE
      SC=HUGE*VA(I,J,K,6)
C=================
C---ZONE 5
C----SIDE WALL 2
C=================
    ELSE IF(J.LT.LB(8).AND.J.GE.LB(7).AND.I.GE.LB(2).AND.I.LT.LB(3)
     1  .AND.K.GT.2.AND.K.LT.LB(9)) THEN
      JE=LB(7)
      RK=SQRT(VA(I,J,K,5))*(Y(J)-YY(2*JE-1))/VISCL
      FMU=(1-EXP(-0.0198*RK))
      ALL=6.41*CAPPA*(Y(J)-YY(2*JE-1))
      RL=SQRT(VA(I,J,K,5))*ALL/VISCL
      C(I,J,K,4)=0.085*FMU*(1+13.2/RL)
      VA(I,J,K,6)=VA(I,J,K,5)**1.5/ALL*(1+13.2/RL)
      SP=-HUGE
      SC=HUGE*VA(I,J,K,6)
    ENDIF
C=================
C INSIDE BUILDING
    IF(I.GE.LB(2).AND.I.LT.LB(3).AND.J.GE.LB(6).AND.J.LT.LB(7).AND.
     2  K.LT
     1  .LB(9)) THEN
      VA(I,J,K,6)=TINY
      SP=-HUGE
```

```
      SC=HUGE*VA(I,J,K,6)
      ENDIF
        RETURN
             END

C-------------------------------------------
       SUBROUTINE Tms(TRIDI,OV,LEN,TO,T1)
       DIMENSION P(100),Q(100),OV(LEN),TRIDI(4,LEN)
        DOUBLE PRECISION  P,Q,TRIDI,TEMP
c
C  TRIDIAGONAL MATRIX SOLVER
C
        P(1)=TRIDI(2,1)/TRIDI(1,1)
        Q(1) =(TRIDI(4,1)+TRIDI(3,1)*TO)/TRIDI(1,1)
      DO 10 I=2,LEN
        TEMP=TRIDI(1,I)-TRIDI(3,I)*P(I-1)
        P(I)=TRIDI(2,I)/TEMP
c       IF(P(I).EQ.1.0)P(I)=0.999
10       Q(I)=(TRIDI(4,I)+TRIDI(3,I)*Q(I-1))/TEMP
        OV(LEN)=Q(LEN)+P(LEN)*T1
      DO 20 I=LEN-1,1,-1
20       OV(I)=P(I)*OV(I+1)+Q(I)
        RETURN
        END
C-------------------------------------------
          SUBROUTINE outf
          INCLUDE 'THREE.INC'
        DO 1 I=1,IMAX
        DO 1 J=1,JMAX
        VA(I,J,1,5)=VA(I,J,2,5)
1       VA(I,J,1,6)=VA(I,J,2,6)*2
     DO 3 K=2,KMAX-1
        DO 3 J=2,JMAX-1
          va(1,J,K,7)=va(2,J,K,7)
3          va(IMAX,J,K,7)=va(IMAX-1,J,K,7)
     DO 5 I=1,IMAX
        DO 4 K=2,KMAX-1
          va(I,1,K,7)=va(I,2,K,7)
4          va(I,JMAX,K,7)=va(I,JMAX-1,K,7)
        DO 5 J=2,JMAX-1
          va(I,J,1,7)=va(I,J,2,7)
          va(I,J,KMAX,7)=va(I,J,KMAX-1,7)
5       CONTINUE
        DO 2 I=1,IMAX
         va(I,1,1,7)=va(I,2,2,7)
         va(I,JMAX,1,7)=va(I,JMAX-1,2,7)
         va(I,1,KMAX,7)=va(I,2,KMAX-1,7)
2     va(I,JMAX,KMAX,7)=va(I,JMAX-1,KMAX-1,7)
CCCCc
      DO 6 J=1,JMAX
      DO 6 I=1,IMAX
        DO 6 K=1,KMAX
6       VA(I,J,K,7)=VA(I,J,K,7)-(2./3.)*VA(I,J,K,5)
        PO=VA(2,JMAX-2,KMAX-2,7)
```

```
          DO 8 I=1,IMAX
          DO 8 J=1,JMAX
          DO 8 K=1,KMAX
8         VA(I,J,K,7)=VA(I,J,K,7)-PO
C  PRINT MAIN VARIABLES
       WRITE(3) IMAX,JMAX,KMAX
       WRITE(3) (X(I),I=1,IMAX)
       WRITE(3) (Y(J),J=1,JMAX)
       WRITE(3) (Z(K),K=1,KMAX)
C
        DO 20 L=1,8
          IF(L.EQ.4) GO TO 20
          WRITE(3) L
        DO 20 K=1,KMAX
          write(3) k
        DO 20 J=1,JMAX
          write(3) j
         WRITE(3) (va(I,J,K,L),I=1,IMAX)
20         CONTINUE
          CLOSE(3,STATUS='KEEP')
          RETURN
          end
C-------------------------------------------------------
C
       SUBROUTINE HELPER(TXT,REPLY)
C  ASK YES OR NO OF TEXT
      CHARACTER*2 ANS
      CHARACTER*40 TXT
      LOGICAL REPLY
      WRITE(*,'(A40)') TXT
      READ(*,'(A1)') ANS
      REPLY=.FALSE.
      IF(ANS.EQ.'Y') REPLY = .TRUE.
            RETURN
            END
c_____________________________________________________________________
```

D.2 The included files

three.inc

```
      PARAMETER(C1=1.44,C2=1.92,CAPPA=.4,HUGE=1.E17,TINY=1.E-6)
   PARAMETER(PRTE=1.0,PRED=1.32,SMALL=1.E-6)
   parameter(eplus=9.0,yplus=11.65,viscl=0.133)
   common /varv/ va(75,75,75,8)
   common /varc/ c(75,75,75,4)
   COMMON /vloc/ X(75),Y(75),Z(75),imax,jmax,kmax
   COMMON /PLOC/ XX(150),YY(150),ZZ(150),IIMAX,JJMAX,KKMAX
   common /othe/ lb(10),urf(8),res(6),vis(6)
   COMMON /GRID/ DX,DXE,DXW,DY,DYN,DYS,DZ,DZT,
  1    DZB,UE,UW,VN,VS,WT,WB
```

```
      COMMON /AREA/ VOL,AREAX,AREAY,AREAZ
      DIMENSION A(6)
```

four.inc

```
      PARAMETER(C1=1.44,C2=1.92,CAPPA=.41,HUGE=1.E20,TINY=1.E-6)
      PARAMETER(PRTE=1.0,PRED=1.32,SMALL=1.E-6)
      PARAMETER(EPLUS=9.0,yplus=11.65,viscl=0.133)

      REAL VA(75,75,75,8)
      COMMON /VARV/ VA

      REAL C(75,75,75,4)
      COMMON /VARC/ C

      REAL X(75), Y(75), Z(75)
      INTEGER IMAX, JMAX, KMAX
      COMMON /VLOC/ X,Y,Z,IMAX,JMAX,KMAX

      REAL XX(150),YY(150),ZZ(150)
      INTEGER IIMAX,JJMAX,KKMAX
      COMMON /PLOC/ XX,YY,ZZ,IIMAX,JJMAX,KKMAX

      REAL URF(8),RES(6),VIS(6)
      INTEGER LB(10)
      COMMON /OTHE/ LB,URF,RES,VIS

      REAL DX,DXE,DXW,DY,DYN,DYS,DZ,DZT
      REAL DZB,UE,UW,VN,VS,WT,WB
      REAL F2,F1,I,J,K,I1,J1,K1
      COMMON /GRID/ DX,DXE,DXW,DY,DYN,DYS,DZ,DZT,
     1     DZB,UE,UW,VN,VS,WT,WB
     1     ,F2,F1,I,J,K,I1,J1,K1

      REAL*8 VOL,AREAX,AREAY,AREAZ
      COMMON /AREA/ VOL,AREAX,AREAY,AREAZ

      DOUBLE PRECISION A
      DIMENSION A(6)
```