The University of Queensland

The True Direction Equilibrium Flux Method and its Application

By
Matthew Ross Smith

B.Eng.(Mechanical and Space)(Hons)
The University of Queensland, 2001
M.Phil.(Mechanical Engineering)
The University of Queensland, 2004

A thesis submitted for the degree of
Doctor of Philosophy

Principal Supervisor: *Dr. Michael N. Macrossan*
Associate Supervisor: *Dr. Madhat M. Abdel-jawad*

Division of Mechanical Engineering,
,
The University of Queensalnd,
Australia.

June 2008

I hereby declare that this submission is my own work and to the best of my knowledge it contains no material previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any other degree or diploma at UQ or any other educational institution, except where due acknowledgement is made in the thesis. Any contribution made to the research by colleagues, with whom I have worked at UQ or elsewhere, during my candidature, is fully acknowledged.

I also declare that the intellectual content of this thesis is the product of my own work, except to the extent that assistance from others in the project's design and conception or in style, presentation and linguistic expression is acknowledged.

Matthew Ross Smith

Acknowlegdements

I would like to extend my thanks to my primary supervisor Dr. Michael Macrossan and my associate supervisor Dr. Madhat Abdel-Jawad. I will have fond memories of our insightful and entertaining discussions for a long time to come. I especially acknowledge their contribution to the written components of this work: their experience in technical writing and general academia was invaluable. I would also like to extend my thanks to the staff in the Department of Mechanical Engineering, in particular to the members of the Centre for Hypersonics. Finally, I want my wife and daughter to know that none of this would have been possible without them in my life.

This project was supported by an Australian Postgraduate Award (APA) from January 2007 to December 2007. I would also like to acknowledge the financial support offered by the Centre for Hypersonics covering the expense of presenting this work at the 16th Australasian Fluid Mechanics Conference.

Abstract

The True Direction Equilibrium Flux Method (TDEFM) is mathematically derived and applied to various flow problems. Rather than employing the conventional approach of calculating fluxes of mass, momentum and energy in a series of one dimensional fluxes across cell interfaces, TDEFM models the transportation of mass, momentum and energy based on the mechanism used by a direct solver (such as DSMC). The resulting expressions allow fluxes of mass, momentum and energy to be transfered from a specified source volume to a specified destination volume regardless of whether or not these regions share an adjacent interface. The fluxes of mass, momentum and energy calculated by TDEFM are the analytical solution to the free flight phase of a direct simulation when the flow is in thermal equilibrium and flow properties (such as density) are assumed uniform over each cell volume. The TDEFM fluxes are calculated by integrating the Maxwell-Boltzmann equilibrium distribution function over both velocity space and the physical volume of each cell. The primary advantage to this approach is that the TDEFM fluxes are true directional - fluxes of mass, momentum and energy can be transported in their physically correct direction and do not rely upon one dimensional reconstructions for flux calculation.

Direct solvers possess the ability to maintain gradients of density within each cell through simulation particle location. To increase the physical realism of TDEFM the fluxes are reconstructed using linear variations of density. The revised method, named Density TDEFM (DTDEFM), provides results which are closer to a direct solver in the equilibrium limit than conventional TDEFM without significantly increasing the computational expense. For completeness further flux expressions are developed for the inclusion of linearly varying velocity, resulting in Velocity TDEFM (VTDEFM).

The capacity of TDEFM to capture unaligned flows on a regular grid is demonstrated in various one and two dimensional problems. The effects of using true directional fluxes are first demonstrated by testing the two dimensional radial blast wave and implosion problem. These results obtained show that the results obtained using true directional fluxes better capture the radial motion of gas on a regular cartesian grid when compared to other selected first order continuum solvers. The true directional fluxes were then used to simulate various hypersonic flow problems. The development of these true directional fluxes ultimately lead to the creation of FASTWAVE, a tool capable of predicting blast wave behaviour in city environments in a matter of minutes on a standard desktop PC or laptop. Finally, extensions to viscous flow using en route collisions, adaptive mesh refinement and the hybridisation of TDEFM with a BGK solver is discussed.

Publications arising from this research

1) Smith, M.R., Macrossan, M.N. and Abdel-jawad, M.M., 'Effects of Direction Decoupling in flux calculation in Euler Solvers', *Journal of Computational Physics*, In Press, 2008.
2) Smith, M.R., Macrossan, M.N. and Adbdel-Jawad, M.M., 'Two Dimensional Isotropic Mesh Adaptation for the Euler Equations using TDEFM', In *Proceedings of the 16th Australasian Fluid Mechanics Conference*, 2-7th December, 2007, Gold Coast Australia.
3) Smith, M.R., Macrossan, M.N., Abdel-jawad. M.M. and Ferguson, F., 'DSMC in the Euler Limit and its Approximate Kinetic Theory Flux Method', In *Proceedings of the 14th National Computational Fluid Dynamics Conference, Taiwan*, 16-18th August, 2007, Chi-Tou, Taiwan.
4) Macrossan, M.N., Smith, M.R., Metchnik, M. and Pinto, P.A.,'True Direction Equilibrium Flux Method: Applications on Rectangular 2D Meshes', In *Proceedings of the 25th International Symposium on Rarefied Gas Dynamics*, Edited by M.S. Ivanov and A. K. Rebrov, Siberian Branch of the Russian Academey of Sciences : 239-244, 2007.
5) Smith, M.R., 'True Direction equilibrium fluxes with molecular collision considerations', Department of Mechanical Engineering Report, The University of Queensland, December 2007.
6) Smith, M.R., 'Finite region modification to the Equilibrium Flux Method', Department of Mechanical Engineering Report, The University of Queensland, October 2007.
7) Smith, M.R., Macrossan, M.N. and Abdel-jawad, M.M., 'Direction coupled equilibrium fluxes and the influence of source gradients on flux expressions', Department of Mechanical Engineering Report, The University of Queensland, December 2006.
8) Smith, M.R., Macrossan, M.N, Metchnik, M. and Pinto, P.A., 'True Direction Equilibrium Flux Method: Applications on Rectangular 2D Meshes', Department of Mechanical Engineering Report, The University of Queensland, March 2006.

Contents

List of Figures

List of Tables

Nomenclature

a	= Isentropic speed of sound = $\sqrt{\gamma RT}$	(m/s)
c	= Thermal velocity	(m/s)
c_{mp}	= Most probable thermal speed in an equilibrium gas	(m/s)
$f(x)$	= Continuous probability distribution function of x	(Units of x^{-1})
f_{eq}	= Equilibrium probability distribution function	(Units of s/m)
g	= Relative velocity between molecules	(m/s)
C	= Molecular internal (structural) energy	(J/kg)
ζ_{dof}	= Total Number of Degrees of freedom	(-)
ζ_{trans}	= Number of translational degrees of freedom	(-)
C_v	= Specific heat capacity for constant volume process	$(J/kg.K)$
C_p	= Specific heat capacity for constant pressure process	$(J/kg.K)$
γ	= Ratio of specific heats = C_p/C_v	(-)
R	= Specific gas constant	$(J/kg.K)$
Kn	= Knudsen Number	(-)
λ	= Molecular mean free path between collisions	(m)
τ	= Molecular mean free time between collisions	(s)
N	= Number of particles	(-)
$\mathbf{v}$	= Velocity vector = $[v_x, v_y, v_z]$	(m/s)
$\mathbf{x}$	= Position vector = $[x, y, z]$	(m)
$\mathbf{F}$	= Force vector = $[F_x, F_y, F_z]$	(N)
v_i	= Molecular velocity in direction i prior to collision	(m/s)
$v_i^{'}$	= Molecular velocity in direction i after collision	(m/s)
T	= Temperature	(K)
m	= Bulk (mean) molecular velocity	(m/s)
s	= Equilibrium velocity distribution variance = $\sqrt{RT}$	(m/s)
k	= Boltzmann's constant = $1.3806503e-23$	$(m^2kgs^{-2}K^{-1})$
e_{trans}	= Energy held in translational modes	(J/kg)
e_{rot}	= Energy held in rotational modes	(J/kg)
e_{vib}	= Energy held in vibrational modes	(J/kg)
Δx	= Cell width in direction x	(m/s)
Δt	= Time step size	(s)

Chapter 1

Introduction

Bird's Direct Simulation Monte-Carlo method [15] simulates a rarefied flow by following the motion and collisions of a large number of simulator particles as they move through the flow. DSMC in the high collision rate limit has been used as an Euler solver [80, 73, 52] and as the 'continuum' part of a hybrid DSMC/continuum solver. DSMC is generally more robust than a conventional Euler solver but suffers from statistical scatter which requires large amounts of CPU power to reduce to acceptable limits. One reason for DSMC's stability is that the fluxes of mass, momentum and energy are carried by particles which move in the physically correct directions; in any time step fluxes may flow from any cell to any other cell in the computational domain.

In continuum solvers the fluxes are traditionally 'direction decoupled': one dimensional flux calculations are performed in the direction normal to the interface between two cells, and the fluxes are only exchanged with cells that share an interface. For example, on a 2D structured grid the fluxes flow in two coordinate directions and never flow in one time step between cells which are diagonally contiguous (share a vertex in common) but do not have a common interface. Work by Cook [24] and Smith *et. al.* [93, 71, 98] have shown that when the cell structure is not well aligned with the physical structures in the flow, 'direction-decoupled' methods may produce non-physical results such as negative temperatures or densities where strong shocks occur or interact. These solvers may also produce asymmetrical results where symmetrical results are theoretically predicted.

Macrossan *et al.* [70] used the 'Particle Flux Method' to mimic the effect of DSMC as an Euler solver while cutting down greatly on computational effort. Nevertheless some statistical scatter was unavoidable since particles, which were generated statistically from the Maxwell-Boltzmann distribution within each cell, were used to carry the fluxes to other cells. Prior to the free flight phase of the PFM method, these particles were uniformly distributed within each cell. These particles were free to move from their source cell to any surrounding cell, regardless of grid considerations.

Pullin [80] proposed the Equilibrium Flux Method (EFM) in which the fluxes carried by particles having velocities conforming to the local Maxwell-Boltzmann distribution were calculated analytically across cell surfaces for the limit of an infinite number of particles over a very

small time step Δt. EFM eliminates the statistical scatter associated with the previously proposed particle flux methods. The expressions obtained by Pullin's EFM represent the fluxes of mass, momentum and energy through a surface and therefore require the direction decoupling approach to be used in higher dimensions.

Presented is the True Direction Equilibrium Flux Method (TDEFM) which aims to maintain the analytical foundation of EFM while employing the physical mechanism of transport employed by a direct solver such as DSMC. The fluxes of mass, momentum and energy are determined by integration of the local Maxwell-Boltzmann distribution over both velocity space and the physical volume of each cell. This novel approach allows fluxes to be transported from any specified source volume to any specified destination volume. Unlike EFM, flux exchange between cells is not limited to those sharing adjacent interfaces. The fluxes obtained using TDEFM represent the analytical solution to the free flight phase of a direct simulation in the limit of an infinite number of simulation particles for any time step when conditions in each cell are uniform and in thermal equilibrium.

Direct simulation techniques such as DSMC allow variation of density across each cell through the discrete placement of simulation particles. To increase the physical realism of the TDEFM fluxes, a linear variation of density is then applied. This method, named Density TDEFM (DTDEFM), provides results closer to that of DSMC in the high collision rate limit (Pullin's Equilibrium Particle Simulation Method) than the previously implemented TDEFM fluxes while maintaining the physical realism of true directional fluxes. For mathematical completion, new flux expressions are developed for the addition of a linearly varying bulk velocity, named Velocity TDEFM. Both methods have been shown to provide superior results for unaligned flow on regular cartesian grids than selected, first order continuum solvers [98, 95].

The introduction of Adaptive Mesh Refinement (AMR) has allowed complex problems requiring high levels of grid refinement to be solved by concentrating computational effort on important regions. To increase the applicability of TDEFM the method is then employed using a cartesian-based adaptive mesh which can split or combine cells as the simulation runs. Such methods have been used in the past [47, 50, 97] and have been shown to effectively simulate flows over bodies of arbitrary shapes. Preliminary results using TDEFM when target cell sizes are based on the local mean free path are shown to approximately capture viscous flow as simulated by DSMC.

The assumption of thermal equilibrium required by TDEFM for the analytical derivation of the fluxes is invalid for many engineering flows. For example, simulations of hypersonic flow include shocked regions where the flow time is too small for sufficient collisions to reach thermal equilibrium to occur. The use of DSMC or a BGK solver in all regions of such a flow is often computationally prohibitive. The recent emergence of hybrid methods has presented a solution to this problem. Recent efforts by Kolobov *et. al.* [49, 50] propose a Unified Flow Solver (UFS) where a BGK solver is used in regions of thermal non-equilibrium and a kinetic theory based continuum solver (such as EFM) is used in regions of thermal equilibrium. The fluxes used

are direction decoupled, i.e. fluxes are exchanged between regions sharing adjacent interfaces. Presented is a true directional Unified Flow Solver which employs TDEFM in regions of thermal equilibrium and a BGK solver which is used in regions of thermal non-equilibrium. Following Kolobov *et. al.* [49, 50], the method is then applied using adaptive mesh refinement.

The ability of TDEFM to capture unaligned flow on a regular cartesian grid is then applied in FASTWAVE, a rapid blast wave prediction tool for the two dimensional simulation of blast waves in city environments. Conventional commercially available CFD packages are currently unable to be used in a predictive fashion due to the large amount of time required to create a computational grid appropriate to the geometry and flow conditions. FASTWAVE allows the user to obtain a solution in a matter of minutes instead of hours. The software is validated by comparison of simulations by Long and Sharma [60] whom employed DSMC with collision limiting. FASTWAVE is then applied to the simulation of blast waves in various city environments with solid and indestructible geometries.

Chapter 2

Kinetic Theory of Gases

2.1 Introduction and Summary

This chapter provides some important results taken from the Kinetic Theory of gases. Firstly, binary elastic collisions are discussed, followed by the persistence of velocity ratio following collisions. The Boltzmann Equation and its approximate BGK equation are then presented. The internal structure of a molecule and its corresponding degrees of freedom are later introduced. Since this research focuses almost entirely on the equilibrium condition, emphasis is placed on the derivation of the equilibrium distributions of velocity and energy. The mean free path, an important concept in a Kinetic Theory gas is then discussed.

2.2 Binary Elastic Collisions

The laboratory frame of reference velocity of a particle is:

$$v \equiv c + u \tag{2.1}$$

where c is the thermal velocity and u is the bulk velocity of the gas. The bulk velocity is the mean of the molecular velocities ($u = \overline{v}$) with the thermal velocities varying with temperature and molecular mass. Consider two molecules A and B traveling with velocity $\mathbf{v}_A$ and $\mathbf{v}_B$, where $\mathbf{v}_A$ is a vector of velocities $\mathbf{v} = (v_x, v_y, v_z)$ for molecule A. The pre-collision relative velocity between these particles is $\mathbf{g}_{AB} = \mathbf{v}_A - \mathbf{v}_B$. From the center of mass reference frame, these velocities become:

$$\mathbf{v}_{A\to m} = \frac{m_B}{m_A + m_B}\mathbf{g}_{AB} \tag{2.2}$$

$$\mathbf{v}_{B\to m} = \frac{-m_A}{m_A + m_B}\mathbf{g}_{AB} \tag{2.3}$$

where m_A and m_B is the mass of molecules A and B. The center of mass velocity is:

$$\mathbf{v}_m = \frac{m_A\mathbf{v}_A + m_B\mathbf{v}_B}{m_A + m_B} \tag{2.4}$$

The velocities of particles A and B can be rewritten in terms of a reduced mass $\widetilde{m}$ as:

$$\widetilde{m} = \frac{m_A m_B}{m_A + m_B} \tag{2.5}$$

$$\mathbf{v}_A = \frac{\widetilde{m}}{m_A}\mathbf{g}_{AB} \tag{2.6}$$

$$\mathbf{v}_B = -\frac{\widetilde{m}}{m_A}\mathbf{g}_{AB} \tag{2.7}$$

Since the collision is completely elastic, the post collision velocities must be such that the collision conserves energy and momentum. The post collision velocities of particles A and B are:

$$\mathbf{v}_A^{'} = \mathbf{v}_m + \frac{\widetilde{m}}{m_A}\mathbf{g}_{AB}^{'} \tag{2.8}$$

$$\mathbf{v}_B^{'} = \mathbf{v}_m - \frac{\widetilde{m}}{m_B}\mathbf{g}_{AB}^{'} \tag{2.9}$$

$$\tag{2.10}$$

where $\mathbf{v}_A^{'}$ and $\mathbf{v}_B^{'}$ are the post collision velocities and $\mathbf{g}_{AB}^{'}$ is the post collision relative velocity. In order to conserve energy, the post-collision relative velocity $\mathbf{g}_{AB}^{'}$ must be equal to the pre-collision relative velocity $\mathbf{g}_{AB}$. As a result of the collision, the particles will experience a change in trajectory, which may be defined as the deflection angle χ. From the center of mass reference, the deflection is the same for both particles A and B [107]. It is commonly known that for simple hard spheres, all deflection angles are equally likely [45]. Taking advantage of this fact plus using Marsaglia's method to simplify the trigonometric transformations, a simple algorithm for calculating the post-collision velocities of hard spheres is:

1. Select two particles for collision, A and B.

2. Calculate the center of mass velocities $\mathbf{v}_M = (v_{mx}, v_{my}, v_{mz})$

$$v_{mx} = \frac{m_A v_{Ax} + m_B v_{Bx}}{m_A + m_B}$$
$$v_{my} = \frac{m_A v_{Ax} + m_B v_{Bx}}{m_A + m_B}$$
$$v_{mz} = \frac{m_A v_{Ax} + m_B v_{Bx}}{m_A + m_B}$$

3. Calculate the trigonometric transforms f_x, f_y and f_z using Marsalglia's method. Set a dummy variable κ_1 to 10. While κ_1 is larger than 1, calculate:

$$\kappa_2 = 2\mathrm{R_f} - 1$$
$$\kappa_3 = 2\mathrm{R_f} - 1$$
$$\kappa_1 = \kappa_2^2 + \kappa_3^2$$

where R_f is a random fraction (valued from 0 to 1) calculated at each iteration. Using the values of κ_1-κ_3, the trigonometric transforms are:

$$\begin{aligned} \kappa_4 &= 2\sqrt{(1-\kappa_1)} \\ fx &= 1-2\kappa_1 \\ fy &= \kappa_2\kappa_1 \\ fz &= \kappa_3\kappa_4 \end{aligned}$$

4. Calculate the post collision velocities in the center of mass reference frame:

$$\begin{aligned} v'_{A\to m} &= \frac{\widetilde{m}}{m_A} g_{AB} \\ v'_{B\to m} &= \frac{-\widetilde{m}}{m_B} g_{AB} \end{aligned}$$

5. Use the trigonometric transforms to transform these to the global coordinate system. For molecule A, this will be:

$$\begin{aligned} v'_{A\to m,x} &= f_x v'_{A\to m} \\ v'_{A\to m,y} &= f_y v'_{A\to m} \\ v'_{A\to m,z} &= f_z v'_{A\to m} \end{aligned}$$

6. Perform the same transformation for molecule B. The final velocity for particle A in the coordinates x, y, z is:

$$\begin{aligned} v'_{Ax} &= v_{mx} + v'_{A\to m,x} \\ v'_{Ay} &= v_{my} + v'_{A\to m,y} \\ v'_{Az} &= v_{mz} + v'_{A\to m,z} \end{aligned}$$

The final velocity for particle B is calculated in the same way. This routine is only valid for hard sphere collisions where all possible resulting deflection angles are equally likely.

2.3 Persistence of Velocity

For hard sphere particle elastic collisions particles tend to maintain, on average, a fraction of their initial velocity in any given direction. This is a well known fact [20, 45] and quoting Sir James Jeans:

> We shall find that in general a collision does not necessarily reverse the velocity in the original direction of motion, or even reduce it to rest: there is a marked tendency for the original velocity to persist to some extent after collision.

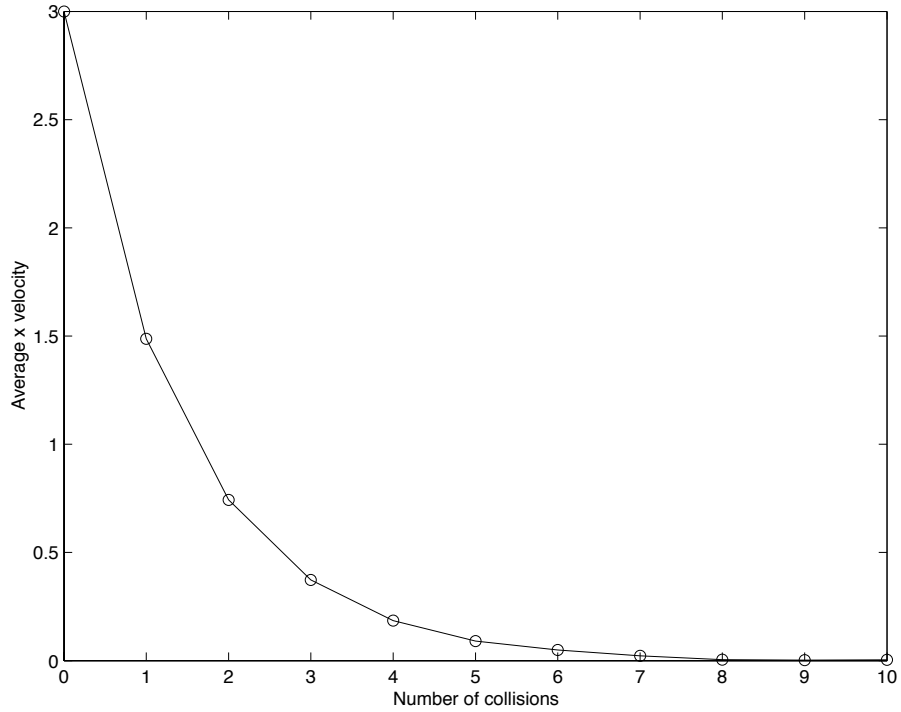

Figure 2.1: Average velocity as a function of collision number. The initial velocity of all particles before collisions was $v_x = 3\sqrt{RT}$. The collision partners are generated from an equilibrium gas at rest with temperature T with the same value of R.

This can be numerically demonstrated to be true by using the collision routine shown in Section 2.2. The procedure for this numerical experiment is:

1. Create n particles, all with an x velocity of $v_{x0} = 3\sqrt{RT}$. Assign each molecule y and z velocities $v_{y0} = v_{z0} = R_N\sqrt{RT}$, where R_N is a normally distributed random number with a mean of 0 and a variance of 1.

2. Each particle k is collided with a randomly generated molecule. This molecule is generated from a gas in equilibrium with a mean of m (For the results presented, $m = 0$) and a variance of RT. The resulting velocity $v_{1,k}$ of the particle is recorded and the generated particle destroyed.

3. The average resulting velocity $\overline{v_{x1}}$ is calculated using $\overline{v_{x1}} = \frac{1}{n}\sum_{k=1}^{n} v_{1,k}$.

4. Each particle k is again collided with a newly randomly generated molecule, this time using the resulting velocity from the previous collision $v_{x0} = v_{1,k}$. The resulting velocity $v_{2,k}$ is recorded for use in the next collision.

5. The average resulting velocity after the second collision $\overline{v}_2$ is calculated.

6. The procedure is repeated for the required number of collisions.

Figure 2.1 shows the mean velocity $\overline{v_x}$ as a function of the number of collisions. After one collision, on average, particles still posses approximately one half of their initial velocity. After

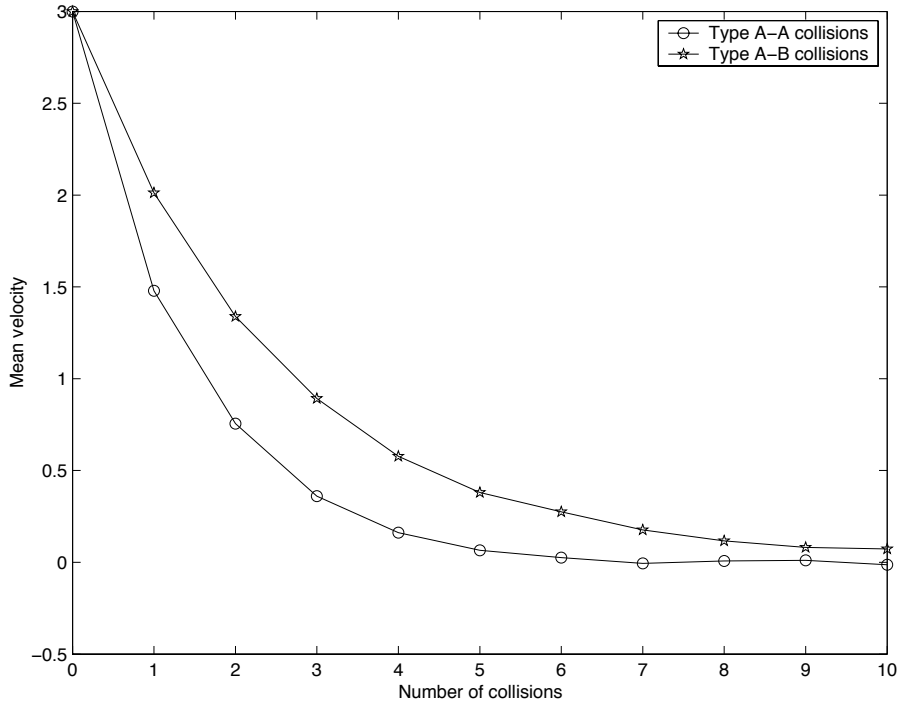

Figure 2.2: Testing the average change in velocity of a type A particle as a function of collisions for particles of different mass. In this instance, type A particles have a mass $m_A = 1$ while type B particles have a mass of $m = 0.5$. 5000 simulation particles were used to find the average post collision velocities.

another collision, the particles average one quarter of their initial velocity. Therefore, for this numerical experiment, the persistence of velocity ratio $\varpi = 1/2$. As the number of collisions increases, the average velocity reduces until the mean value is reached.
The theoretical value of the velocity persistence ratio ϖ is provided by Jeans [45] For any two molecules traveling with a velocity v_A and v_B, the persistence of velocity ratio is:

$$\begin{aligned} \varpi &= \frac{15(\eta^4+1}{10\eta^2(3\eta^2+1)} \qquad [\eta > 1] \\ \varpi &= \frac{3\eta^2+5}{5(\eta^2+3)} \qquad [\eta < 1] \end{aligned} \tag{2.11}$$

where η is the ratio of pre-collision velocities $\eta = v_A/v_B$. In this case, the mass of particles A and B are assumed identical. Depending on the value of η, the persistence of velocity ratio varies from $1/2$ to $1/3$. When the masses of A and B differ we expect the value of ϖ to differ since a heavier particle colliding with a lighter particle will tend to keep more of its pre-collision velocity. This is demonstrated in Figure 2.2. Heavy particles colliding solely with lighter particles posses a larger velocity persistence ratio when when colliding with other heavy particles. Jeans [45] shows that when colliding particles are of different mass, the modified persistence ratio is:

$$\varpi_{AB} = \left(\frac{m_A - m_B}{m_A + m_B}\right) + \left(\frac{2m_B}{m_A + m_B}\right)\varpi_{AA} \tag{2.12}$$

where ϖ_{AB} is the persistence of velocity ratio for collisions of particle type B to particle type B, m_A and m_B is the masses for particle type A and B and ϖ_{AA} is the persistence ratio for collisions of type A particles, shown above to be $\varpi_{AA} = 1/2$.

2.4 Boltzmann Equation

In kinetic theory, the state of a monatomic gas is given by the molecular velocity distribution $f(\mathbf{x}, \mathbf{v}, t)$, where $\mathbf{x}$ is the position vector $\mathbf{x} = (x, y, z)$, $\mathbf{v}$ is the velocity vector $\mathbf{v} = (v_x, v_y, v_z)$ and t is the time. At time t, the number of particles dN in the phase space $dxdydzdv_xdv_ydv_z$, denoted $d\mathbf{x}d\mathbf{v}$, is:

$$dN = n(\mathbf{x}, t)f(\mathbf{x}, \mathbf{v}, t)d\mathbf{x}d\mathbf{v} \tag{2.13}$$

The value of f is expected to evolve through time and space. Neglecting long range particle interactions and limiting molecular interactions to binary collisions, the evolution of f is described by the Boltzmann equation. The Boltzmann equation is:

$$\frac{\partial(nf)}{\partial t} + \mathbf{v}.\frac{\partial(nf)}{\partial \mathbf{x}} + \mathbf{F}.\frac{\partial(nf)}{\partial \mathbf{x}} = \left[\frac{\partial(nf)}{\partial t}\right]_{coll} \tag{2.14}$$

The term $\mathbf{v}.\partial(nf)/\partial\mathbf{x}$ represented the movement of molecules out of spacial element $d\mathbf{x}$ due to molecular velocity $\mathbf{v}$. The term $\mathbf{F}.\partial(nf)/\partial\mathbf{v}$ accounts for acceleration on particles due to an external force $\mathbf{F}$. The collision term (designated by subscript *coll*) accounts for velocity changes due to molecular collisions.

Bhtanagar, Gross and Krook [11] and Weylander [109] developed a simplified form of the collision term shown in equation 2.14. The proposed replacement for the collision term is given by:

$$\left[\frac{\partial(nf)}{\partial t}\right]_{coll} = nv(f_{eq} - f) \tag{2.15}$$

Where v is the collision frequency and f_{eq} is the value of the local equilibrium velocity distribution function. Here the collision frequency is assumed a function of temperature and density but independent of velocity [107].

2.5 Molecular structure and internal degrees of freedom

A molecule is a collection of atoms bound together by a rather complex intramolecuar force. [6] The amount of energy a gas can hold is defined by the way individual molecules can hold energy. These 'modes' are summarised in Figure 2.5, and described below:

- Translational Energy - the kinetic energy molecules posses through movement of the center of mass of each molecule. Since space can be broken up into 3 components - x, y,

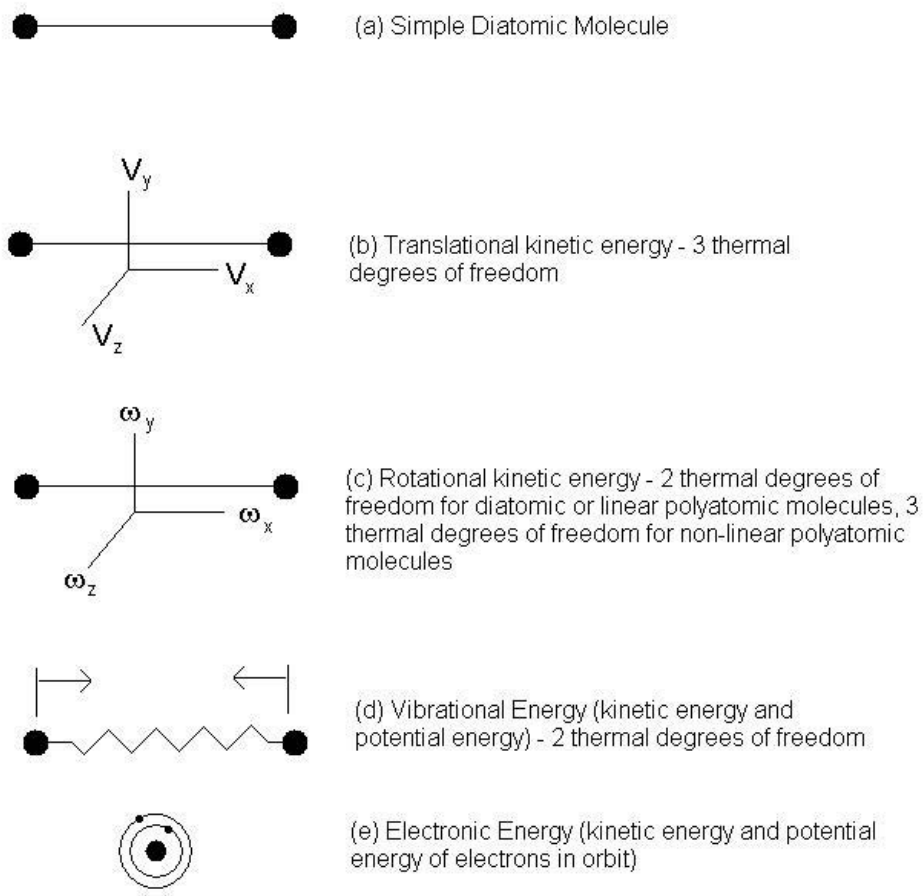

Figure 2.3: Modes of molecular energy

and z, it is said that this mode possesses '3 thermal degrees of freedom'. Each degree of freedom is assumed to be as capable of holding energy as the other translational modes.

- Rotational Energy - the rotational kinetic energy a molecule possesses through rotation about its center of mass. The kinetic energy of a rotating object is proportional to its rotational moment of inertia, which is a proportional to R^2, and its mass. Therefore, in the case of a linear polyatomic molecule, the rotational moment of inertia on its axis is extremely small, and only 2 effective thermal degrees of freedom are added. For non-linear polyatomic molecules, it is assumed that energy can be equally shared around each mode, and 3 effective thermal degrees of freedom are added.

- Vibrational Energy - Molecules are modeled by the connection of individual atoms by springs. Energy can be held by the atoms as a result of this vibration, in addition to the potential energy contained in the springs holding them together. Each mode of vibration will contribute to the vibrational energy a molecule can posses. Therefore, diatomic molecules have two thermal degrees of freedom, one for the kinetic energy of the atoms

and the other for the potential energy held in the intramolecular force. Larger molecules posses higher modes of vibration and thus contribute to more thermal degrees of freedom.

- Electronic Energy - This is energy held in the orbit of the electrons around the nucleus. This will be neglected.

Using statistical mechanics the energy held in each mode can be determined. Each of these modes is then broken up into distinct energy levels, designated as ϵ_i, where i represents the energy level of a specific mode. (i.e. translational, rotational) Another result from quantum mechanics shows that molecular orientation is also quantised. Therefore, it is possible for any given particles with the same energy to have a different orientation. The energy a particle is holding, along with its orientation, is described as its 'state'. Therefore, inside each energy level there are a set number of possible states. If there are N_j molecules at each energy state, then the total energy held in system is

$$E = \sum_j \epsilon_j N_j \tag{2.16}$$

Each different combination of ϵ and N will produce a different macrostate. Over a set of time, as a system of molecules goes to equilibrium, one specific set of N_j for each energy level in each mode will occur more frequently. The macrostate associated with this occurrence is called the most probable macrostate, and is the definition for thermodynamic equilibrium from a statistical mechanics framework. [6] However, recalling that each energy level in each mode can posses a certain number of orientations, it is feasible to assume that any given macrostate can be made up of different combinations of molecules distributed with identical numbers of molecules in each energy level with different orientations. These combinations are called 'microstates', and since the orientation of the molecule has no effect on the total energy in the energy level, and thus on E, any given macrostate could be made up of a large number of possible microstates. Thus, it stands to reason that the most probable macrostate is the one with the largest number of associated microstates, especially if each microstate occurs with equal probability.

In order to count the number of possible microstates for each given macrostate, and thus find the equilibrium macrostate, another important distinction must be made. Molecules possessing an even number of elementary particles shall be called Bosons, and molecules with odd numbers of elementary particles shall be called Fermions. It is shown [6] that the number of Bosons in any given state can be infinite, while only one Fermion can occupy any given state. This will affect the way the microstates are counted.

Assuming each state is equally likely in a specific energy level ϵ_j, the number of distinct ways that N_j molecules can be distributed among g_j possible 'states' for that given energy level is

$$W_j = \frac{[N_j + (g_j - 1)]!}{(g_j - 1)!N!} \tag{2.17}$$

where W_j is the number of possible microstates for the energy level ϵ_j. Each of these possible combinations represents an identical macrostate, but a unique microstate. Now consider all possible energy levels and all molecules, and the net number of microstates is given by

$$W = \sum_j W_j = \prod_j \frac{[N_j + (g_j - 1)]!}{(g_j - 1)!N!} \tag{2.18}$$

It cab be shown that the net number of microstates for Fermions is given by [6]

$$W = \sum_j \frac{g_j!}{(g_j - 1)!N!} \tag{2.19}$$

Taking natural logarithms of both sides of Equation 2.18 we obtain

$$\ln W = \sum_j [\ln(N_j + g_j - 1)! - \ln(g_j - 1)! - \ln(N_j)!] \tag{2.20}$$

Assuming that

- The number of possible orientations g_j for each energy level j is significantly larger than the number of molecules in that level N_j
- The values of g_j and N_j are significantly larger than 1
- The factorial of a natural logarithm is given by

$$\ln x! = x \ln x - x \tag{2.21}$$

We now obtain

$$\ln W = \sum_j \left[N_j \ln \left(1 + \frac{g_j}{N_j} \right) + g_j \ln \left(1 + \frac{N_j}{g_j} \right) \right] \tag{2.22}$$

By taking the derivative of Equation 2.22 and through the use of lagrange multipliers, we find the maximum number of microstates occurs when

$$N_j^* = \frac{g_j}{e^{\alpha} e^{\beta \epsilon_j'} - 1} \tag{2.23}$$

where N_j^* is the number of molecules in each energy level ϵ_j that corresponds to the maximum number of microstates. It can be shown [5] that β is given by

$$\beta = \frac{1}{kT} \tag{2.24}$$

Using this, Equation 2.23 can be simplified to obtain

$$N_j^* = N\frac{g_j \exp\left(-\epsilon_j/kT\right)}{\sum g_j \exp\left(-\epsilon_j/kT\right)} \tag{2.25}$$

Equation 2.25 is referred to as the Boltzmann distribution and provides a value of N_j^* at which the system is in equilibrium. This can be rewritten though the use of the 'state sum', or partition function Q, by

$$N_j^* = N\frac{g_j \exp\left(-\epsilon_j/kT\right)}{Q} \tag{2.26}$$

where Q is defined as

$$Q = \sum_j g_j \exp\left(-\epsilon_j/kT\right) \tag{2.27}$$

It can be shown [6] that Q can be related to many major thermodynamic variables though the equations

$$e = RT^2\left(\frac{\partial \ln Q}{\partial T}\right)_V \tag{2.28}$$

$$h = e + pv = e + RT = RT + RT^2\left(\frac{\ln Q}{\partial T}\right)_V \tag{2.29}$$

$$S = Nk\left(\ln\frac{Q}{N} + 1\right) + NkT\left(\frac{\partial \ln Q}{\partial T}\right)_V \tag{2.30}$$

$$p = NkT\left(\frac{\partial \ln Q}{\partial T}\right)_V \tag{2.31}$$

The value of the partition function Q is given by

$$Q = Q_{trans}Q_{rot}Q_{vib}Q_{elec} \tag{2.32}$$

The value of each of $Q_{trans}, Q_{rot}, Q_{vib}$ and Q_{elec} is shown to be [6, 38, 39]

$$Q_{trans} = \left(\frac{2\pi mkT}{\widetilde{h}^2}\right)^{\frac{3}{2}} V \tag{2.33}$$

$$Q_{rot} = \frac{8\pi^2 IkT}{\widetilde{h}^2} \tag{2.34}$$

$$Q_{vib} = \frac{1}{1 - \exp\left(-\widetilde{h}\widetilde{v}/kT\right)} \tag{2.35}$$

$$Q_{elec} = \sum_{l=0}^{\infty} g_l \exp\left(-\epsilon_l/kT\right) \tag{2.36}$$

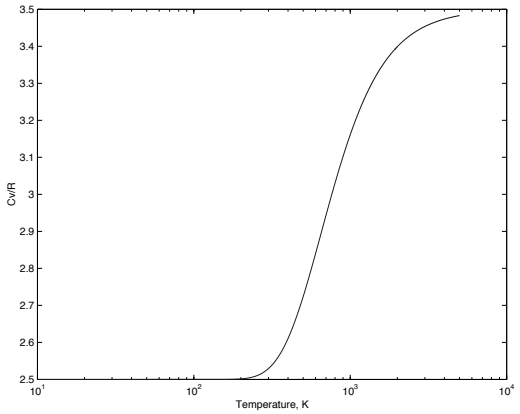

Figure 2.4: C_v/R for Oxygen in temperatures ranging from 10 K to 5000 K.

where $\widetilde{h}$ is Planks constant and $\widetilde{v}$ is a specific fundamental vibrational frequency. (For oxygen, $\widetilde{v} = 4.737\text{x}10^{13}s^{-1}$ [25]) Spectroscopic data is used to obtain the terms $\epsilon_1, \epsilon_2....$ for use in Equation 2.36, though the series can be truncated after 3 terms for molecules where $T \leq 15,000K$.

The energy held by translational modes at equilibrium is found by taking natural logarithms of Equation 2.33

$$\ln Q_{trans} = \frac{3}{2}\ln T + \frac{3}{2}\ln\frac{2\pi mk}{\widetilde{h}^2} + \ln V \tag{2.37}$$

and

$$\left(\frac{\partial(\ln Q_{trans})}{\partial T}\right)_V = \frac{3}{2T} \tag{2.38}$$

Substituting this result into Equation 2.28 obtains

$$e_{trans} = RT^2\left(\frac{\partial(\ln Q_{trans})}{\partial T}\right)_V = RT^2\frac{3}{2T} = \frac{3}{2}RT \tag{2.39}$$

$$e_{rot} = RT^2\left(\frac{\partial(\ln Q_{rot})}{\partial T}\right)_V = RT^2\frac{1}{T} = RT \tag{2.40}$$

$$e_{vib} = RT^2\left(\frac{\partial(\ln Q_{vib})}{\partial T}\right)_V = RT^2\left(\frac{hv/kT^2}{\exp(hv/kT)-1}\right) = \left(\frac{hv/kT}{\exp(hv/kT)-1}\right)RT \tag{2.41}$$

Therefore, the value for C_v for a molecule (disregarding electronic energy) is

$$C_v = \frac{3}{2}R + R + \frac{(hv/kT)^2\exp(hv/kT)}{(\exp(hv/kT)-1)^2}R \tag{2.42}$$

A plot of C_v/R for Oxygen is shown in Figure 2.4 following Equation 2.42. At lower temperatures, only the translational and rotational modes are excited, giving 5 thermal degrees of freedom. At higher temperatures vibrational excitation occurs and (when fully excited) adds

another 2 thermal degrees of freedom. The energy per particle can be taken from this expression, or from experimental observations, as

$$E_i = C_{part} \equiv \frac{1}{2}(\zeta_{dof} - \zeta_{trans})RT \tag{2.43}$$

where $\zeta_{dof} = 2C_v/R$ is the total number of thermal degrees of freedom and ζ_{trans} is the number of 'utilized' translational degrees of freedom. E_i is the internal energy (i.e. disregarding translational energy) resulting from temperature T.

2.6 Equilibrium Distributions of Energy and Velocities

In a small region of dilute gas, disregarding external forces, it can be shown that the Boltzmann Equation can be rewritten as

$$\frac{df}{dt} = 2\pi \int_{-\infty}^{\infty} \int_{0}^{\pi} n \left[f(v_1')f(v_2') - f(v_1)f(v_2) \right] gS(g,\chi) \sin\chi d\chi dv_2 \tag{2.44}$$

In this case, χ represents a deflection angle following a binary collision, v_1 and v_2 represent molecular velocities and v_1', v_2' represent post collision velocities. Boltzmann's H-theorem shows the irreversible nature of non-equilibrium systems and is shown by defining the quantity

$$H(t) = \int_{-\infty}^{\infty} f ln(nf) d\mathbf{v} \tag{2.45}$$

and substituting into equation 2.44, leaving

$$\frac{dH}{dt} = \int_{-\infty}^{\infty} \left[1 + ln(nf) \right] \frac{df}{dt} d\mathbf{v} \tag{2.46}$$

It can be shown [15, 19, 41] that the value dH/dt is always negative and therefore H always decreases until it reaches an equilibrium state where $dH/dt = 0$. Boltzmann's H-theorem essentially states that a system of molecules, through binary collisions, must move toward equilibrium.

Using the H-theorem, the equilibrium Maxwell-Boltzmann distribution of molecular thermal velocities is given by

$$f_{eq}(c_x) = \left(\frac{m}{2\pi kT} \right)^{\frac{1}{2}} \exp\left(-\frac{m}{2kT} c_x^2 \right) = \frac{1}{\sqrt{2\pi} s} \exp\left(-\frac{c_x^2}{2s^2} \right) \tag{2.47}$$

where c is the thermal velocity of a molecule in direction x, $s = \sqrt{RT}$ is the standard deviation of the distribution, m is the molecular mass of the molecule, k is Boltzmann's constant, R is the specific gas constant and T is the temperature. This distribution has a mean of 0 and a variance of RT. The equilibrium probability distribution function for molecular velocities v_x from the laboratory frame of reference is:

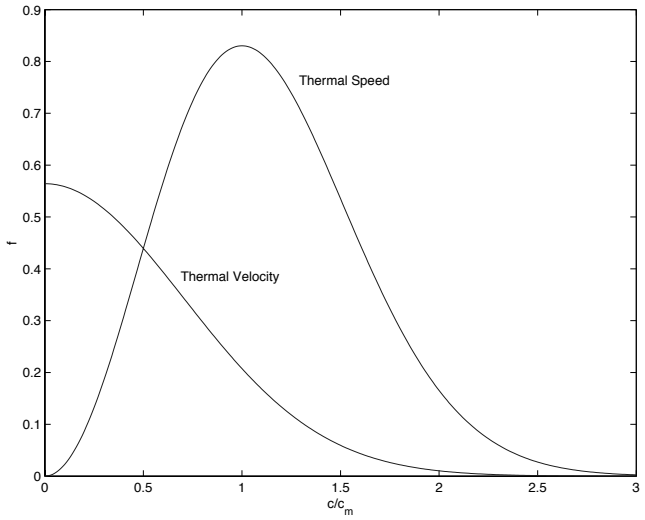

Figure 2.5: Equilibrium distributions of thermal velocity and thermal speed as a function of normalised thermal speed and velocity.

$$f_{eq}(v_x) = \frac{1}{\sqrt{2\pi}s} \exp\left(-\frac{(v_x - \overline{v_x})^2}{2s^2}\right) \tag{2.48}$$

where $\overline{v_x}$ is the mean (or bulk) velocity of the gas particles. The distribution of thermal speeds $c = |c_x, c_y, c_z|$ is given by

$$f(c) = 4\pi c^2 \left(\frac{1}{2\pi RT}\right)^{\frac{3}{2}} \exp\left(-\frac{c^2}{2RT}\right) \tag{2.49}$$

For a set of molecules in thermal equilibrium, the most probable thermal speed c_m is

$$c_m = (2kT/m)^{\frac{1}{2}} = (2RT)^{\frac{1}{2}} \tag{2.50}$$

and the mean thermal speed is

$$\overline{c} = \left[\frac{8kT}{\pi m}\right]^{\frac{1}{2}} = \sqrt{\frac{8RT}{\pi}} \tag{2.51}$$

The equilibrium distributions of velocity and speed are shown in Figure 2.5 with only the positive values of thermal velocity shown. The molecular speed and velocity are normalised to the most likely molecular velocity. It can be seen that the chance of a molecule possessing a thermal velocity c_x, c_y or c_z larger than 3 variances away from the mean is very unlikely, with approximately 1 in 50,000 particles possessing such a velocity.

2.7 Mean free path and collision time

For a given viscosity μ, the approximate nominal mean free path can be shown to be [107]:

$$\lambda = \frac{2\mu}{\rho\overline{c}} \tag{2.52}$$

where ρ is the density of the gas. The mean free time τ is simple defined as:

$$\tau = \frac{1}{\nu} \tag{2.53}$$

where ν is the collision rate of the gas. The number of collisions in a region of gas over the time Δt is

$$N_{coll} = \frac{1}{2} N \nu \Delta t \tag{2.54}$$

where N is the number of particles in the given region.

2.8 Kinetic Temperatures

This chapter briefly reviews the relations required to calculate the overall kinetic temperature T_k in a gas. The translational temperature is calculated by taking a second moment of the velocity distribution function. The total energy of particles per unit mass in direction x, neglecting structural internal energy, is:

$$e_x = \int_{-\infty}^{\infty} \frac{1}{2} (v_x)^2 f(v_x) dv_x \tag{2.55}$$

If the gas is in thermal equilibrium, the above integral becomes:

$$\begin{aligned} e_x &= \int_{-\infty}^{\infty} \frac{1}{2} (v_x)^2 f_{eq}(v_x) dv_x \\ &= \frac{1}{2} \left(\overline{v_x}^2 + s^2 \right) \end{aligned}$$

The kinetic energy can clearly be divided into a bulk kinetic energy component $(1/2)\overline{v_x}^2$ and a thermal energy component $(1/2)s^2 = (1/2)RT_x$. This result is in agreement with Equation 2.39 since there are three translation degrees of freedom ($\varsigma_{trans} = 3$) and, in thermal equilibrium, each translational degree of freedom must hold the same amount of energy. The total thermal kinetic energy due to the degrees of freedom associated with translation will be:

$$e_{trans} = e_{trans,x} + e_{trans,y} + e_{trans,z} \tag{2.56}$$

A real gas made of discrete particles may not be in thermal equilibrium and the velocity distribution function is generally unknown. Thus, the thermal kinetic energy of the gas can be found by taking samples of the particle population. In this case, the overall kinetic thermal energy is found by summing each degree of freedom's energy. Regardless, the kinetic temperature for

translation will be:

$$\begin{aligned} T_{trans} &= \frac{2e_{trans}}{\varsigma_{trans}R} \\ &= \frac{2\left(e_{trans,x} + e_{trans,y} + e_{trans,z}\right)}{3R} \\ &= \frac{1}{3}(T_x + T_y + T_z) \end{aligned} \tag{2.57}$$

The temperature of a gas can be calculated in terms of the number of degrees of freedom. Recalling Equation 2.43, the temperature is defined as:

$$T_k = \frac{2E}{\varsigma_{dof}R} \tag{2.58}$$

where E is the total energy as a result of the temperature T_k. If only rotational and translational modes are considered, E is a linear function of T_k and thus T_k is simple to solve. However, vibrational modes are non linear with T_k, as demonstrated by Equation 2.41. Thus, at higher temperatures the value of C_v increases as demonstrated in Figure 2.4. Therefore, the calculation of an effective number of vibrational degrees of freedom is required. This value, ς_{vib}, can be calculated from 2.41 and then applied to Equation 2.58. The calculation of the temperature of a region of gas where the effective degrees of freedom for vibration is not constant with temperature usually requires an iteration procedure to determine the temperature for any given energy.

Chapter 3

Governing Flow Equations and Properties

3.1 Introduction and Summary

This chapter begins with the definition of the Knudsen number and the continuum breakdown parameter. These quantities are useful guides in classifying flow regimes. In rarefied hypersonic flows the gas rarely has sufficient time for enough collisions to occur for the gas to reach local thermal equilibrium. In such regions, the classic definition of temperature fails as the kinetic temperatures in each degree of freedom can differ from each other. In such flows, direct solvers such as DSMC and BGK solvers can be used to properly solve the flow. The breakdown parameter is a concept used to predict the breakdown of convention continuum solvers and is discussed. The BGK equation is then discussed, with the famous Euler equations following this. The Navier stokes equations are also introduced.

3.2 Knudsen Number

The Knudsen number is a dimensionless quantity which characterises the degree of rarefaction. It is defined as:

$$\mathrm{Kn} \equiv \frac{\lambda}{L} \tag{3.1}$$

where L is a characteristic dimension. For any given mean free path, a measure of the rarefaction is provided by this dimension. Flow is typically designated as rarefied for Kn values larger than 1. For example, the mean free path of air at standard temperature and pressure (STP) is on the order of $0.1\mu m$. The Knudsen number for flow around a vehicle of length $L = 2m$ would be $\mathrm{Kn} = 5e-8$. At these Knudsen numbers, the flow is collision dominated - there are almost certainly enough collisions to ensure thermal equilibrium. However, at much lower scales such as those involved in the design of computer hard drives, the characteristic distance is on the order of a mean free path. In this instance, there may not be enough collisions to guarantee local thermal equilibrium and a direct solver may be used.

In the previous examples the characteristic dimension L was a physically measured dimension. An alternative to this is to define the length in terms of the local gradient length scale:

$$L = Q \left| \frac{\partial Q}{\partial x} \right|^{-1} \tag{3.2}$$

where Q is some arbitrary flow property such as the local density or local mean free path. Using a length based on the local gradient length scale is a useful tool for predicting continuum breakdown [102]. This concept is easily applicable to CFD methods when the gradient $\partial Q/\partial x$ is easily and accurately calculated. However, in direct simulations such as DSMC, the statistical scatter can made it difficult to accurately calculate.

3.3 Continuum Breakdown Parameter

Previous discussion on the local Knudsen number has been independent of the local flow speed. In a flow where the Knudsen number is quite low, one might expect thermal equilibrium. However, if the flow speed is sufficiently high, the particles are swept downstream before there is opportunity to reach a local thermal equilibrium. Bird's definition of the breakdown parameter is based on the inclusion of this flow speed.

Flow speed can be taken into account by examining the time required to pass by an object of length L. This time τ_{flow} is simply $\tau_{flow} = L/u$, where u is the bulk velocity $\overline{v_L}$ in the direction of the orientation of length L. The ratio of this flow time to the local mean collision time τ is:

$$\frac{\tau}{\tau_{flow}} = \frac{\lambda u}{D\overline{c}} = \left(\frac{\pi\gamma}{8}\right)^{\frac{1}{2}} \mathrm{KnM} \tag{3.3}$$

Thus, the product Kn M might be applied as a continuum breakdown parameter. Following Bird [13] the following breakdown parameter to predict continuum breakdown in gaseous expansions:

$$P = \frac{u}{\rho\nu} \left| \frac{d\rho}{dx} \right| \tag{3.4}$$

Macrossan [68] proposed the modification

$$P = \frac{u_{char}}{\overline{c}} \frac{\lambda}{l_{char}} \tag{3.5}$$

where u_{char} is a characteristic speed and l_{char} a characteristic length. By replacing the characteristic speed u_{char} with the mean thermal speed and the characteristic length l_{char} with the gradient length calculated using the local mean free path, Macrossan's breakdown parameter reduces to:

$$P = \left| \frac{d\lambda}{dx} \right| \tag{3.6}$$

3.4 Kinetic CFL number

Traditional CFL numbers are defined by the fraction of distance a disturbance can propagate across a cell in a given time step to the local cell size. In most existing contexts, the speed

at which a disturbance can travel is the local speed of sound. Since this study focuses on the behaviour of a gas from a kinetic theory perspective, a kinetic CFL number is introduced as:

$$CFL = \frac{(|V| + \sigma\sqrt{RT})\Delta t}{\Delta x} \tag{3.7}$$

where σ is a selected number of variances of the equilibrium distribution and $|V|$ is the magnitude of the velocity in the cell. Higher values of σ ensure that the surrounding neighbours of the source cell capture a larger fraction of the mass. Small values of σ allow the time step to be large enough for particles to travel in free-flight beyond the surrounding neighbours. If these distant cells are not registered as neighbours to the source cell, then this flux will be neglected and the results will be inaccurate.

The meaning of this CFL number can be physically defined. Let a particle possess a velocity of $V + 5(RT)^{0.5}$ where, in this instance, $\sigma = 5$. The chance of a particle possessing this velocity is extremely low - only approximately 1 particle out of 1e+12 particles will have a velocity this large at equilibrium conditions. The kinetic CFL defined in Equation 3.7 represents the fraction of the distance across a cell that this particle will be able to travel in free flight. The traditional definition of the CFL number (or Courant number) is:

$$CFL = \frac{(|V| + \sqrt{\gamma RT})\Delta t}{\Delta x} \tag{3.8}$$

The Courant number arises from stability analysis of the finite difference representations of the Euler Equations. In a conventional finite volume solver this value physically represents the largest fraction of a cell width that a propograting wave can span. Courant numbers larger than 1 are traditionally associated with instability in finite volume solvers.

3.5 BGK Equation

For flows in which the transition time is small in comparison to the mean free path, the gas is not in thermal equilibrium and continuum methods are no longer valid. In these regions it becomes necessary to solve the Boltzmann equation. However, the collision term of the Boltzmann equation is quite complex, making even simple flows very difficult to solve. It is for this reason that various flow model equation methods have been developed. The Boltzmann equation is shown in Equation 3.9 and the collision term is given in Equation 3.10.

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \frac{\partial f}{\partial \mathbf{x}} + \mathbf{F} \cdot \frac{\partial f}{\partial \mathbf{v}} = \left[\frac{\partial f}{\partial t}\right]_{coll} \tag{3.9}$$

$$\left[\frac{\partial f}{\partial t}\right]_{coll} = \int_{-\infty}^{\infty} \int_{0}^{4\pi} \left(f^* f_1^* - f f_1\right) c_r \sigma \, \mathrm{d}\Omega \, \mathrm{d}\mathbf{c}_1 \tag{3.10}$$

The Bhatnagar-Gross-Krook (BGK) model equation [11] (also proposed by Weylander [109]) replaces the collision term of the Boltzmann equation by a simpler source term which retains the

important features of the original. In the BGK method, collisions are represented by a relaxation at a finite rate of the velocity distribution function to the equilibrium distribution. This method may be implemented in a simple, decoupled approach by solving the linear advection equation and then performing the relaxation within each time step.

Unfortunately, the BGK model equation requires a significant amount of computational power as each step of the solution must be performed over the entire range of molecular velocities. It becomes necessary, therefore, to discover methods of reducing the computational expense to a reasonable amount. One possible approach is to use an accurate linear advection solver that is stable for CFL numbers greater than 1. This is discussed in more detail later.

One of the biggest advantages of the BGK model equation when compared to the more common rarefied methods such as DSMC solvers is the lack of statistical scatter in the results. Another advantage is its validity over a large range of Knudsen numbers while DSMC is limited to dilute flows only.

The BGK model equation without external forces is shown below in Equation 3.11, where ν_M is the collision frequency and f_M is the local Maxwellian velocity distribution function.

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \frac{\partial f}{\partial \mathbf{x}} = \nu_M \left(f_M - f \right) \tag{3.11}$$

where the collision frequency is given by

$$\nu_M = \frac{\rho R T}{\mu}$$

The local Maxwellian velocity distribution function in three dimensions is given by

$$f_M = \frac{n}{(2\pi RT)^{\frac{3}{2}}} \exp\left(\frac{-c^2}{2RT}\right) \tag{3.12}$$

where $c^2 = (u - \overline{u})^2 + (v - \overline{v})^2 + (w - \overline{w})^2$.

The BGK method conserves mass, momentum and energy;

$$\int \int \nu_M \left(f_M - f \right) \psi_\alpha \, d\mathbf{v} = 0 \tag{3.13}$$

where $\alpha = 1, 2, 3, 4, 5$ and $\psi_\alpha = \left\{1, u, v, w, \frac{1}{2}\mathbf{v}^2\right\}$ and
$\mathbf{v} = u + v + w$.

As the Boltzmann equation describes irreversible processes, the entropy must increase. A rigorous proof of the entropy condition for the Boltzmann equation is presented in [20]. Defining the entropy density as $\mathcal{H}(f) = f \ln f d\mathbf{v}$, the BGK method also satisfies the entropy condition

$$\frac{\partial \mathcal{H}}{\partial t} + \frac{\partial \mathcal{H}_i}{\partial x_i} \leq 0 \tag{3.14}$$

The basic physical assumption underpinning the modified Boltzmann equation is that the collision frequency is independent of the relative speed of the colliding molecules [51]. This assumption is justified only for Maxwellian particles.

In recent years, the development of the gas-kinetic BGK model equation method has been quite strong. Single and multidimensional BGK methods have been developed.Aoki *et al.* [9] used the model equation to solve for unsteady flow between two infinite parallel plates. Kim *et al.* [48] adapted a finite volume gas-kinetic BGK method to unstructured triangular grids with mesh adaption.

One of the most significant drawbacks of the BGK method is that it does not result in the correct Prandtl number. Holway [42] introduced the 'ellipsoidal statistics' (ES-BGK) method in order to fix this issue. The ES-BGK model [42] was developed to adjust the BGK model to give the correct transport coefficients for the Navier-Stokes equation by relaxing toward a Gaussian equilibrium distribution rather than a Maxwellian and allowing for the addition of the Prandtl number to the collision term. Andries *et al.* [8] conducted some numerical comparisons between this method, the standard BGK method and DSMC for transitional reentry flows that showed reasonable improvements in most flow aspects. Chae *et al.* [17] also improved the BGK method by correcting the Prandtl number. Li and Zhang [53, 54, 55] also modified the BGK method to correct for the Prandtl number and removed the continuous dependence of the distribution function on the velocity space through their discrete velocity ordinate method. This model is of particular interest in this work and will be described in detail in later sections.

Macrossan [67] developed the 'relaxation time simulation method' (RTSM) by extending the Equilibrium Particle Simulation Method (EPSM) of Pullin [80]. EPSM works in a similar way to the Direct Simulation Monte Carlo (DSMC) methods, but rather than simulating collisions, the momentum and energy of the particles in each cell are redistributed toward local Maxwellian equilibrium at each time step. RTSM differs in that only a fraction of the particles are adjusted to equilibrium.

In order to include descriptions of thermal non-equilibrium, multi-temperature BGK methods have been developed in recent years [46, 119, 120, 121]. Xu *et al.* [118] also developed a diatomic gas BGK method with rotational and translational degrees of freedom included.

Gross and Krook [33] extended the original BGK method to allow for two component gas mixtures. One major drawback of their method is that when both species are defined identically, the one component method is not recovered. Bhatnagar [12] also continued development on this two component method, as did Sirovich [92]. In the paper by Andries *et al.* [7], a two component method is presented which satisfies positivity and the entropy inequality, has the correct exchange coefficients and degenerates to the single component model.

3.6 Euler Equations

The flow of an inviscid, compressible gas is to be considered. The behaviour of such a gas is governed by the Euler equations. In two dimensional form (with the absence of source terms) these equations are given by

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} + \frac{\partial \mathbf{G}}{\partial y} = 0 \tag{3.15}$$

where

$$\mathbf{U} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{bmatrix} \mathbf{F} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uH \end{bmatrix} \mathbf{G} = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vH \end{bmatrix} \tag{3.16}$$

For an ideal gas, the total energy E and total enthalpy H can be written as

$$\begin{aligned} E &= \tfrac{p}{\rho(\gamma-1)} + \tfrac{1}{2}\left(u^2 + v^2\right) \\ H &= E + \tfrac{p}{\rho} \end{aligned} \tag{3.17}$$

The Euler equations are the limiting form of the general viscous flow equations in the limit of infinite Reynolds number [5].

3.7 Navier-Stokes Equations

The flow of a viscid, heat conducting, unsteady compressible gas is considered. The governing equations for flow in 2 dimensions can be expressed as

$$\frac{\partial}{\partial t}\int\int_{\Omega} \mathbf{U} dxdy + \int_S (\mathbf{F} - \mathbf{F}_v)\, dy - \int_S (\mathbf{G} - \mathbf{G}_v)\, dx = \int\int_{\Omega} \mathbf{Q} dxdy \tag{3.18}$$

where

$$\mathbf{U} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{bmatrix} \mathbf{F} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uH \end{bmatrix} \mathbf{G} = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vH \end{bmatrix} \tag{3.19}$$

represent the conserved quantities and inviscid flux vectors as shown in Equation 3.15 and

$$\mathbf{F}_v = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{yx} \\ \tau_{xx}u + \tau_{yx}v + q_x \end{bmatrix} \mathbf{G}_v = \begin{bmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ \tau_{xy}u + \tau_{yy}v + q_y \end{bmatrix} \tag{3.20}$$

are the viscid flux vectors. Q is a vector of source terms which will be neglected in this study. The total energy E and total enthalpy H are as shown in Equation 3.17. The viscous stresses are given by

$$\begin{aligned}\tau_{xx} &= 2\mu\frac{\partial u}{\partial x} + \eta\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)\\ \tau_{yy} &= 2\mu\frac{\partial v}{\partial y} + \eta\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)\\ \tau_{xy} &= \mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\end{aligned} \tag{3.21}$$

where μ and η are the first and second coefficients of viscosity. Stokes' hypothesis (of zero bulk viscosity) can be used to give $\eta = -2/3\mu$ [43] and the viscosity μ can be taken from measurements or Sutherland's law. Neglecting the presence of multiple species, the heat fluxes are

$$\begin{aligned}q_x &= \widetilde{k}\frac{\partial T}{\partial x}\\ q_y &= \widetilde{k}\frac{\partial T}{\partial y}\end{aligned} \tag{3.22}$$

where $\widetilde{k}$ is the heat transfer coefficient for the gas in question.

Chapter 4

Finite Volume methods in Computational Fluid Dynamics

4.1 Introduction and Summary

Finite volume methods are based on a discretization of the integral forms of the conservation equations [16]. The flow field is divided up into a large number of discrete volumes called 'cells'. Fluxes of mass, momentum and energy are then exchanged across the surfaces of the cell. In order to better understand this, the procedure is described in terms of the conservation integral equation. The general form of the conservation integral equation is given by:

$$\frac{\partial}{\partial t}\int\int\int_{\Omega} U d\Omega + \int\int_{S} \vec{F}\cdot d\vec{S} = \int\int\int_{\Omega} Q_v d\Omega + \int\int_{S} Q_s dS \tag{4.1}$$

This is simply a more general form of Equation 3.18. The two dimensional discretised form of this equation is given by [16]:

$$\frac{\partial}{\partial t}(U\Omega_{ij}) + \sum_{\text{sides}}(\vec{F}\cdot\vec{S}) = (Q_v)_{ij}\Omega_{ij} + \sum_{\text{sides}}(Q_s)_{ij}S \tag{4.2}$$

where Ω_{ij} is the volume of the cell (or finite volume element) with an indexing of i, j as shown in Figure 4.1 and $\vec{F}$ is the vector of fluxes of conserved quantities through surface S. Source terms, or mechanisms by which mass, momentum or energy are added to the cell without going through a cell surface, are represented as Q_v. Consider the application of equation of Equation 4.2 to the Euler equations demonstrated in § 3.5. Without source terms and enforcing the notation used in Equation 4.2, the Euler equations can be rewritten as:

$$\frac{\partial U}{\partial t} + \frac{\partial F_1}{\partial x} + \frac{\partial G_1}{\partial y} = -\left(\frac{\partial F_2}{\partial x} + \frac{\partial G_2}{\partial y}\right) \tag{4.3}$$

where

$$U = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ E_t \end{bmatrix} F_1 = \begin{bmatrix} \rho u \\ \rho u^2 \\ \rho uv \\ uE_t \end{bmatrix} F_2 = \begin{bmatrix} 0 \\ p \\ 0 \\ pu \end{bmatrix} G_1 = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 \\ vE_t \end{bmatrix} G_2 = \begin{bmatrix} 0 \\ 0 \\ p \\ pv \end{bmatrix} \tag{4.4}$$

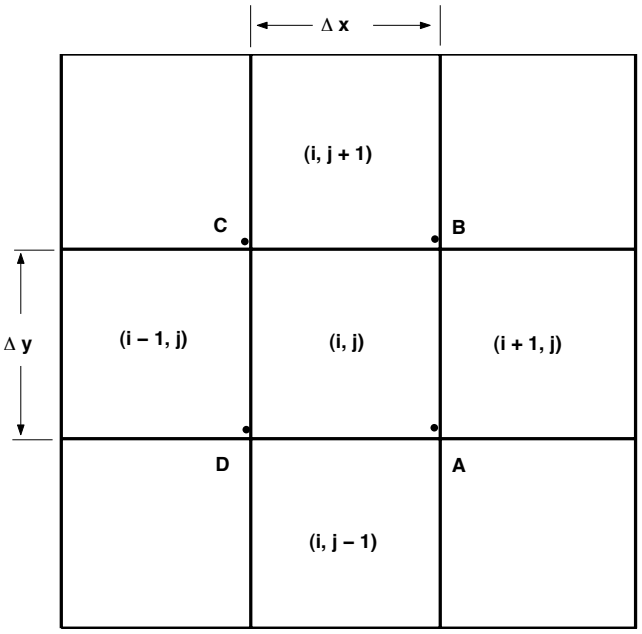

Figure 4.1: Finite volume representation of space showing discretisation into computational cells.

where E_t is equal to ρE defined in Equation 3.17. The components of the flux vectors E_1 and F_1 can be represented by the single flux vector $\vec{F}$ with e and f such that [16]:

$$\vec{F} = e\vec{i} + f\vec{j} \tag{4.5}$$

The flux term in Equation 4.2 can now be written as:

$$\begin{aligned} \sum_{\text{sides}} \vec{F} \cdot \vec{S} &= e_{AB}\Delta y + f_{BC}\Delta x - e_{CD}\Delta y - f_{DA}\Delta x \\ &= (f_{BC} - f_{DA})\Delta x + (e_{AB} - eCD)\Delta y \end{aligned} \tag{4.6}$$

The evaluation of the fluxes around the surfaces of element (i, j) will depend on the spatial scheme selected. Hirsch [40] discusses many possible choices including the cell centered approach. In this case, the fluxes at the interface are simply the average fluxes of the adjoining cells. For example:

$$e_{AB} = \frac{1}{2}(e_{i+1,j} + e_{i,j}) \tag{4.7}$$

This flux is essentially a one dimensional flux normal to the surface AB of cell $(i, j$. It is clear to see from Equation 4.6 that fluxes of mass, momentum and energy can only be exchanged between cells that share some common interface or surface. There is no allowance for the fluxes to be exchanged to diagonally adjacent cells since the term $\vec{S}$ will always be zero. The concept of evaluating a one dimensional flux across cell surfaces is referred to here as 'direction decoupling'. The numerical and theoretical consequences of this are discussed in the following section.

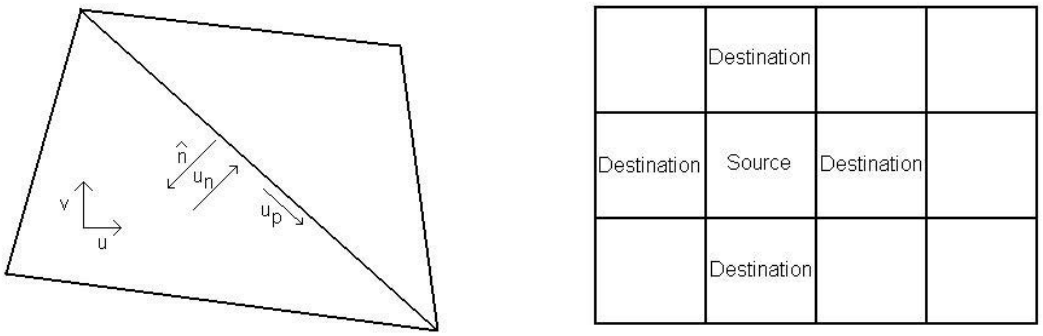

Figure 4.2: (Left) Direction decoupling at a cell interface. Components of the flow are broken into components normal and parallel to the cell interface and 1 dimensional fluxes are calculated. (Right) Representation of cells on a regular rectangular mesh for use in an existing continuum solver. Fluxes are only exchanged between the source and destination cells.

The method of solving a finite volume problem by examining the fluxes around the surfaces is also commonly referred to as Godunov's method. A solver which employs this approach is commonly referred to as a Godunov solver. Godunov's influential 1958 paper [28] presented a method where the conditions in a cell where continuous over the cell volume, i.e. uniform. Godunov's famous theorem stated that as long as a solver employed uniform conditions throughout the cell the accuracy could be no greater than first order. The proof for this is based on a linear approximation to the advection scheme, thus if a flux calculation employed a higher order (i.e. non linear) method there is an opportunity to increase the order of accuracy [106].

The evolution of a gas at an interface with differing conditions on either side is referred to as Riemann's initial value problem. Thus a Riemann solver is referred to here as a method which solves Riemann's problem. In these family of solvers, the fluxes across an interface depend on conditions on either side and thus are labeled difference split fluxes. There are many exact and approximate solutions to this problem: these are discussed in more detail later.

Through the assumption of local equilibrium, the Kinetic Theory of Gases can also be used as a tool to evaluate the fluxes across surfaces. One such method, Pullin's Equilibrium Flux Method [80], is discussed. A mathematical splitting of the fluxes across an interface is also possible, as performed by Van Leer's flux method [105]. These methods both rely upon splitting the fluxes across an interface and are commonly called vector split fluxes. Finally, the proceedure of slope limiting is discussed.

4.2 Direction Decoupling

Existing CFD methods utilise 'direction decoupling', as described in Figure 4.2. Fluxes of mass, momentum and energy are exchanged between cells sharing common interfaces. These fluxes are calculated by finding the components of the flow normal and parallel to the interface and

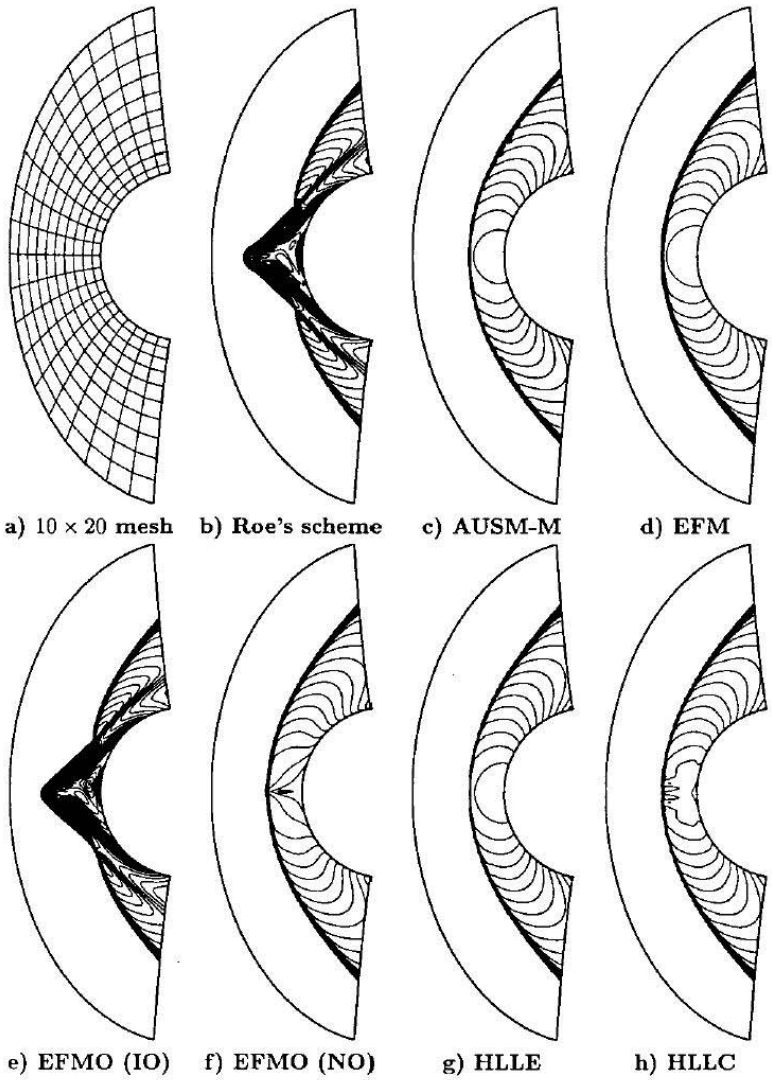

Figure 4.3: Examples of the carbuncle phenomena from [31] for flow over a cylinder at $M = 10$ for various schemes. Plotted are contours of temperature.

then calculating 1 dimensional fluxes. In this way, simulations of 2 and 3 dimensions can be performed. Fluxes calculated in this fashion are directly proportional to the size of the time step and of the shared interface area. Pullin's [80] Equilibrium Flux Method is an example of such a solver. As shown on Figure 4.2, only cells with a shared interface will exchange fluxes. This is physically unrealistic - from common sense and kinetic theory, we know that fluxes should be exchanged between all cells where particles could travel in a given time step.

A direct consequence of this is that flows traveling in a direction not aligned with the mesh are not captured in a single time step. Flows unaligned with the mesh in a finite volume CFD method can create physically unrealistic results [93, 71, 98, 24] especially for unsteady simulations. This feature of existing methods may be partly responsible for the Carbuncle phenomenon. The carbuncle phenomenon is a shock instability mechanism which ruins all efforts to compute grid-aligned shock waves using low-dissipative upwind schemes [32]. This

phenomena often occurs in the simulation of high speed flows, and is commonly represented by a section of a bow shock extruding away from the body causing it. Examples of this are shown in Figure 4.3 taken from [31]. The phenomena was first discovered by Peery and Imlay [78]. It consists of a spurious stagnation point which moves the shock upstream along the symmetry axis. The carbuncle phenomenon is highly grid-dependent, but does not require a large number of points to appear [32].

Direct simulation (or particle based) methods, such as DSMC, EPSM and PFM calculate mass, momentum and energy fluxes through the direct simulation of particles. Since these particles are permitted to travel in any direction within a given time step, flux exchanges are not limited to cells sharing an interface. There is no evidence to suggest that direct simulations suffer from the Carbuncle phenomenon. Therefore, it is not important in direct simulations that the mesh be aligned with the flow. These methods are very stable - in fact, the size of the time step is controlled only by the demand for physically meaningful results [79] instead of stability. The main reasons that direct methods are not more commonly used is the computational expense and statistical scatter apparent in the results. Efforts have been made to make DSMC faster for use in near continuum flows [68, 66] and recent efforts by various authors including Long [60] have used 'collision limited' DSMC as a continuum solver.

Despite these advances, direct simulations used as continuum solvers are still very expensive solutions to the Euler equations. A continuum method, derived from kinetic theory with the conceptual strengths of direct simulations and the speed of continuum solvers would be extremely useful in situations where accurate results are required quickly. Examples of such situations are in the materials handling industry, outlined by van den Berg [103], in the simulation of blast waves through urban tunnel networks. Other research [60, 83] has been conducted on the effects of blast waves on structural bodies and in the mining industry. These are examples of situations where there is inadequate time to run a direct simulation and results from direction decoupled solvers may be inadequate.

4.3 Riemann Solvers

The Riemann initial value problem is defined as the evolution of gas at an interface separating two gases [106]:

$$U_o(x) = \begin{cases} U_l, \mathrm{x} < 0 \\ U_r, \mathrm{x} > 0 \end{cases} \tag{4.8}$$

Depending on the conditions, four conceptually possible outcomes can result as shown in Figure 4.4. From these, only two are physically permitted to happen. Figure 4.4(a) and (b) show propagating shock and expansion waves separated by a contact discontinuity. Examining the conditions in Figure 4.4(a) from right to left, the pressure increases across the shock wave, remains constant across the contact discontinuity and increases again through the expansion wave. For most families of Riemann solvers, the motion of the shock wave, contact discontinuity

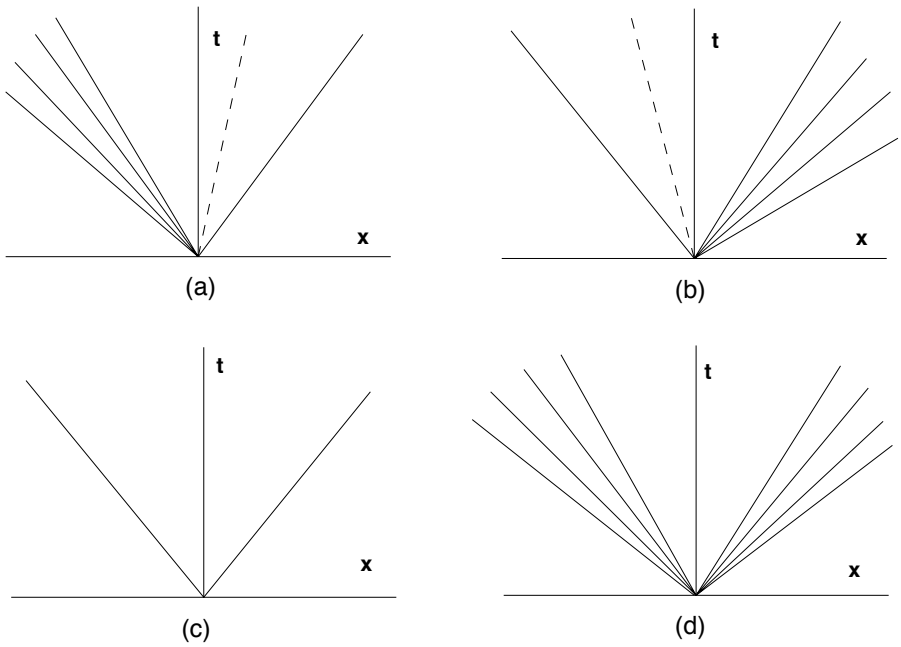

Figure 4.4: Conceptually possible outcomes from the Riemann initial value problem. (a) A shock wave propagating right with an expansion wave propagating left, (b) A shock wave propagating left with an expansion wave propagating right, (c) two propagating shock waves, (d) two propagating expansion waves.

and the expansion wave are linear in time. Therefore, at any time t the conditions are constant in time and are determined by the location of the shock wave, contact surface and expansion wave.

There are a large number of Riemann solvers available, including 'exact' iterative schemes [30] and approximate (non-iterative) schemes [87, 77, 26]. Here we will focus on Jacobs approximate, non-iterative Roe solver [44, 43]. This solver is selected due to its relatively low computational expense and track record. The general procedure used to calculate the flux of mass, momentum and energy is:

1. Calculate the Riemann invariants:

$$\begin{aligned} \overline{U_L} &= u_L + \frac{2a_L}{\gamma - 1} \\ \overline{U_R} &= u_R - \frac{2a_R}{\gamma - 1} \end{aligned}$$

   where u is the velocity normal to the interface and a is the local speed of sound. Subscripts L and R indicate reference to conditions on the left and right sides of the interface.

2. Guess the value of the pressure and velocity in the intermediate region P^* and u^* between

the shock and expansion wave using perfect gas relations [43, 29].

$$\begin{aligned} u^* &= \frac{\overline{U_L}z + \overline{U_R}}{1+z} \\ P^* &= P_L\left(\frac{\gamma-1}{2}\right)\left(\frac{\overline{U_L}-\overline{U_R}}{a_L(1+z)}\right)^{\frac{2\gamma}{\gamma-1}} \\ z &= \left(\frac{a_R}{a_L}\right)\left(\frac{P_L}{P_R}\right)^{\frac{\gamma-1}{2\gamma}} \end{aligned}$$

3. If the value of P^* is larger than $10P_L$ and $10P_R$ both waves may be taken as strong shock waves. In this case, the estimate of the pressure and velocity in the intermediate region can be calculated from shock relations:

$$\begin{aligned} P^* &= \frac{\gamma+1}{2}\rho_L\left[\frac{\sqrt{\rho_R}}{\sqrt{\rho_R}+\sqrt{\rho_L}}(u_L-u_R)\right]^2 \\ u^* &= \frac{\sqrt{\rho_L}u_L + \sqrt{\rho_R}u_R}{\sqrt{\rho_R}+\sqrt{\rho_L}} \end{aligned}$$

4. If the value of P^* is larger than either P_L or P_R, the estimate of P^* and u^* can be improved by taking multiple Newton-Raphson steps of the form [43]:

$$P^*_{\text{NEW}} = P^*_{\text{OLD}} - F_{\text{OLD}}\left(\frac{dF_{\text{OLD}}}{dP^*}\right)^{-1}$$

Expressions for F_{OLD} and its derivative can be found in [43].

5. The values of P^* and u^* are used to calculate other flow properties. Focusing on a left moving wave, if the pressure increases across the wave (i.e. $P^* > P_L$) the left moving wave is assumed to be a shock. In this case, the density is calculated using the Rankine-Huginoit relation:

$$\begin{aligned} \rho^*_L &= \rho_L\left[\frac{(\gamma+1)P^* + (\gamma-1)P_L}{(\gamma+1)P_L + (\gamma-1)P^*}\right] \\ e^*_L &= \frac{P^*}{(\gamma-1)\rho^*_L} \\ a^*_L &= (\gamma(\gamma-1)e^*_L)^{\frac{1}{2}} \end{aligned}$$

where e^*_L is the specific internal energy. The wave velocity w, relative to the velocity of the initial left state, is:

$$u_L - w_L = \left[\frac{\gamma+1}{2}\frac{P^*}{\rho_L}\left(\frac{P^*}{P_L}+\frac{\gamma-1}{\gamma+1}\right)\right]^{\frac{1}{2}} \tag{4.9}$$

6. If the pressure decreases across the left wave (i.e. $P^* < P_L$) the wave is assumed to be an expansion. In this case, the isentropic flow equations and the Riemann invariants are

used to obtain the conditions:

$$\begin{aligned} a_L^* &= \frac{1}{2}\left(\overline{U_L} - u_L^*\right)(\gamma - 1) \\ e_L^* &= \frac{(a_L^*)^2}{\gamma(\gamma - 1)} \\ \rho_L^* &= \frac{P^*}{e_L^*(\gamma - 1)} \end{aligned}$$

The relative wave velocity is simply $u_L - w_L = a_L$.

By solving the Riemann problem at each interface around a finite volume we can compute its evolution in time. At the end of each time step, after fluxes are exchanged between cells, the new values of mass, momentum and energy are evenly redistributed evenly over each computational cell and the process repeated. By forcibly redistributing the conserved quantities evenly over the cell the resulting flow can become smeared [106]. This is a common trait with all finite volume solvers.

4.4 Equilibrium Flux Method

The Equilibrium Flux Method was introduced by Pullin [80] as a solution method for the Euler equations. EFM is based on the observation that the Euler equations are moments of the Boltzmann equation [81]. The fluxes are derived from kinetic theory by assuming an equilibrium distribution of molecular velocities in each computational cell. The fluxes may be formally derived from the Boltzmann equation, but here we derive the fluxes as those which result from Pullin's Equilibrium Particle Simulation Method [80] in the limit of an infinite number of particles in each cell, no gradients of density within the cells, and small time δt.

Consider a typical cell surface S with unit normals $\widehat{n}$,$\widehat{p}$ and $\widehat{q}$ attached to the surface. The fluxes can be separated into two parts, f^+ and f^-, corresponding to the flow of molecules across the surface in the positive and negative f-direction. Let the velocity of a molecule be denoted by $\vec{v}$, with components $v_n = \vec{v}.\widehat{n}$, $v_p = \vec{v}.\widehat{p}$, and $v_q = \vec{v}.\widehat{q}$. Let:

$$Q = \left[\, m,\ mv_n,\ mv_p,\ mv_q,\ m\left(\frac{1}{2}\vec{v}.\vec{v} + e_{st}\right)\right] \tag{4.10}$$

Note that $\frac{1}{2}\vec{v}.\vec{v}$ is the specific translational energy of the molecule and e_{st} is the specific energy of molecular structure, such as rotational, vibrational or electronic energy.

The flux of Q across the surface may be evaluated if the distribution functions for molecular velocities on either side of the surface, $g^+ = n_L f_L$ and $g^- = n_R f_R$, are known. Here, n is the number density and $f(\vec{v}, e_{st})d\vec{v}de_{st}$ gives the fraction of molecules with velocity in the range and energy of molecular structure in the range . The subscript L and R denote conditions on the left and right of the surface respectively; $\widehat{n}$ points from left to right. The net flux is:

$$F_Q = F_Q^+ + F_Q^- \tag{4.11}$$

Here:

$$F_Q^+ = \int_0^\infty \int_{-\infty}^\infty \int_{-\infty}^\infty \int_\infty^0 Q_g^+ v_n dv_n dv_p dv_q de_{st} \tag{4.12}$$

is the flux arising from molecules traveling from the left of the surface to the right. The flux arising from molecules traveling from the right side to the left side is:

$$F_Q^- = \int_0^\infty \int_{-\infty}^\infty \int_{-\infty}^\infty \int_\infty^0 Q_g^- v_n dv_n dv_p dv_q de_{st} \tag{4.13}$$

If the molecular velocities on either side take on the equilibrium distribution, Equation (4.12) can be evaluated as:

$$F_Q^+ = W_L \begin{bmatrix} \rho v_n \\ \rho v_n v_n + \rho RT \\ \rho v_n v_p \\ \rho v_n v_q \\ \rho v_n \left(\frac{1}{2}\vec{v}.\vec{v} + C_p T\right) \end{bmatrix} + D_L \begin{bmatrix} \rho c_m \\ \rho c_m v_n \\ \rho c_m v_p \\ \rho c_m v_q \\ \rho c_m \left(\frac{1}{2}\vec{v}.\vec{v} + \frac{1}{2}(\gamma+1)C_v T\right) \end{bmatrix} \tag{4.14}$$

i.e.

$$\begin{aligned} W_L &= \tfrac{1}{2}\left[1 + \operatorname{erf}\left(S_n\right)\right]_L \\ D_L &= \tfrac{1}{2\sqrt{\pi}} \exp\left(-S_n^2\right)_L \\ S_n &= \left(v_n / c_m\right)_L \\ c_m &= \sqrt{2RT}_L \end{aligned} \tag{4.15}$$

C_p and C_v are the specific heats at constant pressure and volume respectively, and $\gamma = \frac{C_p}{C_v}$. Equivalently, Equation (4.13) can be evaluated as:

$$F_Q^- = W_R \begin{bmatrix} \rho v_n \\ \rho v_n v_n + \rho RT \\ \rho v_n v_p \\ \rho v_n v_q \\ \rho v_n \left(\frac{1}{2}\vec{v}.\vec{v} + C_p T\right) \end{bmatrix} + D_R \begin{bmatrix} \rho c_m \\ \rho c_m v_n \\ \rho c_m v_p \\ \rho c_m v_q \\ \rho c_m \left(\frac{1}{2}\vec{v}.\vec{v} + \frac{1}{2}(\gamma+1)C_v T\right) \end{bmatrix} \tag{4.16}$$

I.e

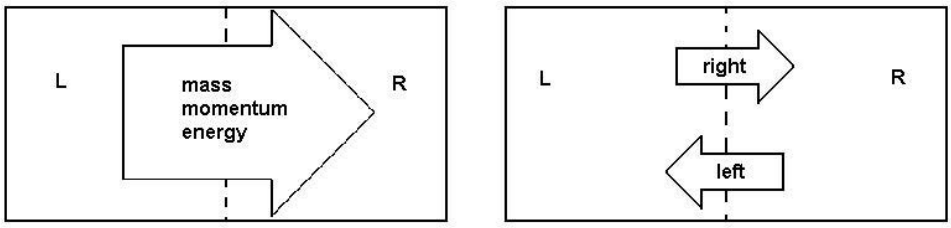

Figure 4.5: Cell interface between cells L and R with a net fluxes of mass, momentum and energy traveling from cell L to R.

$$\begin{aligned} W_R &= \tfrac{1}{2}\left[1 - \operatorname{erf}\left(S_n\right)\right]_R \\ D_R &= -\tfrac{1}{2\sqrt{\pi}} \exp\left(-S_n^2\right)_R \\ S_n &= \left(v_n / c_m\right)_R \\ c_m &= \sqrt{2RT}_R \end{aligned} \tag{4.17}$$

For moderate and low values of Mach number, EFM suffers greatly from numerical diffusion which can smear out contact discontinuities [81]. Macrossan has observed [62] that this numerical dissipation vanishes as the Mach number increases. Being a member of the Flux Vector splitting (FVS) schemes, EFM is robust in its capture of strong shock and rarefaction waves.

4.5 Van Leer's Method

The concept behind split fluxes is displayed in Figure 4.5. We will assume here that all flows remain subsonic and therefore a left and right moving flux is always present. The net flux moving from cell L to cell R is the difference of the flux moving left from cell R and the flux moving right from cell L. The right moving mass flux is

$$m_r = \rho_L a_L \left[\frac{1}{2}(M_L + 1)\right]^2 \tag{4.18}$$

where a is the speed of sound, ρ is the density and M is the Mach number, with subscript L indicating conditions in cell L. The subscript r indicates the flux is moving right. The flux moving left from cell R is

$$m_l = \rho_R a_R \left[\frac{1}{2}(-M_R + 1)\right]^2 \tag{4.19}$$

As long as the flow is subsonic, the net mass flux (per unit time per unit area) is therefore

$$mass = m_r - m_l = \rho_L a_L \left[\frac{1}{2}(M_L + 1)\right]^2 - \rho_R a_R \left[\frac{1}{2}(-M_R + 1)\right]^2 \tag{4.20}$$

The right moving momentum flux is

$$p_r = \rho_L a_L^2 \left[\frac{1}{2}(M_L + 1)\right]^2 \left[\frac{(\gamma - 1)}{\gamma} M_L + \frac{2}{\gamma}\right] \tag{4.21}$$

where γ is the ratio of specific heats. The effective momentum flux moving left from cell R is

$$p_l = -\rho_R a_R^2 \left[\frac{1}{2}(-M_R + 1)\right]^2 \left[-\frac{(\gamma - 1)}{\gamma} M_R + \frac{2}{\gamma}\right] \tag{4.22}$$

The net momentum (per unit time per unit area) is

$$\begin{aligned} mom &= p_r - p_l \\ &= \rho_L a_L^2 \left[\tfrac{1}{2}(M_L + 1)\right]^2 \left[\tfrac{(\gamma-1)}{\gamma} M_L + \tfrac{2}{\gamma}\right] + \rho_R a_R^2 \left[\tfrac{1}{2}(-M_R + 1)\right]^2 \left[-\tfrac{(\gamma-1)}{\gamma} M_R + \tfrac{2}{\gamma}\right] \end{aligned} \tag{4.23}$$

Following Van Leer [105], the energy flux moving from right from cell L can be written in terms of the previously found momentum and mass fluxes:

$$e_r = -\left(\frac{\gamma^2}{2(\gamma^2 - 1)}\right) \frac{p_l^2}{m_l} \tag{4.24}$$

The left moving flux is similarly

$$e_l = -\left(\frac{\gamma^2}{2(\gamma^2 - 1)}\right) \frac{p_r^2}{m_r} \tag{4.25}$$

The net energy flux (per unit energy per unit time) is

$$eng = e_r - e_l = \left(\frac{\gamma^2}{2(\gamma^2 - 1)}\right) \left[\frac{p_r^2}{m_r} - \frac{p_l^2}{m_l}\right] \tag{4.26}$$

Equations 4.20, 4.23 and 4.26 can be used to calculate the decrease in mass, momentum and energy from cell L and the increase in mass, momentum and energy in cell R per unit area per unit time. In the case of flows where local Mach numbers

4.6 Particle Flux Method

Particle Flux Method (PFM) is a modification to Pullin's EPSM [80] proposed by Macrossan *et al.* [70] with the goal of emulating a conventional continuum solver through the use of a direct simulation approach. In PFM, fluxes of mass, momentum and energy are calculated through the creation of simulation particles in each cell at each time step which are then moved in free flight into destination cells. The quantities of mass, momentum and energy each particle carries is then added to the destination cell, and the particle then discarded. Fluxes which are

transfered are not limited to cells sharing an interface - the particles are free to move to any cell reachable in a given time step.

PFM is a finite volume method in that the flow is subdivided into finite volumes (cells), fluxes exchanged between neighbouring cells and the resulting mass, momentum and energy evenly redistributed around each computational cell. Conservation is forced through the complete discretisation and distribution of the conserved quantities into a fixed number of simulation particles which are precisely tracked. Because of this, PFM does not suffer from the effects of direction decoupling. It does, however, suffer from the smearing of conserved quantities over cell volumes that other finite volume solvers suffer [106].

The general PFM procedure for any gas is described by:

1. The flow field is divided into a finite number of computational cells.

2. At each time step, a new set of simulation particles is created. The variance and mean of the velocity distributions are scaled to ensure complete capture of momentum and energy. The mass in the source cell is evenly distributed among the simulation particles, as is any additional structural internal energy. The particle locations are distributed evenly around the cell. The same set of simulation particles is used in each cell, each time with scaled velocities to match the local cell conditions.

3. Each of these particles is moved in time $\mathbf{x}^* = \mathbf{x} + \mathbf{v}\Delta t$ where x is a vector of positions $x = [x, y, z]$, v is a vector of velocities $\mathbf{v} = [v_x, v_y, v_z]$ and Δt is the time step.

4. The mass, momentum and energy each particle carries is added to that of the destination cell.

5. Assuming even distribution of flow quantities over each cell, the revised state (i.e. temperature, bulk velocity) is calculated.

6. The process is repeated for each cell, until a certain time has been reached or the flow is steady.

The method is fundamentally different to that of EPSM because no density gradients are maintained and particles are created and destroyed at every time step. Effectively, PFM is simply a method for the calculation of mass, momentum and energy fluxes between cells and can replace a conventional flux solver. Another important feature of PFM is its ability to calculate fluxes of mass, momentum and energy in regions of low density. In a conventional direct solver, regions of relatively low density (when compared to other regions in the flow field) are represented by relatively few simulation particles. When using PFM, the number of simulation particles used is identical for each cell.

Because the local conditions are assumed to be in perfect thermal equilibrium, the corresponding collision time is theoretically very small. EFM, EPSM and PFM suffer from numerical

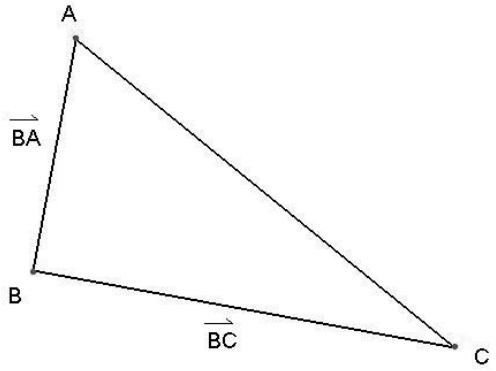

Figure 4.6: Typical triangular cell used in PFM simulation.

dissipation because particles are often carried further in free flight than local conditions would otherwise allow [70]. This leads to the artificial thickening of features such as shock waves. The smearing that occurs due to PFM's finite volume nature also contributes toward this. More advanced direct simulations such as DSMC will suffer from similar dissipation if the time step used for free molecular flight is larger than the mean collision time, but are not affected by the smearing which occurs in finite volume solvers because information regarding particle location is explicitly maintained.

Another advantage of PFM is its ease of implementation on unstructured grids. If the solver uses triangular cells, then the PFM simulation particles can be distributed evenly around the cell through the following procedure:

1. Calculate R_1, a vector of random numbers $R_1 = [r_1, r_2 ... r_N]$ where there are N simulation particles.

2. Calculate R_2, a new vector of random numbers $R_2 = [r_1^{0.5}, r_2^{0.5} ... r_N^{0.5}]$.

3. Given the point B on the triangle (See Figure 4.6) and vectors $\vec{BC}$ and $\vec{BA}$, the equation for the new particle locations are:

$$\mathbf{X} = R_2 \left(R_1 \vec{BA} + (1 - R_1) \vec{BC} \right) + B \tag{4.27}$$

   where $\mathbf{X}$ is a 2D vector of position $X = [x, y]$.

A similar routine can be used for 3 dimensional simulations. Particles are then moved in free flight (i.e. straight lines) to their destination region. This region can be found by using the line equation created by the particle in flight and calculating possible intercepts across triangle surfaces.

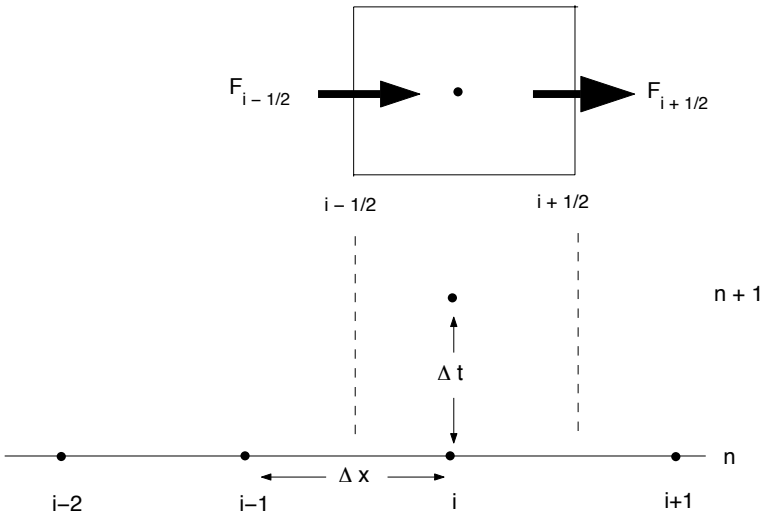

Figure 4.7: Finite volume discretisation of a one dimensional partial differential equation. The region is divided into a large number of volumes with cell centers $i-1, i, i+1$ separated by distance Δx. The flux of conserved quantities across volume interfaces is F. Time is progressed in steps of Δt with new fluxes F calculated at each step.

4.7 High Resolution Schemes

High resolution schemes are defined by Roe [89] as a class of algorithms designed for solving problems involving partial differential equations in which wave propagation is an important feature. Many wave propagation problems are governed by equations such as the Euler or Navier-Stokes equations, commonly known as the conservation equations. Using the finite volume approach, these equations are typically solved by dividing space into control volumes and tracking fluxes of conserved quantities (i.e. mass, momentum and energy) across cell surfaces. In order to increase the order of accuracy above first order, the flux calculator used to estimate the fluxes of conserved quantities must be nonlinear [89, 106].

Roe [89] demonstrates some of the issues of solving the conservation equations by examining the inviscid Burger's equation:

$$\frac{\partial u}{\partial t} + \frac{\partial (1/2)u^2}{\partial x} = 0 \tag{4.28}$$

where u is a conserved quantity and the equivalent flux function $f(u) = (1/2)u^2$. Burgers equation is often presented as a purely abstract problem which allows us to examine the quality of finite volume solvers which maintaining wave propagating qualities [89]. The finite volume discretisation for this problem is shown in Figure 4.7. The space in x is divided onto finite volumes, or in this case lengths of width Δx. Time is discretised into steps of Δt. The conserved quantity u is tracked in time and space through the exchange of fluxes of conserved quantity F between finite volumes.

Therefore, using the notation described in Figure 4.7, the value of the conserved quantity

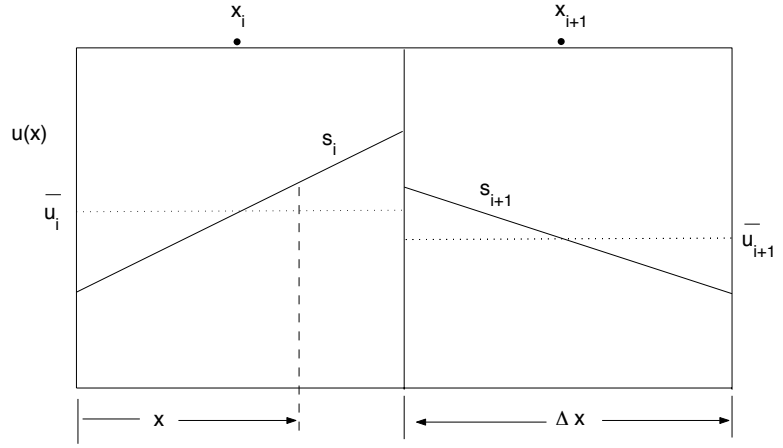

Figure 4.8: Finite volume discretisation with conserved quantity u linearly distributed through the cell volume. The mean value $\overline{u_i}$ in cell i is the value of the conserved quantity u at the cell center location x_i. The slope of the conserved quantity in cell i is s_i.

u at location i at time $n+1$ is equal to:

$$u_i^{n+1} = u_i^n - \frac{\Delta t}{\Delta x}\left[F_{i+1/2}^{n+1/2} - F_{i-1/2}^{n+1/2}\right] \tag{4.29}$$

where $F^{n+1/2}i + 1/2$ is the flux evaluated at the interface between volumes i and $i+1$ and at time $n + 1/2$. A simple explicit scheme can be created where the fluxes F is evaluated using nearby states at time n. If the flux is a function only of states at time n, the method will be first order accurate in time. An example calculation of F has already been presented in Equation 4.7 and is:

$$F_{i+1/2}^{n+1/2} = \frac{1}{2}\left[(u^2/2)_i^n + (u^2/2)_{i+1}^2\right] \tag{4.30}$$

This is easily recognised as a central difference and possesses a second order degree of accuracy in space. However, this is unstable without the addition of numerical dissipation [89]. The concept of upwinding is to use information taken from the flow in the direction the flow is coming from. In this case, the estimate of the flux may be calculated as:

$$F_{i+1/2} = \begin{cases} u_i^2/2, \mathrm{u_i} + \mathrm{u_{i+1}} > 0 \\ u_{i+1}^2/2, \mathrm{u_i} + \mathrm{u_{i+1}} < 0 \end{cases} \tag{4.31}$$

This is equivalent to Godunov's upwind scheme [28] and is first order accurate. It is important to acknowledge that these fluxes are all based on the assumption of the constant nature of the conserved quantity u on either side of each interface [89]. The value of the conserved quantity u up to (but not on) the interface is its cell centered value u_i.

As long as the conserved quantity is distributed in the cell volume in a uniform fashion, the method cannot exceed first order accuracy in space. The next logical step to overcome Godunov's dilemma is to assume some distribution of the conserved quantities in a cell. The simplest case, first proposed by Van Leer [104] and is demonstrated in Figure 4.8. In this case,

the conserved quantity u is reconstructed using a linear relationship with location:

$$u_i(x) = \overline{u_i} + s_i(x - \overline{x_i}) \tag{4.32}$$

where $\overline{u_i}$ is the mean value of u over the cell i, s_i is the gradient of the conserved quantity, x is an arbitrary location within cell i and $\overline{x_i}$ is the average of x over the cell or simply the location of the cell center. The gradient s_i can be calculated from the surrounding properties. The simplest form of these gradients is the finite difference approximations:

$$\begin{aligned} s_i^- &= \frac{1}{\Delta x}(\overline{u_i} - \overline{u_{i-1}}) \\ s_i^+ &= \frac{1}{\Delta x}(\overline{u_i + 1} - \overline{u_i}) \end{aligned}$$

where s_i^- and s_i^+ are the forward and backward differences. If the effective value of s_i was calculated as the average of these, the resulting expression would be linear and hence results in oscillations in the solution [89]. In order for the solution to be of higher order, a nonlinear average of s^+ and s^- must be taken. Since we have the freedom to chose the manner in which the conserved quantity u is distributed through the cell volume we can also chose, in any manner we wish, the form of s_i. The only constraint is that the mean value of $u_i(x)$ is always equal to $\overline{u_i}$, which is the case for Equation 4.32.

The creation of the effective slope s_i^* is a function of the value of u of the surrounding cells. The effective state at the interface can be calculated using the effective slope:

$$\begin{aligned} u_{i+1/2}^L &= u_i + \frac{\Delta x}{2} s_i^* \\ u_{i+1/2}^R &= u_{i+1} - \frac{\Delta x}{2} s_{i+1}^* \end{aligned}$$

Various attempts have been made to evaluate the effective slope s^*:

- Godunov - $s^* = 0$. The traditional Godunov solver assumes uniform conditions in a cell.
- Fromm's Method - Central difference slope.

$$s^* = \frac{u_{i+1} - u_{i-1}}{2\Delta x} \tag{4.33}$$

- Beam-Warming Method - Upwind difference slope.

$$s^* = \frac{u_i - u_{i-1}}{\Delta x} \tag{4.34}$$

- Lax-Wendroff Method - Downwind difference slope.

$$s^* = \frac{u_{i+1} - u_i}{\Delta x} \tag{4.35}$$

- Min-Mod slope limiter

$$s^* = \text{minmod}(\frac{u_{i+1} - u_i}{\Delta x}, \frac{u_i - u_{i-1}}{\Delta x}) \tag{4.36}$$

- Monotonized central difference slope limiter (MC limiter)

$$s^* = \text{minmod}[\frac{u_{i+1} - u_{i-1}}{2\Delta x}, \text{minmod}(2\frac{u_{i+1} - u_i}{\Delta x}, 2\frac{u_i - u_{i-1}}{\Delta x})] \tag{4.37}$$

These slopes are then used to provide a better estimate at the interfaces between cells, which are in turn used to calculate fluxes across cell surfaces. It is important to note that once the values at the interfaces have been established, the flux calculators (i.e. Van Leer, Roe solver, EFM) assume the values behind the interface to be uniform. Therefore, the flux calculated is only correct at the instant the flux is to be calculated - for any finite value of time step, the conditions driving the reconstructed fluxes across a surface also change with time.

Chapter 5

Computational solutions to the Boltzmann Equation

5.1 Introduction and Summary

When the gas is considered dilute and distance between molecules sufficiently large, a complete description of the characteristics of the flow is available in the form of the Boltzmann equation. When deviations from equilibrium are small, the Navier Stokes equations represent an approximation to the Boltzmann equation. In theory, if a solver is capable of correctly solving the Boltzmann equation, then that solver would be capable of modeling any flow so long as the dilute gas assumption holds.

While the Navier Stokes equations are an accurate approximation of the Boltzmann equation at small deviations from equilibrium, they are not valid in rarefied flows. In rarefied conditions, the molecular collision rate is too low to maintain small deviations from equilibrium. This is also the case in high speed flows, where the local flow time is on a scale comparable to the local collision time. In these conditions, large deviations from equilibrium are possible. In such conditions, the molecular nature of a gas must be considered and a method which solve the Boltzmann equation itself is desired.

The most popular engineering method available in recent times is the Direct Simulation Monte Carlo (DSMC). In this approach, the motion of molecules is tracked deterministically while the interaction of particles is handled in a stochastic fashion. Each tracked molecule, referred to as a simulation particle, represents a very large number of real particles. The flow is divided into finite regions, or cells, in a similar fashion to finite volume methods. Time is divided into discrete intervals and at each time step, particles are moved in free molecular flight. Simulation particles are then sorted into the computational cells (and sometimes sub cells) and are collided with each other using a stochastic approach to determine the outcome of the collision. Through collisions, energy in the various degrees of freedom is exchanged. With enough collisions, the energy is equally shared amongst the available degrees of freedom and the gas reaches equilibrium.

Another more direct approach to solving the Boltzmann equation is known as the direct Boltzmann or model Boltzmann approach. In this technique, the Boltzmann equation is at-

tacked directly through the discretisation of velocity and translational space and time. Rather than simulating particles directly, such methods discretise the velocity distribution functions and calculate incremental changes in time and space. These are typically very computationally expensive and to obtain accurate results for engineering flows requires significant computational effort. Such methods are covered in detail following the investigation of the Direct Simulation Monte Carlo.

5.2 Direct Simulation Monte Carlo

The Direct Simulation Monte Carlo, or DSMC, was created by Bird as a method for investigating the translational relaxation of a monatomic gas. Since this time, DSMC has been used for a large range of engineering applications including, but not limited to:

- Hypersonic flows, such as orbital reentry,
- The simulation of plasma reactors for use in microelectronics manufacture,
- Drag calculation of satellites in low Earth orbit,
- Calculation of the structure of jet plumes,
- Analysis of shock wave structure,
- Astrophysical calculation.

The scope of this work only requires a brief description of the DSMC method. Firstly, the implementation of a typical DSMC method is outlined. Each of the steps in a DSMC simulation, known as phases, is then described.

A standard DSMC code is implemented as follows:

1. Initialisation phase - Initialise particle velocities and locations using desired initial conditions, create flow geometry and mesh.
2. Free flight phase - move simulation particles through collisionless flight by Δt and enforce boundary conditions. This phase will often involve the creation or deletion of simulation particles.
3. Determine which cell (and sub cell, if applicable) each simulation particle resides in.
4. Sample the flow field and determine desired quantities, i.e. temperature, mean speed.
5. Collision phase - perform a calculated number of collisions in each cell. The outcome of these collisions are probabilistic, moving the conditions in the cell from the initial state toward the equilibrium state.

6. Repeat from (2) for a desired number of steps until the simulation is completed.

7. Save results for post-processing.

5.2.1 Flow phase decoupling

The DSMC method relies heavily upon the assumption that the flow can be split into a 'free flight phase' and a 'collision phase'. During the 'free flight phase' of the direct simulation, molecules are moved to new location x^* from the old location x by distance $v_x \Delta t$, where v_x is the particle velocity in the x direction. In a typical DSMC simulation, no external forces are applied to the particle during this time. Thus simulation particle locations change but their velocities remain untouched. The 'collision phase' is where simulation particles exchange energy through stochastic collisions. During this time, particles are randomly selected for collision and the results of the collision are a change in the simulation particles component velocities and internal energies. During this phase, the velocities are changed but particle locations remain untouched.

For this phase decoupling to remain valid, the time step separating the two phases must be smaller than the mean collision time. Garcia and Wagner [27] and Hadjicontantinou [35, 36] show that the transport coefficients depend on Δt and the error in these coefficients is of the order $O(\Delta t)^2$. Typically, the time step is $\Delta t \approx \tau/3$.

The cell size Δx dictates the size of the region from which properties are sampled. These properties control the calculated collision rate, and thus affect all particles within that region. In many DSMC simulations, collision partners are selected from anywhere within the cell (or sub cell) under the assumption that in a time step Δt it is reasonable to assume two particles in this region are capable of collision. Thus, the cell size must be smaller than the mean free path λ. In regions where the flow gradients are small or zero, such a restriction is not required. Where the characteristic length of these gradients approach λ, the cell size must also. Alexander [1, 2] shows that the effective viscosity and heat transfer present are functions of the cell size, hence great care must be taken to ensure the cells are sized correctly.

5.2.2 Movement phase

During the movement phase, particles undergo free flight and are untouched by external forces. In a 3 dimensional cartesian coordinate system, the vector of new particle locations $x^* = [x^*, y^*, z^*]$ is given by:

$$\mathbf{x}^* = \mathbf{x} + \mathbf{v}\Delta t \tag{5.1}$$

where $x = [x_o, y_o, z_o]$ is the vector of locations prior to time stepping and $v = [v_x, v_y, v_z]$ is the vector of particle velocities.

5.2.3 Collision phase

During the collision phases, molecules undergo probabilistic collisions which results in a change of their velocities and internal energy. Since particle locations are preserved during this phase, a DSMC solver maintains information about the flow field by maintaining a density distribution in the cell. Any velocity gradients resulting from the pre-collision distributions will tend to vanish as more collisions occur. In the no time counter (NTC) method proposed by Bird [14], the number of simulation particle pairs to test for collision are:

$$N_{pairs} = nN(\sigma g)_{max}\Delta t/2 \tag{5.2}$$

This number will result in a real number with a decimal component. The integer component of N_{pairs} can be used in the current time step, with the decimal component kept for the next time step and added to subsequent N_{pairs} calculations [10]. These collision pairs are selected randomly from a cell (or sub cell, if applicable). For each pair, the value of σg is calculated. A random fraction R_f is then calculated, and the chance of the pair of selected simulation particles colliding is:

$$R_f < \frac{\sigma g}{(\sigma g)_{max}} \tag{5.3}$$

The value of $(\sigma g)_{max}$ is updated as the simulation progresses. For a steady flow simulation, the initial value of $(\sigma g)_{max}$ is unimportant.

There are many other alternative schemes for the calculation of the collision phase of a DSMC simulation. Macrossan [69] proposed a scheme which uses temperature dependent molecular cross sections. This method, named μ-DSMC, can use an arbitrary viscosity law (or data provided by the user) to simulate rarefied flows. The Chapman-enskog viscosity [19] for hard sphere molecules is:

$$\mu = \frac{5m\pi^{1/2}}{16}\frac{(RT)^{1/2}}{\sigma} \tag{5.4}$$

Assuming collision pairs are selected when $R_f > g/g_{max}$, Macrossan calculated the collision frequency in terms of the viscosity, given by:

$$\nu = \frac{5\pi}{16}\rho\overline{g}\frac{(R\overline{T})^{1/2}}{\mu(\overline{T})} \tag{5.5}$$

where $\overline{g}$ is the mean relative speed between collision partners and $\overline{T}$ the mean temperature. Macrossan tested this by performing Couette and 1D shock wave simulations [69]. The method was then expanded to include more complete molecular models. However, these modifications to the method fall outside of the scope of this work and will not be investigated.

Another development by Macrossan was the creation of a DSMC method which does not rely upon a collision pair acceptance-rejection scheme. The method, named ν-DSMC, calculated the collision rate based on the Maxwell variable hard sphere model [66]. Since all collision pairs

(in a cell or sub cell) were assumed equally likely no testing was performed. The collision rate calculated for Maxwellian molecules is:

$$\nu = \frac{2\rho RT}{\mu(\overline{T})} \tag{5.6}$$

This collision rate is slightly higher than other existing DSMC collision rates. This is logical since a higher collision rate would be required to compensate for collisions which occur with little effect on the outcome of the simulation, such as collision pairs with low relative velocities [66]. The ν-DSMC results were found to differ only slightly from those of conventional DSMC.

5.2.4 Boundary conditions

The treatment of boundaries in a DSMC simulation can be separated into stream boundaries and interactions of particles with solid boundaries. For the direct simulation of inflow boundaries, this involves the creation of new simulation particles with qualities taken from the desired free stream conditions. These upstream boundaries need to be far enough away from obstructions or flow features to ensure this selection of simulation particle is valid. This generation can be performed in ghost cells outside of the simulation region. Simulation particles are moved in time and those which enter the simulation region are kept while others a ignored and removed.

For downstream boundaries, many hypersonic flow problems have a resulting high velocity exit condition. In such cases, exiting molecules are simply removed. This is valid in many cases since the chance of a molecule generated downstream having sufficient velocity to reenter the flow is extremely small. For this condition to be valid, the downstream flow boundaries must be sufficiently far enough away from obstacles to ensure large exit velocities. In the case of subsonic flow, a similar procedure to that of the inflow routine can be used. These methods are valid for both steady and unsteady flow.

Reflections of solid surfaces are typically treated as either specular or diffuse in nature. A specular reflection is a completely elastic collision with the surface. The velocity component normal to the wall is reversed while the other parallel components remain unchanged. This is demonstrated in two dimensions in Figure 5.1. In specular reflections, no energy is lost by the incident particle and thus no heat transfer across the surface occurs. In engineering applications this is often deemed as unrealistic since incident molecules are often trapped on the surface for a period of time and tend to take on properties similar to that of the surface.

Diffuse reflections better take this property into account. All incident velocities are completely destroyed upon contact with the surface. Simulation particles are then given newly generated velocities. The normal velocity assigned to a reflected simulation particle is given by:

$$v_n^* = (-\ln(R_f))^{1/2}(\frac{2kT_{wall}}{m})^{1/2} \tag{5.7}$$

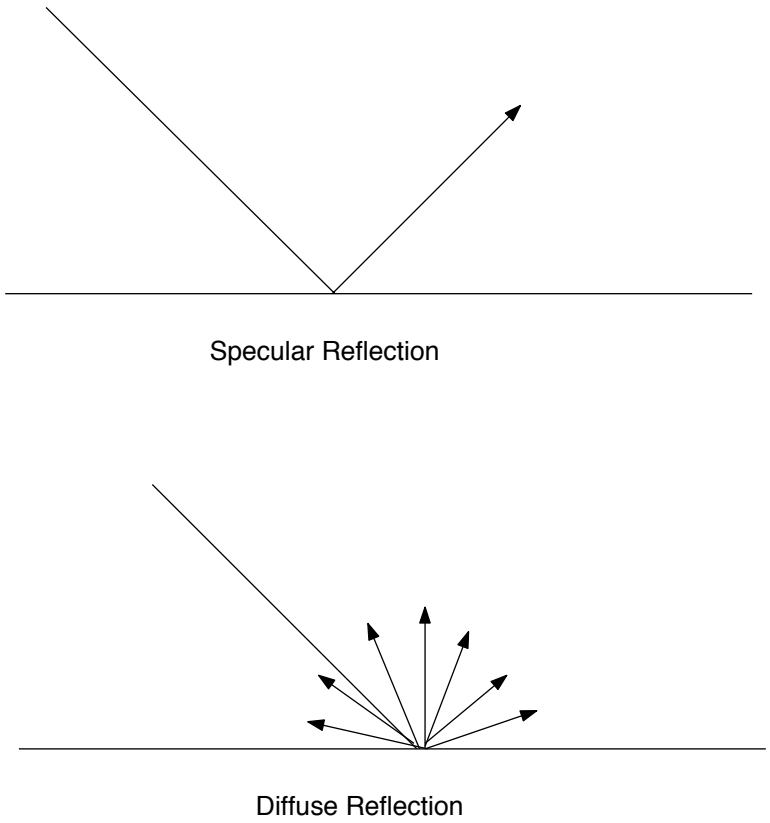

Figure 5.1: Specular and diffuse reflective models for direct simulations.

The velocity components parallel to the surface are generated from the Maxwell-Boltzmann equilibrium distribution function using the temperature of the wall T_{wall} and a zero bulk velocity. This ensures a 'non slip' boundary condition so the bulk velocity close to the wall surface tends to zero. For many engineering flows, this reflective condition provides acceptable results.

5.3 Equilibrium Particle Simulation Method

Equilibrium Particle Simulation Method (EPSM) was proposed by Pullin [80] as a continuum gas solver and represents the infinite collision limit of DSMC. Technically, EPSM only represents a solution to the Boltzmann equation when the distribution functions are assumed to be those of the Maxwell-Boltzmann equilibrium distribution functions [107]. In this case, the Boltzmann equations are well known to reduce to the Euler equations. However, due to its similarities with the DSMC proceedure the method is discussed here.

The collision process critical to DSMC simulations is replaced by a procedure in which all particles within a cell have their velocities forced to an equilibrium velocity as described by the Maxwell-Boltzmann velocity distribution. These velocities are randomly generated in each cell for each time step. In order to conserve momentum and energy exactly, the pre-collision and the post-collision (randomly generated) velocity distributions must have identical an identical mean and variance. This is done through the scaling procedure employed by Montanero *et. al.*

[74] and later used by Macrossan [68].

For a simple monatomic gas, the EPSM process is as follows:

1. N simulation particles are distributed through the flow geometry with scaled, randomly generated velocities to suit the initial conditions.

2. Each of these particles is moved in time $X^* = X + V\Delta t$ where X is a vector of positions $X = [x, y, z]$ and V is a vector of velocities $V = [u, v, w]$ and Δt is the time step.

3. These particles are indexed into cells.

4. The state in each cell is calculated through kinetic theory relations.

5. Without adjusting particle locations, new particle velocities are assigned to each particle according to the state the cell is in. For a monatomic gas, the variance and mean of the new velocity distributions must be identical to the original.

6. The process is continued until the desired time has been reached, or the flow is steady. Due to the statistical scatter encountered with any direct simulation, averages may be required over various time steps to obtain a smooth solution.

EPSM has been used in various hybrid codes [68, 21, 22, 115] and has been used to solver the Euler equations. The strength of such a method lies in the similarities of implementation. In regions deemed to be in thermal equilibrium the velocities of particles are assigned from the equilibrium velocity distribution function while intermolecular collisions are employed elsewhere.

5.4 Model-Boltzmann solver

An alternative method to the solution of the Boltzmann equation is to attempt to directly solve the differential equation itself. The Boltzmann equation and its collision term is:

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \frac{\partial f}{\partial \mathbf{x}} + \mathbf{F} \cdot \frac{\partial f}{\partial \mathbf{v}} = \left[\frac{\partial f}{\partial t}\right]_{coll} \tag{5.8}$$

$$\left[\frac{\partial f}{\partial t}\right]_{coll} = \int_{-\infty}^{\infty} \int_{0}^{4\pi} \left(f^* f_1^* - f f_1\right) c_r \sigma \, \mathrm{d}\Omega \, \mathrm{d}\mathbf{c}_1 \tag{5.9}$$

This equation is very difficult to solve directly. A more conventional approach is to attempt to solve the Bhatnagar-Gross-Krook (BGK) model equation [11] (also proposed independently by Weylander [109]) which replaces the collision term of the Boltzmann equation by a simpler source term which retains the important features of the original. The BGK equation is:

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \frac{\partial f}{\partial \mathbf{x}} = \nu_M \left(f_M - f\right) \tag{5.10}$$

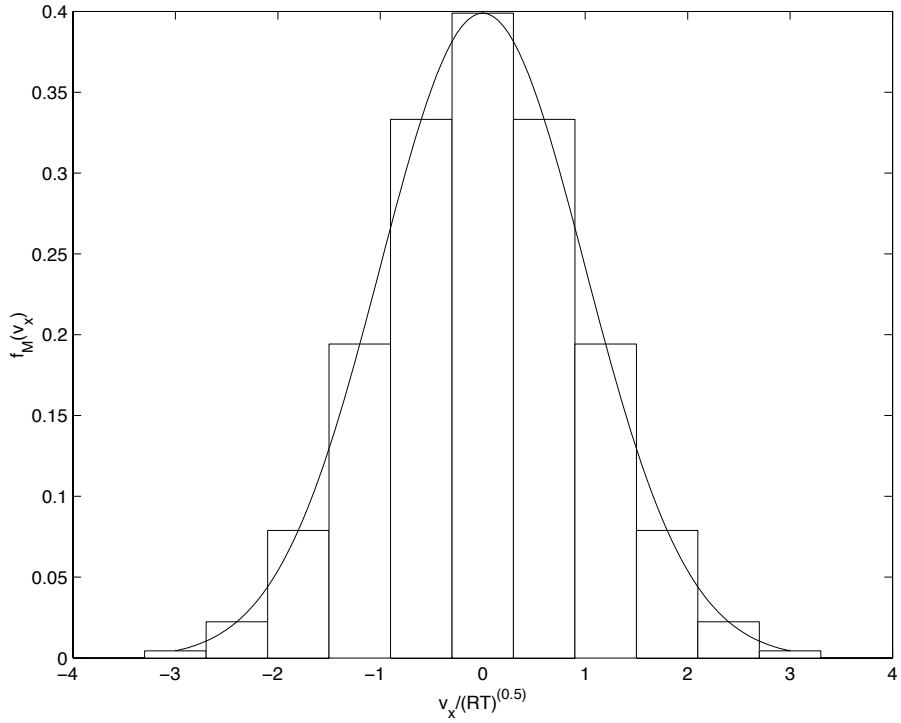

Figure 5.2: Discretisation of velocity space used by BGK solvers. The bulk velocity calculated by from the discretised distribution is $\overline{v} = 3.98e - 17$ with $s = 0.995$.

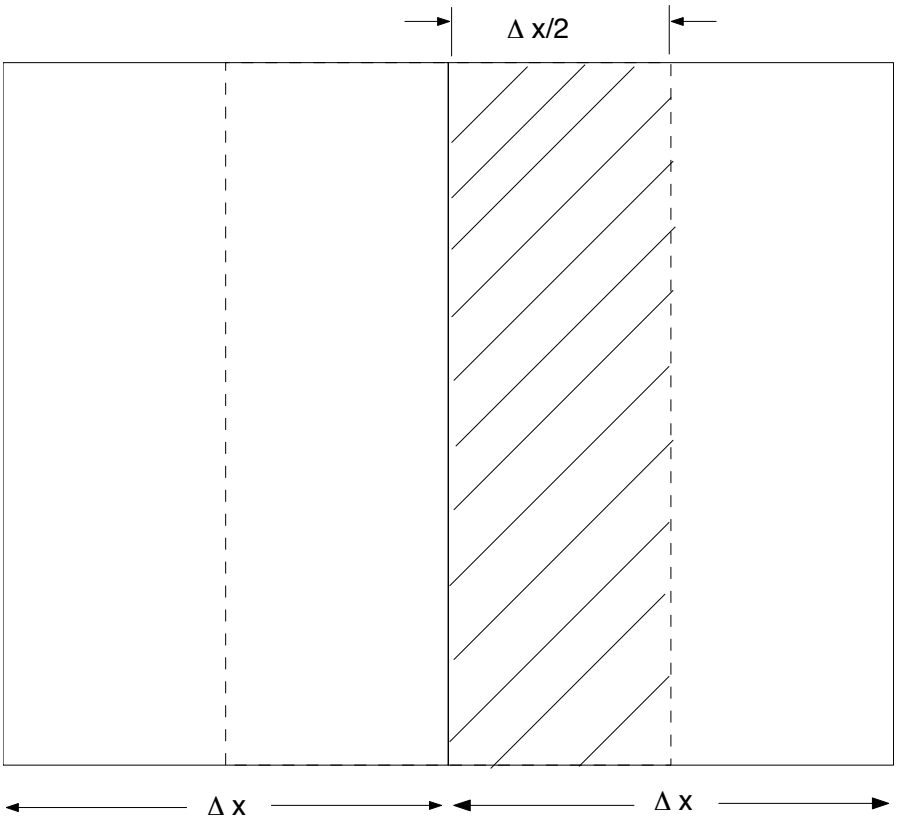

Figure 5.3: The linear advection of mass by ray tracing in a BGK solver. In the case demonstrated, the selected velocity bucket has sufficient velocity to ensure that half of the uniformly distributed mass in the source cell will reach the destination cell.

The method used to solve the BGK equation requires discretisation in time, physical space and velocity space. The continuous velocity distribution functions and energy distribution functions are discretised into a finite number of 'velocity buckets' of velocity width dv. The value of $f(v_x)$ for each bucket is calculated using the velocity at the centre of each bucket. Each velocity bucket is assumed to have this constant value of $f(v_x)$ over velocity space dv. A simple example is demonstrated in Figure 5.2 where the equilibrium distribution function for velocity in the x direction is discretised into 11 'velocity buckets'. Similar discretisations are required for all other degrees of freedom of the gas. The mean and the variance of the discretised velocities are used to calculate the mean velocity and temperature, therefore in an equilibrium gas the mean and the variance must be identical to the continuous equilibrium velocity distribution function.

The typical BGK solver employs flow phase splitting like that used by DSMC. The flow is split into a movement phase, governed by the advection component of the BGK equation, and a collision phase which uses the collision term in the BGK equation to approximate the effect of collisions. While many attempts have been made to solve the advection term of the BGK equation with a high degree of accuracy, this work will focus on the use of simple ray tracing to track movement of the gas. This procedure is shown in Figure 5.3 when restricted to a single dimension. The source cell (left) is assumed to have mass M which is assumed to be uniformly distributed across the cell. For a given velocity v_x, there is a fraction of mass in the cell to possess this velocity $f_M = f(vx)dv_x$. Of this fraction, there will be a smaller fraction which are able to move into the adjacent cell through free molecular flight. Figure 5.3 shows a selected velocity v_x which allows the fraction f_M to move half a cell width in the alloted time step. Therefore, since the mass is uniformly distributed, half of the fraction f_M will fall into the adjacent cell while the remaining half remains in the source cell. While this solution method is simplistic, there is an analytical equivalent in the limit of an infinite number of velocity buckets and an initial velocity distribution function equal to the Maxwell-Boltzmann equilibrium distribution function. This method is discussed in Section 8.

The effect of molecular collisions is to shift a non-equilibrium distribution function toward the equilibrium distribution function. A diagram showing a velocity distribution in non-equilibrium and its hypothetical equilibrium counterpart is shown in Figure 5.4. The simplified collision term states that the rate at which a particular velocity bucket approaches its equilibrium value is proportional to the difference between its pre-collision distribution and its final equilibrium distribution function. This difference, $(f_M(v_x) - f(v_x))$, is represented in Figure 5.4 as shaded regions. After fluxes are moved and pre-collision distributions generated, each velocity bucket is compared to its equilibrium value and relaxed toward it by an amount $\nu\Delta t(f_M(v_x) - f(v_x))$.

The advection and collision phase of the BGK solver needs to be performed for every possible combination of velocity buckets. Therefore, if the simulation of a gas with a single translational degree of freedom requires n computational effort, a gas with two translational

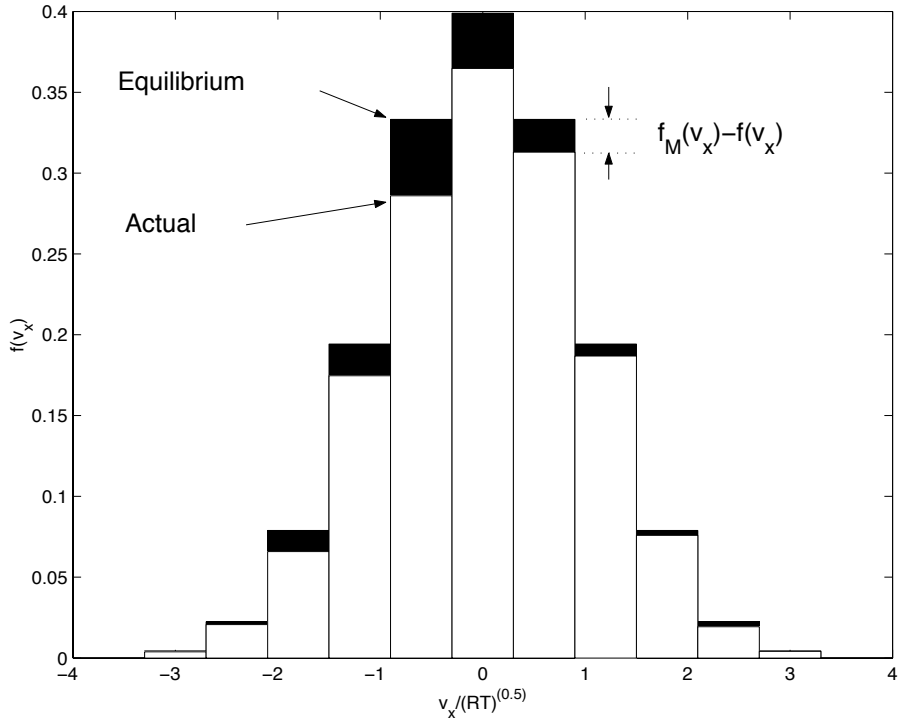

Figure 5.4: Hypothetical velocity distribution before collisions and the equilibrium distribution. The difference between these distributions is shaded.

degrees of freedom will require at least n^2 and three translation degrees of freedom requiring at least n^3. Therefore, if a three dimensional simulation using x velocity buckets was found to have too coarse a discretisation in velocity space and $2x$ buckets were employed, the new simulation would require $2^3 = 8$ times as long to solve.Thus, it is difficult to practically apply this method to real engineering problems due to its computational expense.

Chapter 6

Kinetic Theory based hybrid solvers

6.1 Introduction and Summary

Due to the computational expense associated with solving non-equilibrium flows, a large number of kinetic-theory based hybrid solvers have been developed. A large number of these hybrid solvers employ solutions to the Navier-Stokes equations in regions of thermal equilibrium while using a direct solver in regions of non-equilibrium. While these solvers have been shown to provide valuable results, the fundamental basis for each component of such hybrid solvers varies. This work will focus on the theory of the Unified Flow Solver (UFS), which describes any solver capable of capturing both equilibrium and non-equilibrium regions with each component derived from the same basic principles or theory, in this case the Kinetic Theory of gases. First we examine the particle only hybrid solvers, first introduced by Macrossan [68] and further investigated by Chen *et. al.* [21, 22] and Wu [115]. In such an approach, the Direct Simulation Monte Carlo (DSMC) is used in regions of non-equilibrium while the Equilibrium Particle Simulation Method (EPSM) is used in regions of thermal equilibrium. Following this, we will examine the family of kinetic theory based hybrid solvers where fluxes are analytically determined through integration of a given velocity distribution function over a boundary or otherwise determined through the use of a BGK solver.

6.2 Particle based hybrid solvers

The simulation of flows with both equilibrium and non-equilibrium regimes is of practical importance to modern engineering calculations. This is traditionally done with the use of DSMC in all regions. However, regions of thermal equilibrium are collision dominated; the use of DSMC in such regions is computationally expensive. In addition, the effective outcome of a large number of collisions is well known - the distribution of velocities in these regions will be equal to the Maxwell-Boltzmann distribution of velocities. Macrossan proposed a 'particle only' method which used the local breakdown parameter to separate regions of thermal equilibrium and non-equilibrium [68]. In this method, the equilibrium regions were solved with Pullin's EPSM [80] while DSMC was used in regions of non-equilibrium. The same approach has been

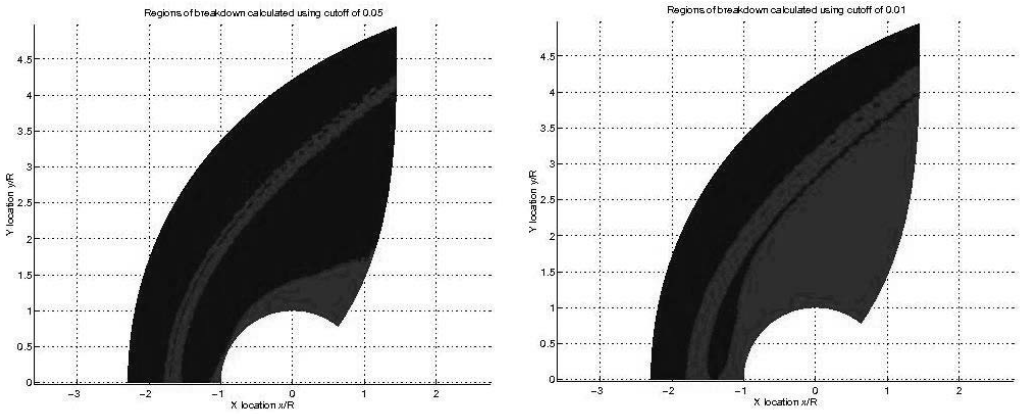

Figure 6.1: Regions of thermal equilibrium breakdown determined using gradients of mean free path used by Smith [97] as part of a hybrid EPSM/DSMC solver. Regions are shown with a breakdown parameter larger than 0.05 (Left) and 0.01 (Right). The flow is hypersonic over an infinitely long cylinder. Free stream temperature was 300K. M = 8, Kn = 0.016, $\gamma = 5/3$. The temperature at the wall was 300K. Steady time was 12ms, time step $\Delta t = 1.25e - 7$s.

used by various authors [22, 21, 97, 115]. The advantage of such an approach is the simple and rapid implementation while obtaining a modest decrease in computational expense. However, since the local breakdown was calculated from existing flow properties which were subject to statistical scatter, accurate detection of non-equilibrium regions was very difficult.

To counter this, Smith [97] utilised a continuum solver in parallel with the direct simulation and used its results to attempt to accurately predict regions of continuum breakdown. The continuum solver used the same computational grid and time step as the direct solver. The resulting flow field from the continuum solver was used to calculate the breakdown parameter and was used by the EPSM/DSMC hybrid to determine which solver to use. The additional computational expense associated with the continuum solver was minimal when compared to the expense of the direct solver. While the method provided modest decreases of computational expense, in many instances the continuum solver (EFM) was unable to accurately place the locations of thermal non-equilibrium. To counter this, a solution to the Navier-Stokes equations, provided by a Riemann solver [44], was used as a guide to locate regions of thermal non-equilibrium. Results showed that the regions of continuum breakdown as determined by EFM accurately corresponded with those seen in the hybrid DSMC/EPSM only when adaptive mesh refinement was employed with the target cell size on the order of the local mean free path [97].

6.3 Finite volume based hybrid solvers

Traditional hybrid methods with a finite volume solver component employ a conventional Navier-Stokes solver in regions of thermal equilibrium [91, 110, 102]. In most cases, the interfaces between DSMC and NS regions are separated by buffer cells and many of the technical challenges associated with a combined DSMC/NS solver have been overcome. However, the

statistical scatter present in the direct simulation component of this hybrid technique can still cause difficulties in accurately predicting regions of continuum breakdown. There may also be compatibility issues between the Navier-Stokes solver and the DSMC solver. Theoretically, if the non-continuum regions were forced to equilibrium, the results obtained should be identical to the results obtained if the continuum solver was used in all regions of the flow. However, traditional Navier-Stokes solvers are based on fundamentally different concepts to a direct solver such as DSMC. The inaccurate matching of numerical viscosity, fluxes of mass, momentum and energy and problems associated with direction decoupling associated with continuum methods could therefore lead to problems with the idea of the hybrid DSMC/NS method.

The recent development of 'unified' flow solvers aim to bridge the gap between existing continuum and direct simulation methods [111, 49, 50]. In a Unified Flow Solver (UFS), both the equilibrium and non-equilibrium solvers are based on the same theoretical concepts. Whereas previously employed hybrid solvers used continuum methods based on mathematical splitting of fluxes (such as Van Leer or AUSM), the recent development of continuum methods based on kinetic theory present an attractive alternative to existing methods due to the similarities in underlying principles [50, 117, 76]. Recent efforts by Kolobov *et al.* [50] to create a unified flow solver combining a Boltzmann Solver and a second order form of EFM were successful in simulating hypersonic flow around blunt bodies. In this method, the breakdown of equilibrium was predicted using the local Knudsen number using gradients of density to calculate a local characteristic length, as shown in Equation 3.2.

The family of Unified Flow Solvers presented by Kolobov *et al.* [50] is capable of solving regions of thermal non-equilibrium without the scatter associated with the use of a direct solver. The use of a kinetic theory based solve in equilibrium regions means the method can solve engineering problems without the computational restrictions associated with the intensive BGK solvers. However, the fluxes employed in regions of thermal equilibrium are one dimensional fluxes. Unlike particle based methods, at least two time steps a required to move fluxes to their physically correct destinations. The use of direct coupled fluxes by both the equilibrium and non-equilibrium solvers would result in results of higher fidelity in flows not aligned with the computational grid.

Chapter 7

Blast Wave Simulations in Computational Fluid Dynamics

7.1 Introduction and Summary

The simulation of the effects of blast waves has been of key importance to the mining industry, health industry and in recent efforts to protect by design against terrorism [60]. The study of the generation and propagation of blast waves were always of great importance from the safety viewpoint for any chemical industry handling explosive materials or flammable gases [86]. A large number of past explosion accidents has led to considerable property damages, in addition to human injuries along with fatalities in some cases [34]. Thus, simulations of blast waves have been performed in large tunnel systems [86, 103] due to the dangerous nature of the chemicals used in them.

7.2 Blast waves and tunnel systems

An example of the work done by van den Berg [103] on tunnel blast waves is shown in Figure 7.1. The blast phenomena in the tunnel system shown in Figure 7.1 was originating from an instantaneous release $50m^3$ of liquid propane at $326K$. The blast phenomena during the early stages of the propagation have been visualised in a sequence of pictures showing the gas dynamics at a few consecutive points in time [103].

7.3 Blast waves and structures

DSMC has been used as a continuum solver to simulate blast waves [60, 4]. Research by Long [60] using DSMC to calculate the effect on structures by blast waves has been conducted with the aim of reducing the cost of live testing using real buildings. Long utilised DSMC and compared results against experiments for the square cavity problem and high speed flow over a box and I-beam. Long used collision limiting and tested conditions in cells to ensure particles within a cell had reached equilibrium in a given time step.

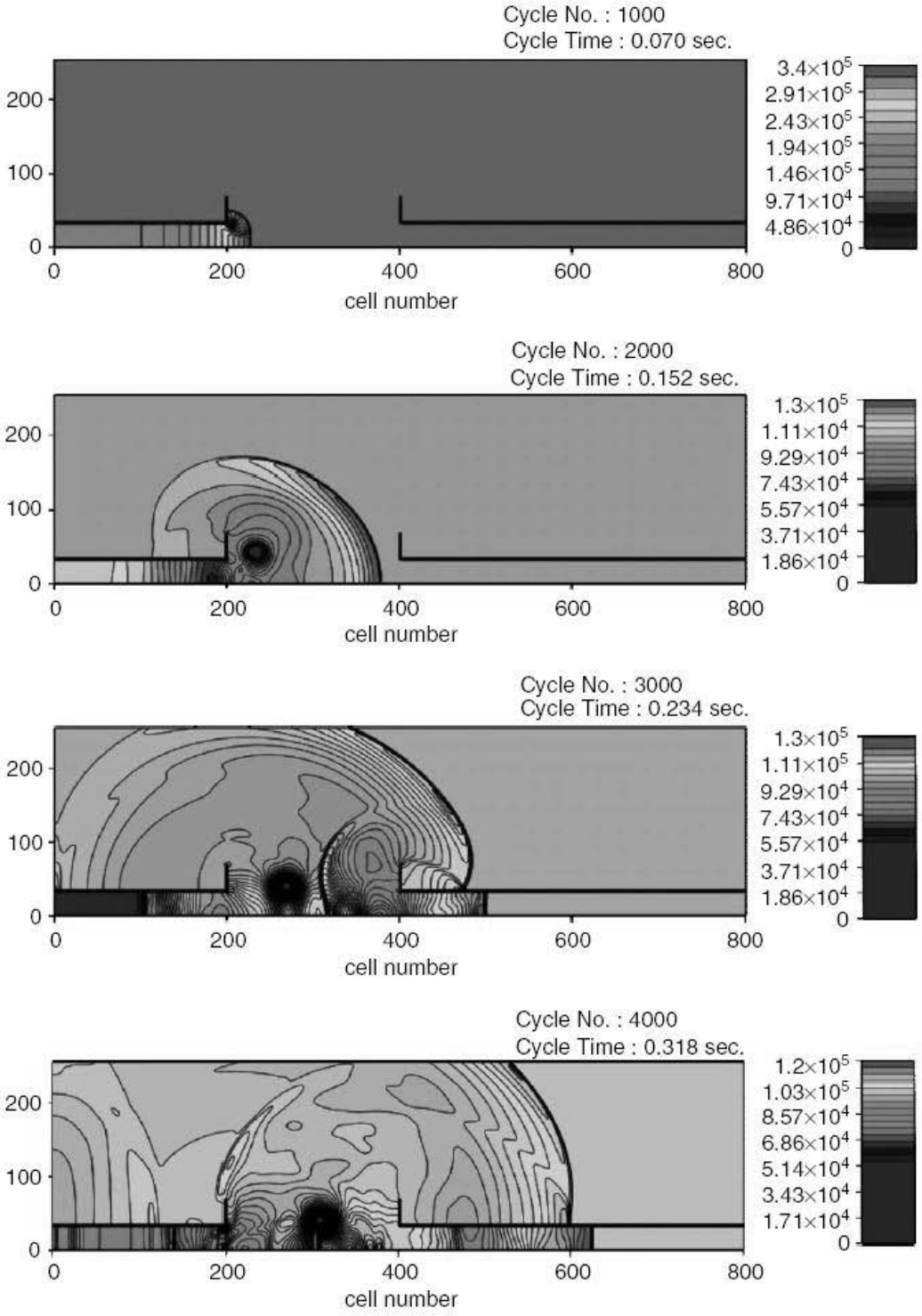

Figure 7.1: Blast propagation from a tunnel tube via an open space into another tunnel tube [103].

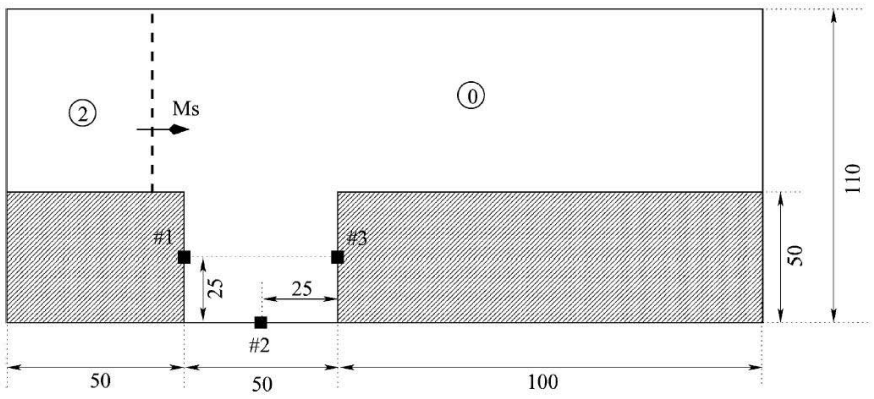

Figure 7.2: A schematic showing the dimensions of the cavity and the pressure transducer locations used by Reichenbach et. at. [83] and Long [60]. All dimensions are in mm.

The square cavity problem investigated by Reichenbach et. al [83] and used by Long to compare against DSMC simulations [60] is shown in Figure 7.2. They presented Shadowgraphs at different times during the experiments for two incident shock wave Mach numbers, 1.3 and 2.032 [60]. Long's simulation used 800x440 cells with 40 particles in each cell at the start of the simulation. These were compared qualitatively to Reichenbach's results and used incident shock wave Mach numbers of 1.43 and 2.032. To help reduce scatter in the resulting profiles, roughly 1200 samples were collected for the results. All solid boundary reflections off surfaces were treated as specular. The shadow graphs taken from the experiment, along with the DSMC results for $M = 2.032$ are shown in Figure 7.3. The results for $M = 1.43$ are shown in Figure 7.4.

The simulation of flow over a square box and I-beam was then considered. The setup geometry for each simulation is shown in Figure 7.5. Each DSMC simulation used 600x300 cells with 60 particles per cell at the start of the simulation. To reduce scatter 1800 flow samples were taken. The initial conditions used were $\rho_1 = 1.14kgm^{-3}$, $T_1 = 196.45K$ and $M_s = 1.98$. The top and bottom boundaries are specularly reflecting walls. The results from DSMC simulations of flow over a square box are presented in Figure 7.6, and the results for the I-beam simulation are presented in Figure 7.7.

The effect of blast waves in city environments has also been a key area of research [100, 90, 61, 85, 84]. Rose [100] recently investigated the propagation of blast waves in city streets by examining different configurations of intersections, street corners and the effect of the surrounding buildings on the path of the flow. Data was collected both experimentally and through the use of computational simulation. Rose examined the confining effects of surrounding buildings in a blast situation using the geometry shown in Figure 7.8. This problem, initially investigated by Smith and Feng [99], describes a situation where a blast is confined by nearby buildings. A charge is located roughly 1/3 of the way down the street. Rose found that, as a result of the confining effects of the buildings, the pressure measured at the location shown in Figure 7.8 was approximately 4 times higher than when buildings were omitted. In reality, this result will be lower due to the likely failures of the buildings in the immediate vicinity of the blast [100].

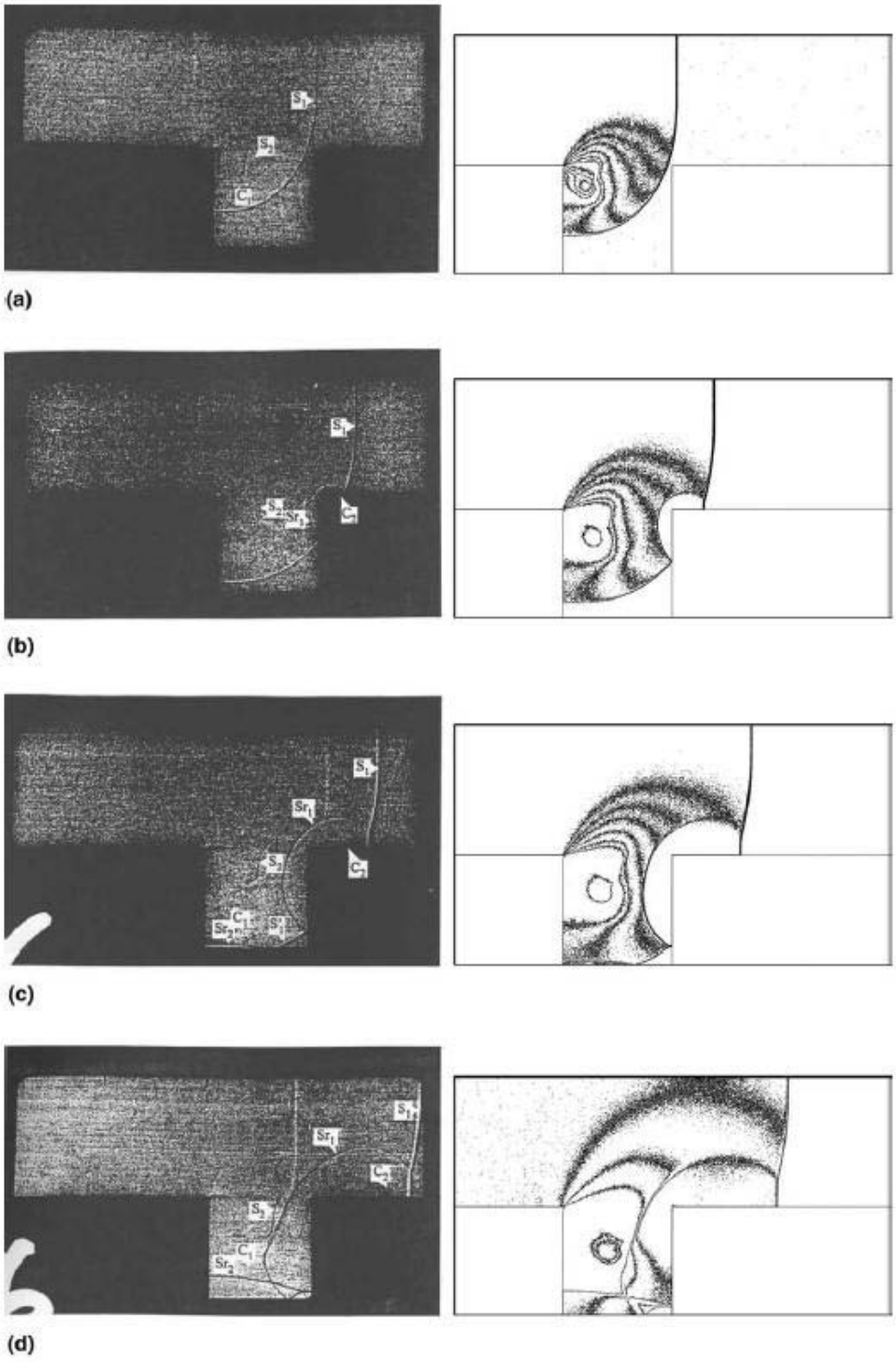

Figure 7.3: The interaction of a traveling planar shock wave (Mach number 2.032) with a square cavity at different times. On the left are Shadow graphs from [83], and on the right are plots of constant density from DSMC simulations [60]. (a)$t = 75\mu s$; (b) $t = 100\mu s$; (c) $t = 125\mu s$; (d) $t = 150\mu s$.

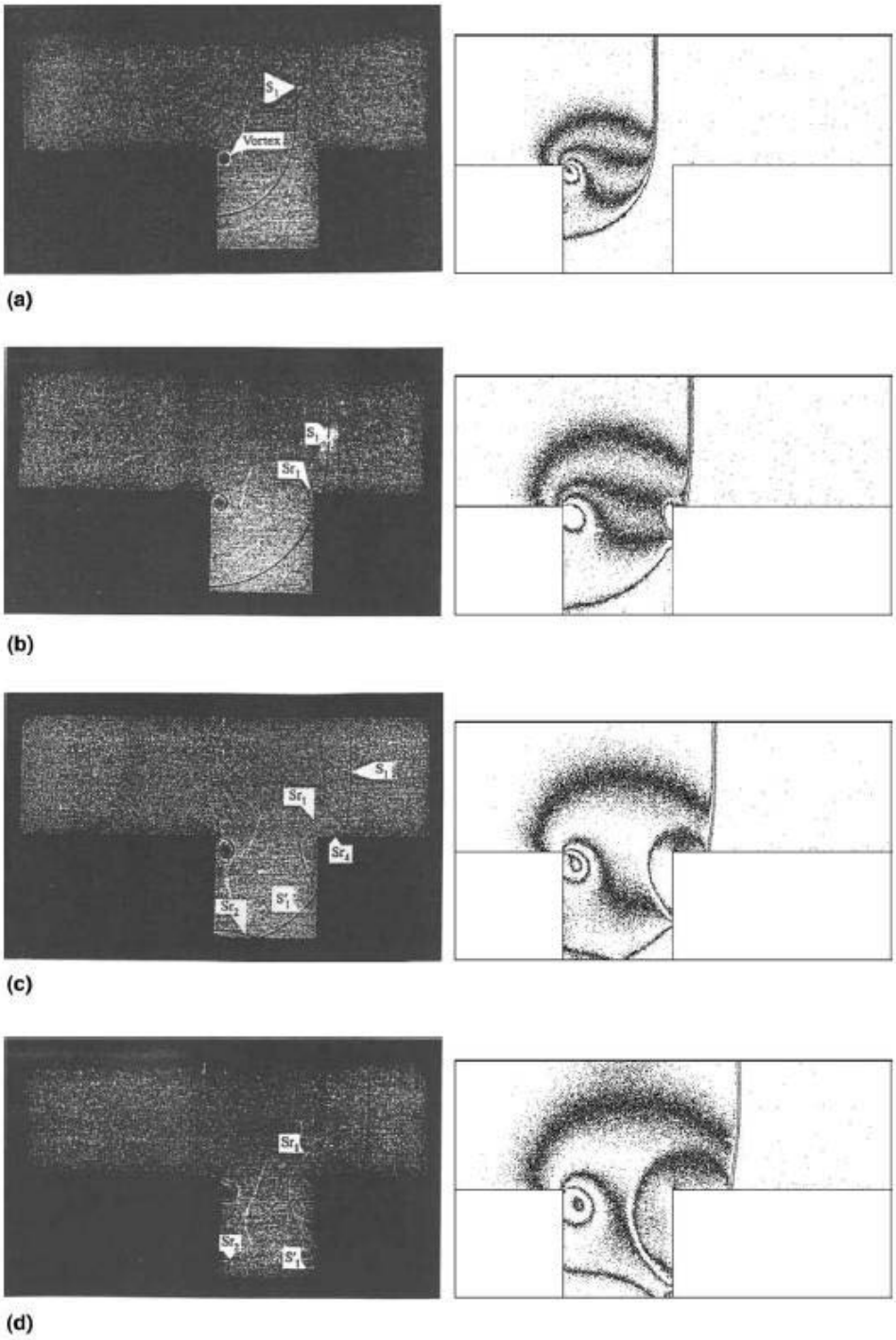

Figure 7.4: The interaction of a traveling planar shock wave (Mach number 1.43) with a square cavity at different times. On the left are Shadow graphs from [83], and on the right are plots of constant density from DSMC simulations [60]. (a)$t = 100\mu s$; (b) $t = 140\mu s$; (c) $t = 160\mu s$; (d) $t = 180\mu s$.

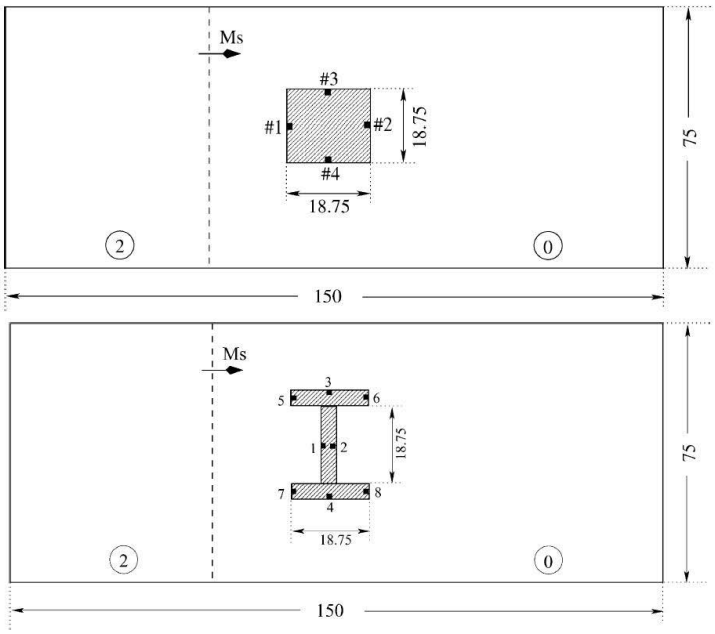

Figure 7.5: Geometry used by [60] for flow over a square box (top) and flow over an I-beam (bottom). The I-beam has a uniform thickness of 3.75 mm. All dimensions are in mm. Square numbered points represent locations where Long [60] collected information on pressure for comparison to physical experiments.

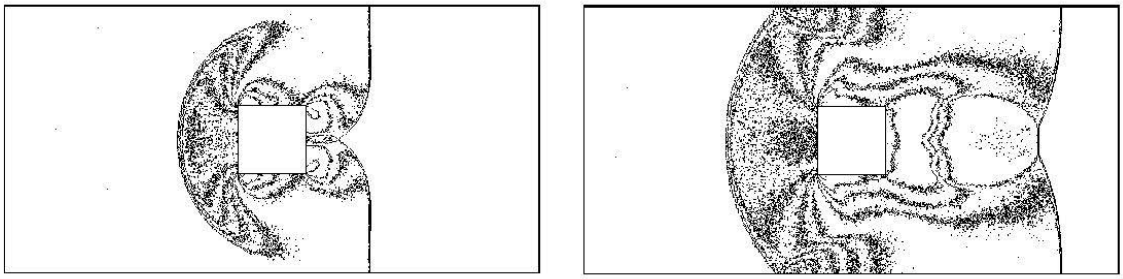

Figure 7.6: Plots of constant density from [60] at different times for the blast impact simulation of a box at $t = 68.1\mu s$ (left) and $t = 113.5\mu s$ (right)

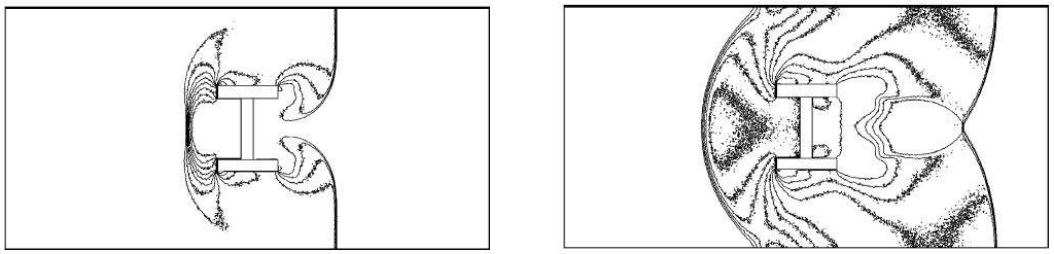

Figure 7.7: Plots of constant density from [60] at different times for the blast impact simulation of an I-beam at $t = 68.1\mu s$ (left) and $t = 113.5\mu s$ (right)

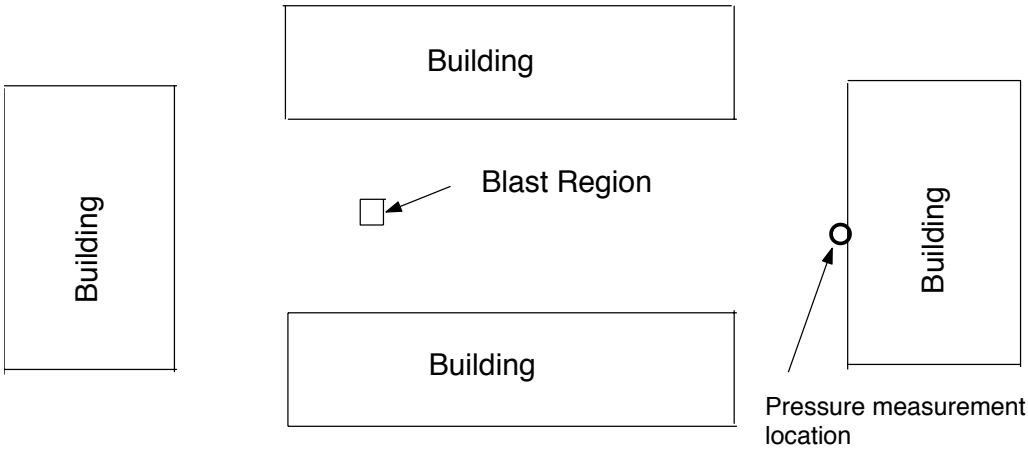

Figure 7.8: The building configuration used by Rose [100] and Feng [99] to test the confining effects of buildings on the propagation of blast waves. The initial charge is located approximately one third the way down the street and pressure measured on the face of the distant building as indicated above.

Chapter 8

The True Direction Equilibrium Flux Method

8.1 TDEFM - True Direction Equilibrium Flux Method

8.1.1 TDEFM with uniformly distributed mass

Derived below are the expressions for the mass, momentum and energy carried by molecules in free-molecular flight for time Δt, starting from a rectangular region (in 2D) to any other rectangular region. For simplicity all forces acting on particles are assumed to be zero, *i.e.* no particle interactions occur while particles are moving. Internal structural energy (such as energy due to rotation and vibration) is included in the energy flux expressions so monatomic, diatomic or polyatomic gases can be simulated.

Uniform conditions are assumed within the cell from which the molecules originate (*i.e.* there are no gradients of density, mean velocity or temperature within the cell) and all the molecules within the cell have velocities conforming to the same Maxwell-Boltzmann distribution. The distribution function for components of molecular velocity, $v_j \equiv v_x$ or v_y or v_z, has the Maxwell-Boltzmann form

$$f\left(v_x, v_y, v_z\right) = g\left(v_x\right) g\left(v_y\right) g\left(v_z\right)$$

where

$$g\left(v_j\right) = \frac{1}{\pi^{1/2} c_m} \exp\left(-\frac{\left(v_j - V_j\right)^2}{c_m^2}\right),\ V_j \equiv \bar{v}_j = \int_{-\infty}^{\infty} v_j g dv_j \text{ and } c_m = \left(2RT\right)^{-1/2}.$$

In other words, the fraction of molecules having a velocity v_x in the range $v_x \rightarrow v_x + dv_x$ is $g\left(v_x\right) dv_x$ and similar expressions hold for v_y and v_z. The components of the mean flow velocity (mean molecular velocity) in any cell are V_x and V_y (and $V_z = 0$ for 2D flow), the mass density is $\rho = m_p n$, where m_p is the mass of one molecule and n is the number density (molecules/m^3). The random thermal velocity is $c_j = v_j - V_j$ and the three components of translational kinetic temperature are given by $RT_j = \int_{-\infty}^{\infty} c_j^2\, dv_j$ where R is the ordinary gas constant.

Setting $s \equiv \sqrt{RT}$ and $m \equiv V_j$ the expression for $g(v)$ can be rewritten as:

$$g(v_j) = \frac{1}{\sqrt{2\pi} s} \exp\left(\frac{-(v_j - m)^2}{2s^2}\right) \tag{8.1}$$

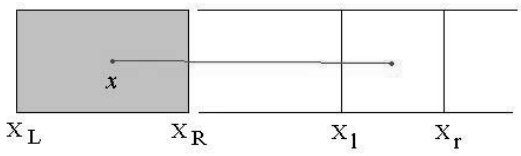

Figure 8.1: Particle moving from x (between x_L and x_R) to a region between x_l and x_r. For the derivations used here, $x_r \geq x_l$ & $x_R \geq x_L$

Referring to Figure 8.1, for a particle at location x to travel to a location between x_l and x_r in a time space t, the velocity range falls between $\frac{x_l - x}{t}$ and $\frac{x_r - x}{t}$. Therefore, the chance of a particle at position x moving to between $x_l - x_r$ is:

$$\begin{aligned} P_m &= \int_{\left(\frac{x_l-x}{\Delta t}\right)}^{\left(\frac{x_r-x}{\Delta t}\right)} \frac{1}{\sqrt{2\pi}s}\exp\left(\frac{-(v_x-m)^2}{2s^2}\right)dv_x \\ &= \frac{1}{2}\left[\mathrm{erf}\left(\frac{m\Delta t + x - x_l}{\sqrt{2}s\Delta t}\right) - \mathrm{erf}\left(\frac{m\Delta t + x - x_r}{\sqrt{2}s\Delta t}\right)\right] \end{aligned} \tag{8.2}$$

The average probability of a particle having the required velocity range over the space $x_L - x_R$ represents the fraction of particles from the region between x_L and x_R possessing the velocities specified and is given by:

$$\begin{aligned} f_M &= \frac{1}{(x_R - x_L)}\int_{x_L}^{x_R} P_m dx \\ &= \mathbf{f}_{\mathrm{M}}(\mathrm{m}, \mathrm{s}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \\ &= M_c\exp\left(\frac{-(m\Delta t + x_R - x_l)^2}{2s^2\Delta t^2}\right) + M_1\mathrm{erf}\left(\frac{m\Delta t + x_R - x_l}{\sqrt{2}s\Delta t}\right) \\ &- M_c\exp\left(\frac{-(m\Delta t + x_R - x_r)^2}{2s^2\Delta t^2}\right) - M_2\mathrm{erf}\left(\frac{m\Delta t + x_R - x_r}{\sqrt{2}s\Delta t}\right) \\ &- M_c\exp\left(\frac{-(m\Delta t + x_L - x_l)^2}{2s^2\Delta t^2}\right) - M_3\mathrm{erf}\left(\frac{m\Delta t + x_L - x_l}{\sqrt{2}s\Delta t}\right) \\ &+ M_c\exp\left(\frac{-(m\Delta t + x_L - x_r)^2}{2s^2\Delta t^2}\right) + M_4\mathrm{erf}\left(\frac{m\Delta t + x_L - x_r}{\sqrt{2}s\Delta t}\right) \end{aligned} \tag{8.3, 8.4}$$

This equation can be used to find the fraction of mass from region $x_L \leftrightarrow x_R$ that flows into the region between $x_l \leftrightarrow x_r$. The mean velocity of particles (or the mean momentum per unit mass) from location x to land in the region between x_l and x_r is:

$$
\begin{aligned}
P_p &= \int_{\frac{x_l-x}{\Delta t}}^{\frac{x_r-x}{\Delta t}} v_x \frac{1}{\sqrt{2\pi}s}\exp\left(\frac{-(v_x-m)^2}{2s^2}\right)dv_x \\
&= \left[-\frac{s}{\sqrt{2\pi}}\exp\left(\frac{-(m-v_x)^2}{2s^2}\right) - \frac{m}{2}\mathrm{erf}\left(\frac{m-v_x}{\sqrt{2}s}\right)\right]_{\frac{x_l-x}{\Delta t}}^{\frac{x_r-x}{\Delta t}}
\end{aligned}
\tag{8.5}
$$

The mean average velocity of particles moving into region $x_l \leftrightarrow x_r$ from region $x_L \leftrightarrow x_R$ is:

$$
\begin{aligned}
f_P &= \frac{1}{(x_R-x_L)}\int_{x_L}^{x_R} P_p dx \\
&= \mathbf{f}_{\mathrm{P}}(\mathrm{m},\mathrm{s},\Delta\mathrm{t},\mathrm{x_R},\mathrm{x_L},\mathrm{x_l},\mathrm{x_r}) \qquad (8.6) \\
&= P_c\exp\left(\frac{-(m\Delta t + x_R - x_l)^2}{2s^2\Delta t^2}\right) + P_1\mathrm{erf}\left(\frac{m\Delta t + x_R - x_l}{\sqrt{2}s\Delta t}\right) \\
&- P_c\exp\left(\frac{-(m\Delta t + x_R - x_r)^2}{2s^2\Delta t^2}\right) - P_2\mathrm{erf}\left(\frac{m\Delta t + x_R - x_r}{\sqrt{2}s\Delta t}\right) \\
&- P_c\exp\left(\frac{-(m\Delta t + x_L - x_l)^2}{2s^2\Delta t^2}\right) - P_3\mathrm{erf}\left(\frac{m\Delta t + x_L - x_l}{\sqrt{2}s\Delta t}\right) \\
&+ P_c\exp\left(\frac{-(m\Delta t + x_L - x_r)^2}{2s^2\Delta t^2}\right) + P_4\mathrm{erf}\left(\frac{m\Delta t + x_L - x_r}{\sqrt{2}s\Delta t}\right) \qquad (8.7)
\end{aligned}
$$

The mean energy of particles (per unit mass) moving from x into the region between x_l and x_r,P_e, is defined as:

$$
\begin{aligned}
P_e &= \int_{\frac{x_l-x}{\Delta t}}^{\frac{x_r-x}{\Delta t}} \frac{(0.5v_x^2+C)}{\sqrt{2\pi}s}\exp\left(\frac{-(v_x-m)^2}{2s^2}\right)dv_x \\
&= \left[\frac{1}{4}\left(m^2+s^2+2C\right)\mathrm{erf}\left(\frac{x-m}{\sqrt{2}s}\right) - \frac{s}{2\sqrt{2\pi}}\exp\left(\frac{-(m-x)^2}{2s^2}\right)\right]_{\frac{x_l-x}{\Delta t}}^{\frac{x_r-x}{\Delta t}}
\end{aligned}
\tag{8.8}
$$

where C is a molecules internal structural energy and is explained in Section 2. The mean energy over the range x_L to x_R to flow into the region between x_l and x_r we will call f_{EE}. This is evaluated as:

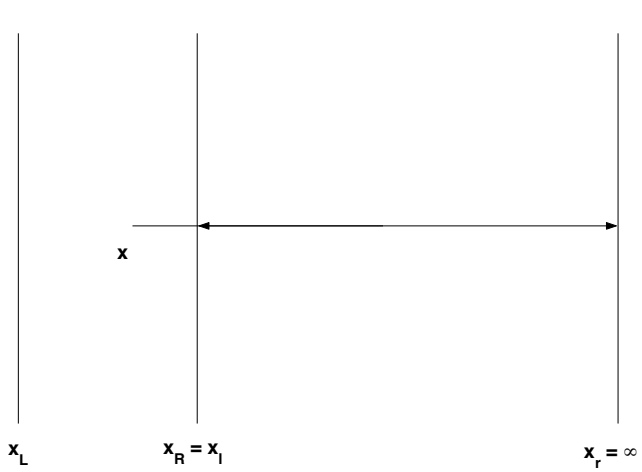

Figure 8.2: Particle moving from x (between x_L and x_R) to a region between x_l and ∞.

$$
\begin{aligned}
f_E &= \frac{1}{(x_R - x_L)} \int_{x_L}^{x_R} P_e dx \\
&= \mathbf{f}_{\mathrm{E}}(\mathrm{m}, \mathrm{s}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \\
&= E_c \exp\left(\frac{-(m\Delta t + x_R - x_l)^2}{2s^2\Delta t^2}\right) + E_1 \mathrm{erf}\left(\frac{m\Delta t + x_R - x_l}{\sqrt{2}s\Delta t}\right) \\
&- E_c \exp\left(\frac{-(m\Delta t + x_R - x_r)^2}{2s^2\Delta t^2}\right) - E_2 \mathrm{erf}\left(\frac{m\Delta t + x_R - x_r}{\sqrt{2}s\Delta t}\right) \\
&- E_c \exp\left(\frac{-(m\Delta t + x_L - x_l)^2}{2s^2\Delta t^2}\right) - E_3 \mathrm{erf}\left(\frac{m\Delta t + x_L - x_l}{\sqrt{2}s\Delta t}\right) \\
&+ E_c \exp\left(\frac{-(m\Delta t + x_L - x_r)^2}{2s^2\Delta t^2}\right) + E_4 \mathrm{erf}\left(\frac{m\Delta t + x_L - x_r}{\sqrt{2}s\Delta t}\right)
\end{aligned}
\tag{8.9}
$$

The fluxes that flow into region x_l and x_r where (x_l & $x_r < x_L$) are calculated using similar integrals to the ones above. To calculate the fraction of mass that remains in the region $x_L - x_R$, we use the result from Equation 8.4 and set $x_l = x_L$ and $x_r = x_R$. The same theory also applies to the momentum and energy fluxes using Equations 8.7 and 8.10.

8.1.2 Generalised EFM flux expressions

Instead of using finite velocity bounds to derive flux expressions we can follow the convention of existing kinetic theory based solvers and use an infinite bound for the far side of the destination region and set the near side to the edge of the source cell. This is shown in Figure 8.2. The distribution function for components of molecular velocity, $v_j \equiv v_x$ or v_y or v_z, has the Maxwell-Boltzmann form

$$
g(v_j) = \frac{1}{\sqrt{2\pi}s} \exp\left(\frac{-(v_j - m)^2}{2s^2}\right) \tag{8.10}
$$

where $s \equiv \sqrt{RT}$ and m is the mean velocity. The probability of a particle with a velocity

between a and b, or P_m, is:

$$P_m^+ = \int_a^b \frac{1}{\sqrt{2\pi}s} \exp\left(\frac{-(v_j - m)^2}{2s^2}\right) dv_j \tag{8.11}$$

Referring to Figure 8.2, for a particle at location x to travel to a location between x_l and x_r in a time space t, the velocity range falls between $\frac{x_l - x}{t}$ and ∞. Therefore, the chance of a particle at position x moving to between $x_l - \infty$ is:

$$\begin{aligned} P_m^+ &= \int_{\left(\frac{x_l - x}{t}\right)}^{\infty} \frac{1}{\sqrt{2\pi}s} \exp\left(\frac{-(v_x - m)^2}{2s^2}\right) dv_x \\ &= \frac{1}{2}\left[\text{erf}\left(\frac{mt + x - x_l}{\sqrt{2}st}\right) + 1\right] \end{aligned} \tag{8.12}$$

The mean probability over the source cell $x_L - x_R$ is found by integrating this result over x and is given by:

$$\begin{aligned} f_M^+ &= \frac{Z}{2\sqrt{\pi}}\left[\exp\left(-S_n^2\right) - \exp\left(-(S_n - \tfrac{1}{Z})^2\right)\right] \\ &+ \tfrac{1}{2} S_n Z\left[\text{erf}(S_n) - \text{erf}\left(S_n - \tfrac{1}{Z}\right)\right] + \tfrac{1}{2}\left[\text{erf}\left(S_n - \tfrac{1}{Z}\right) + 1\right] \end{aligned} \tag{8.13}$$

where the speed ratio $S_n = \frac{m}{c_m}$, $c_m \equiv \sqrt{2RT}$ is the most probable speed and $Z \equiv \frac{t\sqrt{2RT}}{\Delta x} = \frac{tc_m}{\Delta x}$. In this form, it is clear to see that f_M^+ is dimensionless, and represents the fraction of the mass to move from the source region to the destination region. Using the same technique to find momentum and energy fluxes, the new modified flux expressions (per unit time per unit area) are:

$$F_Q^+ = W_L \begin{bmatrix} \rho v_n \\ \rho v_n v_n + \rho RT \\ \rho v_n v_p \\ \rho v_n v_q \\ \rho v_n \left(\frac{1}{2}\vec{v}.\vec{v} + C_p T\right) \end{bmatrix} + D_L \begin{bmatrix} \rho c_m \\ \rho c_m v_n \\ \rho c_m v_p \\ \rho c_m v_q \\ \rho c_m E \end{bmatrix} + B_L \begin{bmatrix} \rho c_m \\ \rho c_m v_n \\ 0 \\ 0 \\ \rho c_m \left(\frac{1}{2}\vec{v}.\vec{v} + C_v T\right) \end{bmatrix} \tag{8.14}$$

i.e.

$$
\begin{aligned}
W_L &= \tfrac{1}{2}\left[\mathrm{erf}(S_n) - \mathrm{erf}\left(S_n - Z_L^{-1}\right)\right]_L \\
D_L &= \tfrac{1}{2\sqrt{\pi}}\left[\exp\left(-S_n^2\right) - \exp\left(-(S_n - Z_L^{-1})^2\right)\right]_L \\
B_L &= \tfrac{1}{2Z_L}\left[1 + \mathrm{erf}\left(S_n - Z_L^{-1}\right)\right] \\
S_n &= \left(v_n/c_m\right)_L \\
Z_L &= \left(\tfrac{c_m \Delta t}{\Delta x}\right)_L \\
E &= \tfrac{1}{2}\vec{v_L}.\vec{v_L} + \tfrac{1}{2}(\gamma+1)C_v T_L \\
c_m &= \sqrt{2RT}_L
\end{aligned}
\tag{8.15}
$$

C_p and C_v are the specific heats at constant pressure and volume respectively, and $\gamma = \frac{C_p}{C_v}$. Equivalently, the backward flux can be evaluated as:

$$
F_Q^- = W_R \begin{bmatrix} \rho v_n \\ \rho v_n v_n + \rho RT \\ \rho v_n v_p \\ \rho v_n v_q \\ \rho v_n \left(\frac{1}{2}\vec{v}.\vec{v} + C_p T\right) \end{bmatrix} + D_R \begin{bmatrix} \rho c_m \\ \rho c_m v_n \\ \rho c_m v_p \\ \rho c_m v_q \\ \rho c_m E \end{bmatrix} + B_R \begin{bmatrix} \rho c_m \\ \rho c_m v_n \\ 0 \\ 0 \\ \rho c_m \left(\frac{1}{2}\vec{v}.\vec{v} + C_v T\right) \end{bmatrix} \tag{8.16}
$$

I.e

$$
\begin{aligned}
W_R &= \tfrac{1}{2}\left[\mathrm{erf}(S_n) - \mathrm{erf}\left(S_n + Z_R^{-1}\right)\right]_R \\
D_R &= \tfrac{1}{2\sqrt{\pi}}\left[\exp\left(-S_n^2\right) - \exp\left(-(S_n + Z_R^{-1})^2\right)\right]_R \\
B_R &= \tfrac{1}{2Z_R}\left[1 - \mathrm{erf}\left(S_n + Z_R^{-1}\right)\right] \\
S_n &= \left(v_n/c_m\right)_R \\
Z_R &= \left(\tfrac{c_m \Delta t}{\Delta x}\right)_R \\
E &= \left(\tfrac{1}{2}\vec{v_R}.\vec{v_R} + \tfrac{1}{2}(\gamma+1)C_v T_R\right) \\
c_m &= \sqrt{2RT}_R
\end{aligned}
\tag{8.17}
$$

For small values of Z, Equations 8.14 and 8.16 reduce to the original EFM equations shown in Equations 4.14 and 4.16. However, in this revised form it is impossible for a cell to 'flux'

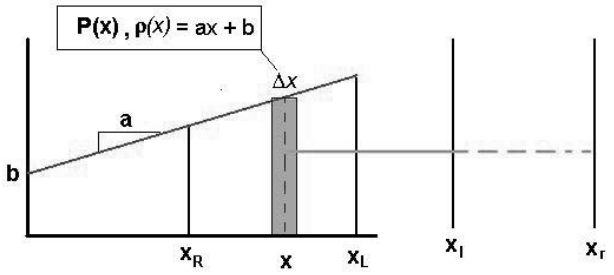

Figure 8.3: Particle moving from x (between x_L and x_R) to a region between x_l and x_r in the presence of a gradient in density. For the derivations used here, $x_r \geq x_l$ & $x_R \geq x_L$

more mass, momentum or energy than is originally present in the source cell, regardless of time step size. In this sense, the mechanism for transport of mass, momentum and energy in the revised EFM flux expressions are more closely related to that used by a direct simulation such as EPSM. However, the method can still become unstable as a result of the assumption that the neighbouring cell captures all of the transported mass, momentum and energy. In an EPSM simulation for a relatively large kinetic CFL, fluxes can pass over neighbouring cells and fall into cells further away. The proposed equations will only be valid when the kinetic CFL is less than 1 to ensure that the flux falling into the neighbouring regions are physically correct.

8.1.3 TDEFM with non-uniform mass distribution

A higher order equivalent of TDEFM is presented here where fluxes are recalculated with the presence of a density gradients in the source cell. The analytical fluxes obtained should then more closely match those of the infinite collision DSMC which allows for a density gradient across the cell[1]. The density in the source cell is assumed to take the form:

$$\rho(x) = ax + b \tag{8.18}$$

where a is the gradient of density across the cell and b is the value of density at x_L. For any given value of a, there is a fixed value of b which ensures the total mass in the cell is correct. Referring to Figure 8.4, for a particle at location x to travel to a location between x_l and x_r in a time space t, the velocity range falls between $\frac{x_l - x}{t}$ and $\frac{x_r - x}{t}$. Therefore, the chance of a particle being able to start from location x to between $x_l - x_r$ is:

$$\begin{aligned} P_m(x) &= \int_{\left(\frac{x_l-x}{\Delta t}\right)}^{\left(\frac{x_r-x}{\Delta t}\right)} \frac{1}{\sqrt{2\pi}s} \exp\left(\frac{-(v_x - m)^2}{2s^2}\right) dv_x \\ &= \frac{1}{2}\left[\mathrm{erf}\left(\frac{m\Delta t + x - x_l}{\sqrt{2}s\Delta t}\right) - \mathrm{erf}\left(\frac{m\Delta t + x - x_r}{\sqrt{2}s\Delta t}\right)\right] \end{aligned} \tag{8.19}$$

[1]During the collision process any temperature or bulk velocity gradient is destroyed, especially for high collision rates.

The mean fraction of the mass between x_L-x_R that will then move to between $x_l - x_r$ is:

$$f_M = \frac{1}{(x_R - x_L)} \int_{x_L}^{x_R} F(x) P_m(x) dx$$

where $F(x)$ is the fraction of mass in element dx, given by:

$$F(x) = \frac{(ax+b)\Delta x}{\int_{x_L}^{x_R}(ax+b)dx} \tag{8.20}$$

This integral is evaluated to obtain:

$$\begin{aligned}
f_M &= \mathbf{f}_{\mathrm{M}}(\mathrm{m}, \mathrm{s}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \\
&= M_1 \mathrm{exp}\left(\frac{-(m\Delta t + x_R - x_l)^2}{2s^2\Delta t^2}\right) + M_5 \mathrm{erf}\left(\frac{m\Delta t + x_R - x_l}{\sqrt{2}s\Delta t}\right) \\
&- M_2 \mathrm{exp}\left(\frac{-(m\Delta t + x_R - x_r)^2}{2s^2\Delta t^2}\right) - M_6 \mathrm{erf}\left(\frac{m\Delta t + x_R - x_r}{\sqrt{2}s\Delta t}\right) \\
&- M_3 \mathrm{exp}\left(\frac{-(m\Delta t + x_L - x_l)^2}{2s^2\Delta t^2}\right) - M_7 \mathrm{erf}\left(\frac{m\Delta t + x_L - x_l}{\sqrt{2}s\Delta t}\right) \\
&+ M_4 \mathrm{exp}\left(\frac{-(m\Delta t + x_L - x_r)^2}{2s^2\Delta t^2}\right) + M_8 \mathrm{erf}\left(\frac{m\Delta t + x_L - x_r}{\sqrt{2}s\Delta t}\right)
\end{aligned}$$

Similarly, the average momentum per unit mass f_P is given by:

$$\begin{aligned}
f_P &= \mathbf{f}_{\mathrm{P}}(\mathrm{m}, \mathrm{s}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \\
&= P_1 \mathrm{exp}\left(\frac{-(m\Delta t + x_R - x_l)^2}{2s^2\Delta t^2}\right) + P_5 \mathrm{erf}\left(\frac{m\Delta t + x_R - x_l}{\sqrt{2}s\Delta t}\right) \\
&- P_2 \mathrm{exp}\left(\frac{-(m\Delta t + x_R - x_r)^2}{2s^2\Delta t^2}\right) - P_6 \mathrm{erf}\left(\frac{m\Delta t + x_R - x_r}{\sqrt{2}s\Delta t}\right) \\
&- P_3 \mathrm{exp}\left(\frac{-(m\Delta t + x_L - x_l)^2}{2s^2\Delta t^2}\right) - P_7 \mathrm{erf}\left(\frac{m\Delta t + x_L - x_l}{\sqrt{2}s\Delta t}\right) \\
&+ P_4 \mathrm{exp}\left(\frac{-(m\Delta t + x_L - x_r)^2}{2s^2\Delta t^2}\right) + P_8 \mathrm{erf}\left(\frac{m\Delta t + x_L - x_r}{\sqrt{2}s\Delta t}\right)
\end{aligned} \tag{8.21}$$

The average energy flux per unit mass f_E is given by:

$$\begin{aligned}
f_E &= \mathbf{f}_{\mathrm{E}}(\mathrm{m}, \mathrm{s}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \\
&= E_1 \mathrm{exp}\left(\frac{-(m\Delta t + x_R - x_l)^2}{2s^2\Delta t^2}\right) + E_5 \mathrm{erf}\left(\frac{m\Delta t + x_R - x_l}{\sqrt{2}s\Delta t}\right) \\
&- E_2 \mathrm{exp}\left(\frac{-(m\Delta t + x_R - x_r)^2}{2s^2\Delta t^2}\right) - E_6 \mathrm{erf}\left(\frac{m\Delta t + x_R - x_r}{\sqrt{2}s\Delta t}\right) \\
&- E_3 \mathrm{exp}\left(\frac{-(m\Delta t + x_L - x_l)^2}{2s^2\Delta t^2}\right) - E_7 \mathrm{erf}\left(\frac{m\Delta t + x_L - x_l}{\sqrt{2}s\Delta t}\right) \\
&+ E_4 \mathrm{exp}\left(\frac{-(m\Delta t + x_L - x_r)^2}{2s^2\Delta t^2}\right) + E_8 \mathrm{erf}\left(\frac{m\Delta t + x_L - x_r}{\sqrt{2}s\Delta t}\right)
\end{aligned} \tag{8.22}$$

As expected these equations are identical to Equations[2] 8.4, 8.7 and 8.10 if $c = 0$ and $d = 1$.

[2]Located in A.1 of the Appendix

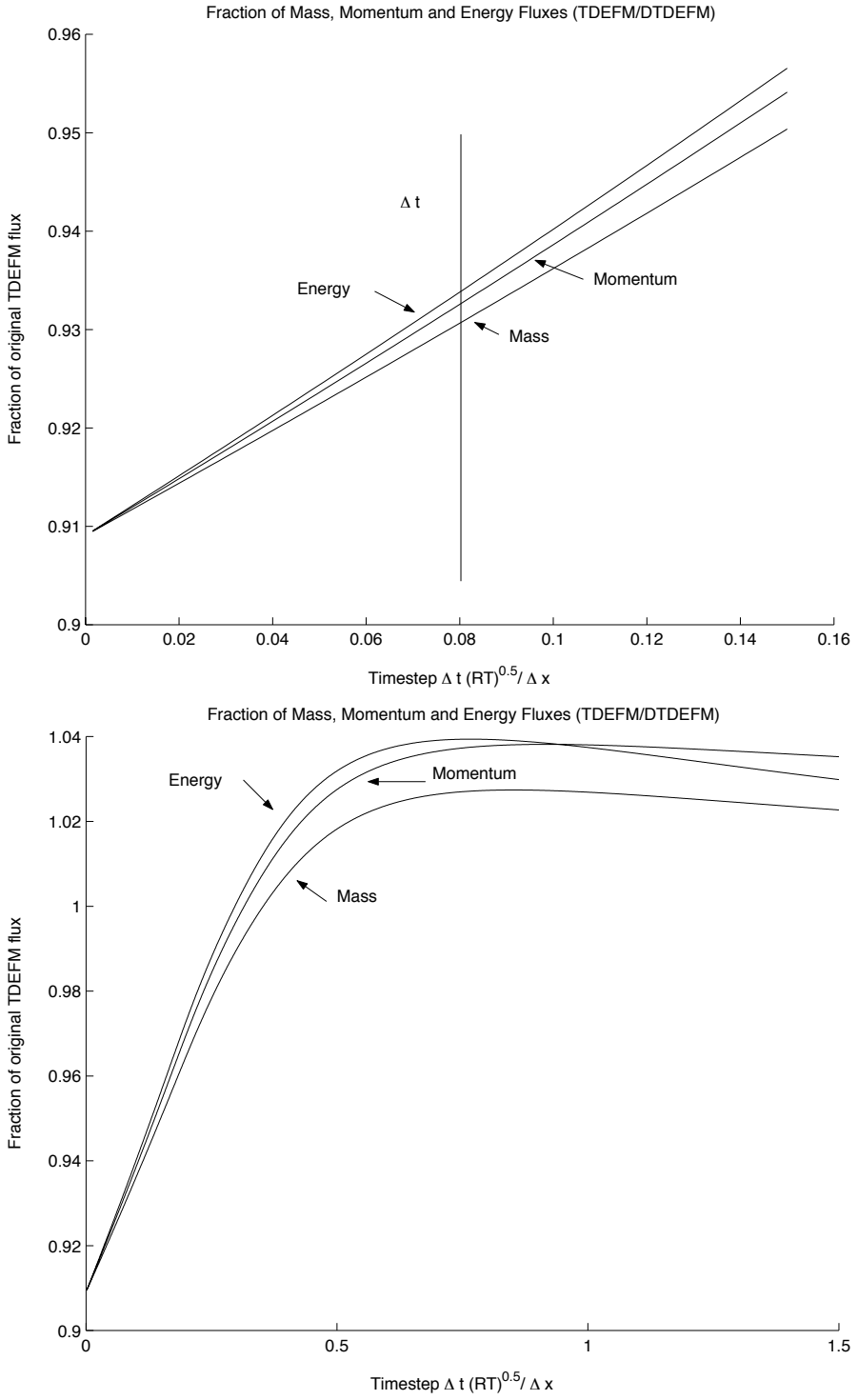

Figure 8.4: The fraction TDEFM/DTDEFM of fluxes for mass, energy and momentum for varying time steps. The density is assumed to increase by 20 percent across the cell. Mach number and temperature in the cell is assumed spatially constant.

Located in Figure 8.4 are the fractions of TDEFM fluxes for mass, momentum and energy for DTDEFM. The condition presented, an increase of 20 percent in the density across the cell width, is similar to the increase that occurs in the blast wave simulation through the shock wave as shown in Section 4. The fluxes are on the order of seven percent larger for a time step of $\Delta t(RT)^{0.5}/\Delta x = 0.08$ where Δx is the width of the source cell. Future potential exists for the inclusion of a density gradient using a higher order relationship, although the increase in accuracy would probably be outweighed by the increase in computational effort.

8.1.4 TDEFM with a non-uniform velocity distribution function

Equations 8.4, 8.7 and 8.10 were derived by assuming m, the bulk velocity in the source cell, was constant. If this velocity was a linear function of location x, then now we have:

$$m(x) = ax + b \tag{8.23}$$

Thus the Maxwell-Distribution becomes:

$$g(v_j) = \frac{1}{\sqrt{2\pi}s} \exp\left(\frac{-(v_x - m(x))^2}{2s^2}\right) \tag{8.24}$$

where $s = \sqrt{RT}$, v_x = speed in the horizontal direction of a particle originating from position x, and $m(x)$ = bulk velocity at location x. All the molecules within the cell have velocities conforming to the same Maxwell-Boltzmann distribution. Referring to Figure 8.1, the probability of a particle at location x having a velocity between $\frac{x_l - x}{t}$ and $\frac{x_r - x}{t}$ is:

$$\begin{aligned} P_m &= \int_{\left(\frac{x_l - x}{\Delta t}\right)}^{\left(\frac{x_r - x}{\Delta t}\right)} g(v_j)dv \\ &= \frac{1}{2}\left[\mathrm{erf}\left(\frac{(a\Delta t + 1)x + b\Delta t - x_l}{\sqrt{2}s\Delta t}\right) - \mathrm{erf}\left(\frac{(a\Delta t + 1)x + b\Delta t - x_r}{\sqrt{2}s\Delta t}\right)\right] \end{aligned}$$

The mean probability, or the fraction of mass, from the region $x_R - x_L$ to fall in the region $x_r - x_l$ is:

$$
\begin{aligned}
f_M &= \frac{1}{(x_R - x_L)} \int_{x_L}^{x_R} P_m dx \\
&= \mathbf{f}_{\mathrm{M}}(\mathrm{m}, \mathrm{s}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \qquad (8.25) \\
&= M_c \mathrm{exp}\left(\frac{-\left((a\Delta t+1)x_R + b\Delta t - x_l\right)^2}{2s^2\Delta t^2}\right) + M_1 \mathrm{erf}\left(\frac{(a\Delta t+1)x_R + b\Delta t - x_l}{\sqrt{2}s\Delta t}\right) \\
&- M_c \mathrm{exp}\left(\frac{-\left((a\Delta t+1)x_R + b\Delta t - x_r\right)^2}{2s^2\Delta t^2}\right) - M_2 \mathrm{erf}\left(\frac{(a\Delta t+1)x_R + b\Delta t - x_r}{\sqrt{2}s\Delta t}\right) \\
&- M_c \mathrm{exp}\left(\frac{-\left((a\Delta t+1)x_L + b\Delta t - x_l\right)^2}{2s^2\Delta t^2}\right) - M_3 \mathrm{erf}\left(\frac{(a\Delta t+1)x_L + b\Delta t - x_l}{\sqrt{2}s\Delta t}\right) \\
&+ M_c \mathrm{exp}\left(\frac{-\left((a\Delta t+1)x_L + b\Delta t - x_r\right)^2}{2s^2\Delta t^2}\right) + M_4 \mathrm{erf}\left(\frac{(a\Delta t+1)x_L + b\Delta t - x_r}{\sqrt{2}s\Delta t}\right) \qquad (8.26)
\end{aligned}
$$

The mean velocity of particles (or the momentum per unit mass) to travel from location x and land in the region between x_l and x_r is defined as:

$$
\begin{aligned}
P_p &= \int_{\frac{x_l - x}{\Delta t}}^{\frac{x_r - x}{\Delta t}} v_x \frac{1}{\sqrt{2\pi}s} \exp\left(\frac{-(v_x - m(x))^2}{2s^2}\right) dv_x \\
&= \left[\frac{-s}{\sqrt{2\pi}} \mathrm{exp}\left(\frac{-(ax+b-v)^2}{2s^2}\right) - \frac{1}{2}(ax+b)\mathrm{erf}\left(\frac{ax+b-v}{\sqrt{2}s}\right)\right]_{\frac{x_l - x}{\Delta t}}^{\frac{x_r - x}{\Delta t}}
\end{aligned}
$$

The average mean velocity taken over the region between $x_R - x_L$ is the momentum (per unit mass) to travel into the region between $x_l - x_r$ and is given by:

$$
\begin{aligned}
f_P &= \frac{1}{(x_R - x_L)} \int_{x_L}^{x_R} P_p dx \\
&= \mathbf{f}_{\mathrm{P}}(\mathrm{m}, \mathrm{s}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \qquad (8.27) \\
&= P_1 \mathrm{exp}\left(\frac{-\left((a\Delta t+1)x_R + b\Delta t - x_l\right)^2}{2s^2\Delta t^2}\right) + P_5 \mathrm{erf}\left(\frac{(a\Delta t+1)x_R + b\Delta t - x_l}{\sqrt{2}s\Delta\Delta t}\right) \\
&- P_2 \mathrm{exp}\left(\frac{-\left((a\Delta t+1)x_R + b\Delta t - x_r\right)^2}{2s^2\Delta t^2}\right) - P_6 \mathrm{erf}\left(\frac{(a\Delta t+1)x_R + b\Delta t - x_r}{\sqrt{2}s\Delta t}\right) \\
&- P_3 \mathrm{exp}\left(\frac{-\left((a\Delta t+1)x_L + b\Delta t - x_l\right)^2}{2s^2\Delta t^2}\right) - P_7 \mathrm{erf}\left(\frac{(a\Delta t+1)x_L + b\Delta t - x_l}{\sqrt{2}s\Delta t}\right) \\
&+ P_4 \mathrm{exp}\left(\frac{-\left((a\Delta t+1)x_L + b\Delta t - x_r\right)^2}{2s^2\Delta t^2}\right) + P_8 \mathrm{erf}\left(\frac{(a\Delta t+1)x_L + b\Delta t - x_r}{\sqrt{2}s\Delta t}\right) \qquad (8.28)
\end{aligned}
$$

The mean energy of particles (per unit mass) moving from x into the region between x_l and x_r,P_e, is defined as:

$$\begin{aligned} P_e &= \int_{\frac{x_l-x}{\Delta t}}^{\frac{x_r-x}{\Delta t}} \frac{(0.5v_x^2+C)}{\sqrt{2\pi}s}\exp\left(\frac{-(v_x-m(x))^2}{2s^2}\right)dv_x \\ &= \left[\frac{-s(ax+b+v)}{2\sqrt{2\pi}}\exp\left(\frac{-(ax+b-v)^2}{2s^2}\right)\right. \\ &\quad \left. + \frac{1}{4}(2C+s^2+(ax+b)^2)\mathrm{erf}\left(\frac{ax+b}{\sqrt{2}s}\right)\right]_{\frac{x_l-x}{\Delta t}}^{\frac{x_r-x}{\Delta t}} \end{aligned} \tag{8.29}$$

The average mean energy (per unit mass) to flow into the region between x_l and x_r is:

$$\begin{aligned} f_E &= \frac{1}{(x_R-x_L)}\int_{x_L}^{x_R} P_e dx \\ &= \mathbf{f}_{\mathrm{E}}(\mathrm{m},\mathrm{s},\Delta\mathrm{t},\mathrm{x_R},\mathrm{x_L},\mathrm{x_l},\mathrm{x_r}) \end{aligned} \tag{8.30}$$

$$\begin{aligned} &= E_1\exp\left(\frac{-\left((a\Delta t+1)x_R+b\Delta t-x_l\right)^2}{2s^2\Delta t^2}\right)+E_5\mathrm{erf}\left(\frac{(a\Delta t+1)x_R+b\Delta t-x_l}{\sqrt{2}s\Delta t}\right) \\ &- E_2\exp\left(\frac{-\left((a\Delta t+1)x_R+b\Delta t-x_r\right)^2}{2s^2\Delta t^2}\right)-E_6\mathrm{erf}\left(\frac{(a\Delta t+1)x_R+b\Delta t-x_r}{\sqrt{2}s\Delta t}\right) \\ &- E_3\exp\left(\frac{-\left((a\Delta t+1)x_L+b\Delta t-x_l\right)^2}{2s^2\Delta t^2}\right)-E_7\mathrm{erf}\left(\frac{(a\Delta t+1)x_L+b\Delta t-x_l}{\sqrt{2}s\Delta t}\right) \\ &+ E_4\exp\left(\frac{-\left((a\Delta t+1)x_L+b\Delta t-x_r\right)^2}{2s^2\Delta t^2}\right)+E_8\mathrm{erf}\left(\frac{(a\Delta t+1)x_L+b\Delta t-x_r}{\sqrt{2}s\Delta t}\right) \end{aligned} \tag{8.31}$$

A simple substitution of $a = 0$ and $b = m$ into these equations gives the same result as Equations 8.4, 8.7 and 8.10 respectively.

Located in Figure 8.5 are the fractions of TDEFM fluxes for mass, momentum and energy for VTDEFM. The condition presented is a decrease of 30 percent in the Mach number across the cell width. The fluxes are on the order of 15-20 percent smaller for a time step of $\Delta t(RT)^{0.5}/\Delta x = 0.08$ where Δx is the width of the source cell. As per DTDEFM, future potential exists for the inclusion of a velocity gradient using a higher order relationship.

8.2 Boundary conditions

8.2.1 Specular boundary reflections

The concept behind the specular boundary reflection is identical to that used by DSMC or other direct simulation methods. In a specular reflection from a wall where all quantities are

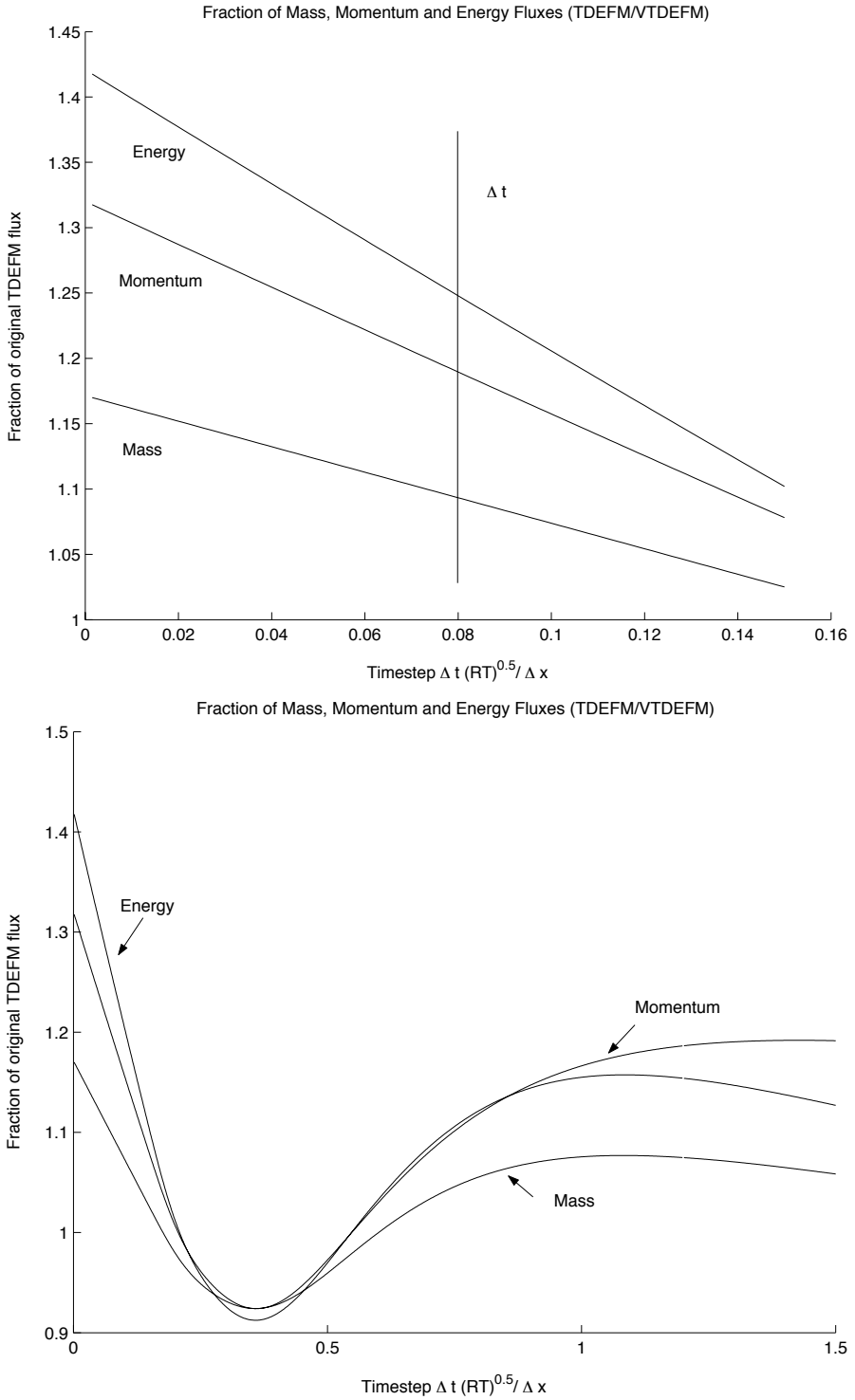

Figure 8.5: The fraction TDEFM/VTDEFM of fluxes for mass, energy and momentum for varying time steps. The Mach number is assumed to decrease by 30 percent across the cell. The density and temperature in the cell are spatially constant.

conserved, the only conserved quantity which needs to be updated is the momentum component normal to the surface of the wall. This is performed by calculating the flux of mass, momentum and energy to travel past the wall surface, changing the sign of the component of momentum perpendicular to the wall and then adding these fluxes back to the original source cell. Details of the implementation of this boundary condition for uniform and non-uniform computational grids are supplied in Section 8.3.

8.2.2 Diffuse boundary reflections

Previous implementations of TDEFM have focused on specular reflections of particles on boundaries [93, 98, 71]. Energy and momentum are conserved and reflected back into the source (or correct destination) cell. This boundary condition is useful for solutions to the Euler equations but not appropriate for real engineering flows [75]. In practise, most particle reflections of engineering surfaces are considered diffuse. Previous work by Mallett *et. al.* [72] employed diffusely reflective boundaries by integrating the probability distribution function for diffusely reflected particles; a similar approach is employed here. All mass leaving the simulation region through a diffusely reflective surface is automatically reinjected back into the flow with new momentum and energy. Details of this boundary implementation can be found in Section 8.3.

The velocity probability distribution function for the component of reflected velocity normal to the wall surface is:

$$f_n(v_n) = v_n s^{-2} \exp\left(\frac{-{v_n}^2}{2s^2}\right) \tag{8.32}$$

such that the probability of a reflected particle having a normal velocity between v_n and $v_n + dv_n$ is $f_n(v_n)dv_n$. The probability distribution functions for parallel components of velocity is assumed to be the Maxwell-Boltzmann equilibrium distribution function. Figure 8.6 shows the computational domain surrounding the diffusely reflecting surface between $x_L - x_R$. The fluxes of mass, momentum and energy (per unit mass) of diffusely reflected particles from region $x_L - x_R$ to fall in region x_l, y_l-x_r, y_r will be:

$$\begin{aligned}
f_M &= \frac{1}{(x_R - x_L)} \int_{x_L}^{x_R} \int_{\frac{(x_l - x)}{\Delta t}}^{\frac{(x_r - x)}{\Delta t}} \int_{\frac{(y_l - y_{Wall})}{\Delta t}}^{\frac{(y_r - y_{Wall})}{\Delta t}} f_n(v_n) f_{eq}(v_p) dv_n dv_p dx \\
f_{P_n} &= \frac{1}{(x_R - x_L)} \int_{x_L}^{x_R} \int_{\frac{(x_l - x)}{\Delta t}}^{\frac{(x_r - x)}{\Delta t}} \int_{\frac{(y_l - y_{Wall})}{\Delta t}}^{\frac{(y_r - y_{Wall})}{\Delta t}} v_n f_n(v_n) f_{eq}(v_p) dv_n dv_p dx \\
f_{E_n} &= \frac{1}{(x_R - x_L)} \int_{x_L}^{x_R} \int_{\frac{(x_l - x)}{\Delta t}}^{\frac{(x_r - x)}{\Delta t}} \int_{\frac{(y_l - y_{Wall})}{\Delta t}}^{\frac{(y_r - y_{Wall})}{\Delta t}} E_n f_n(v_n) f_{eq}(v_p) dv_n dv_p dx
\end{aligned} \tag{8.33}$$

where Δx is the width of the source region $x_R - x_L$ and E_n is the energy of a particle in the direction normal to the wall $E_n = 0.5v_n^2 + C$ where C is the internal energy per simulated

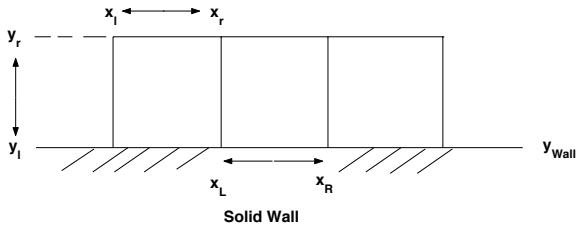

Figure 8.6: Computational domain for diffuse reflection from a surface.

degrees of freedom $C = (1/2SD)((2C_v/R) - SD)s^2$ where $SD = 2$ in a 2D simulation. The Maxwell-Boltzmann equilibrium distribution function f_{eq} is used in the directions parallel to the wall surface. In Equation 8.33 no assumptions are made regarding destination cell location. The destination region does not have to be adjacent to the wall surface, and any fraction of reflected particles can fall into the destination region. If the CFL number in the region near the wall is small, it is reasonable to assume that all reflected particles will be captured in the region between $y_l - y_r$. In this instance, the equations simplify to:

$$\begin{aligned}
f_M &= \frac{1}{(x_R - x_L)} \int_{x_L}^{x_R} \int_{\frac{(x_l - x)}{\Delta t}}^{\frac{(x_r - x)}{\Delta t}} \int_0^{\infty} f_n(v_n) f_{eq}(v_p) dv_n dv_p dx \\
&= \mathbf{f}_M(V_{wall}, s, \Delta t, x_R, x_L, x_l, x_r) \\
f_{P_n} &= \frac{1}{(x_R - x_L)} \int_{x_L}^{x_R} \int_{\frac{(x_l - x)}{\Delta t}}^{\frac{(x_r - x)}{\Delta t}} \int_0^{\infty} v_n f_n(v_n) f_{eq}(v_p) dv_n dv_p dx \\
&= \left(\sqrt{\frac{\pi}{2}} s\right) \times \mathbf{f}_P(V_{wall}, s, \Delta t, x_R, x_L, x_l, x_r) \\
f_{E_n} &= \frac{1}{(x_R - x_L)} \int_{x_L}^{x_R} \int_{\frac{(x_l - x)}{\Delta t}}^{\frac{(x_r - x)}{\Delta t}} \int_0^{\infty} E_n f_n(v_n) f_{eq}(v_p) dv_n dv_p dx \\
&= \left(s^2 + C\right) \times \mathbf{f}_E(V_{wall}, s, \Delta t, x_R, x_L, x_l, x_r)
\end{aligned} \tag{8.34}$$

where $s \equiv \sqrt{RT_{Wall}}$, $\sqrt{\pi/2}s$ is the mean normal velocity of all reflected particles, $s^2 + C$ is the mean normal energy of the reflected particles and V_{Wall} is the velocity of the wall. The proposed model provides an accommodation coefficient of 1.0, which is valid for many engineering applications with a few exceptions [75]. The momentum and energy components parallel to the surface of the wall are found using Equations 8.4, 8.7 and 8.10 exclusively since all particles are assumed to fall into the cell adjacent to the wall. If the wall has a non-zero velocity in the normal direction to its surface the above equations can be modified to accommodate, however, this study is limited to a wall moving parallel to its surface. Details of the implementation of the diffusely reflective boundary condition are provided in Section 8.3.2.

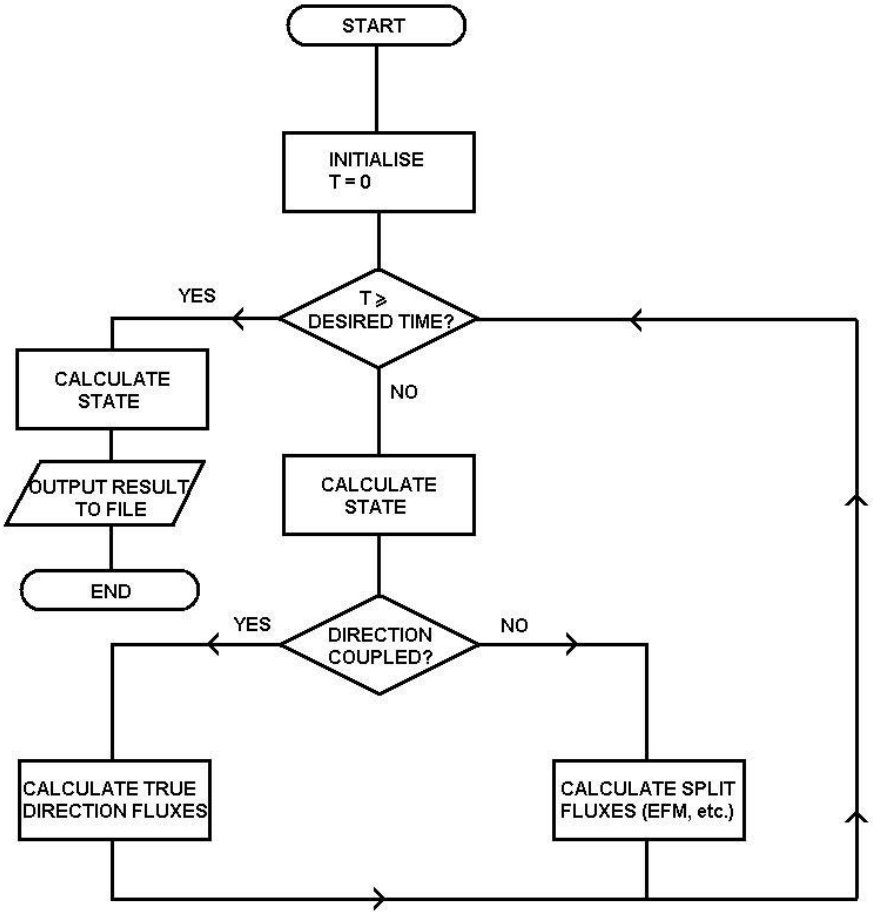

Figure 8.7: Program flowchart for uniform, cartesian computational grids.

8.3 Implementation of TDEFM

8.3.1 Implementation on a cartesian grid

The general implementation of TDEFM is described in the flowchart shown in Figure 8.7. For a regular cartesian grid, the computational space is divided into N_x and N_y cells (in the x and y directions) to provide a total of $N_x \times N_y$ cells. The information contained within these cells are stored in a series of two dimensional arrays. This approach greatly simplifies the implementation since no neighbour searching algorithms are required.

Initialisation and State Calculation

During the initialisation stage the cartesian grid is constructed and the initial conditions of the problem are set. In the case where there are buildings or obstructions in the flow, computational cells are flagged - 0 if the cell contains gas or 1 if the cell contains solid material, where an additional value representing material strength is also assigned.

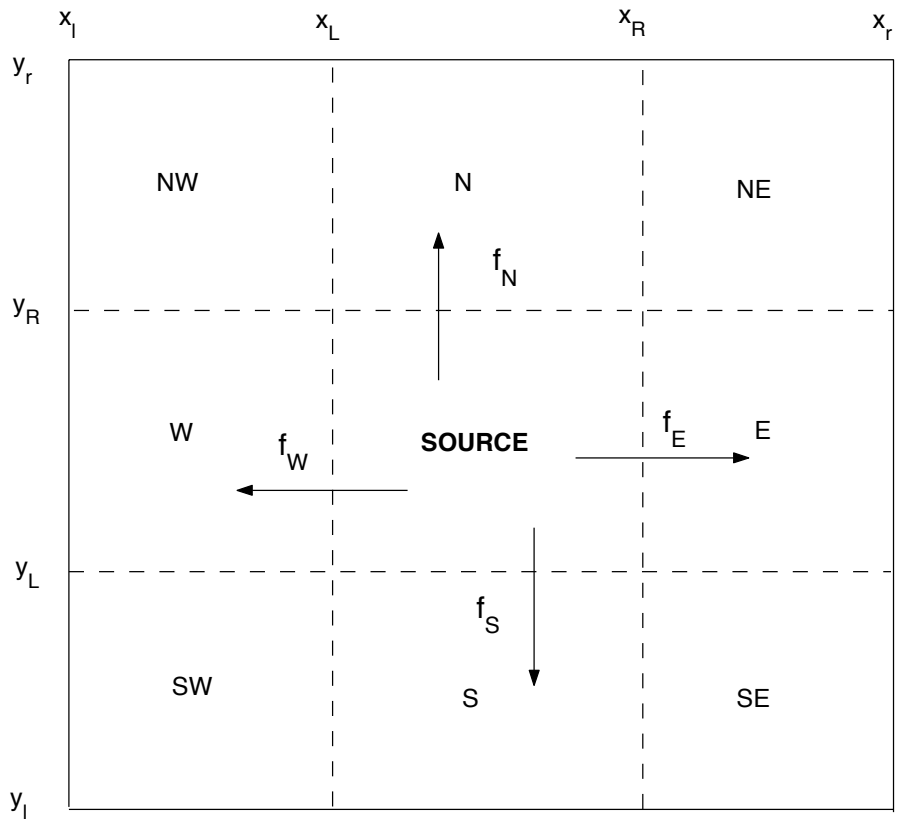

Figure 8.8: Diagram showing source cell (in center) surrounded by destination cells.

In the state calculation, the density, bulk velocities and temperature is calculated from the mass, momentum and energy arrays for each cell. The kinetic CFL is calculated and the time step is adjusted to make sure gas cannot propogate further than a neighbouring cell.

TDEFM flux calculation

The complete TDEFM flux expressions are lengthy and computationally expensive. Significant simplifications of these flux expressions can be performed when the computational domain is a simple cartesian mesh, as displayed in Figure 8.8. To calculate the mass fluxes from the source cell (in the region $x_L \geq x \geq x_R$, $y_L \geq y \geq y_R$) to all surrounding cells, only 4 total evaluations of f_M are required. The flux calculation procedure for the mass fluxes is:

1. Calculate values of f_N, f_S, f_E and f_W. In this instance, these values are:

$$\begin{aligned} f_N &= \mathbf{f}_{\mathrm{M}}(\mathrm{V}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{y_R}, \mathrm{y_L}, \mathrm{y_R}, \mathrm{y_r}) \\ f_S &= \mathbf{f}_{\mathrm{M}}(\mathrm{V}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{y_R}, \mathrm{y_L}, \mathrm{y_l}, \mathrm{y_L}) \\ f_E &= \mathbf{f}_{\mathrm{M}}(\mathrm{U}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_R}, \mathrm{x_r}) \\ f_W &= \mathbf{f}_{\mathrm{M}}(\mathrm{U}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_L}) \end{aligned}$$

   If we assume that (i) the local CFL is small, and (ii) that all of the mass is captured in the surrounding cells, the expressions for these fluxes simplify to Pullin's EFM fluxes,

requiring only a single erf() and exp() function evaluation each.

2. The fluxes of mass to the surrounding neighbours are:

$$\begin{aligned} M_{NW} &= M_0 \times f_N \times f_W \\ M_N &= M_0 \times f_N \times (1 - f_W - f_E) \\ M_{NE} &= M_0 \times f_N \times f_E \\ M_W &= M_0 \times (1 - f_N - f_S) \times f_W \\ M_E &= M_0 \times (1 - f_N - f_S) \times f_E \\ M_{SW} &= M_0 \times f_S \times f_W \\ M_S &= M_0 \times f_S \times (1 - f_W - f_E) \\ M_{SE} &= M_0 \times f_S \times f_E \end{aligned}$$

This procedure can be repeated for the momentum and energy fluxes. If the expressions are further simplified by assuming a small kinetic CFL number and complete capture by the surrounding cells, this procedure reduces the computational expense significantly, requiring 10 percent more computational time that ordinary EFM. If required, the "cell catchment" region could be increased to include more distant cells; however this would mean that the flow might posses an artificially large mean free path. In this study, the capture of fluxes is restricted to the surrounding 8 neighbours of each source cell.

During this phase, boundary conditions are also managed. The boundary conditions employed in this study on a cartesian grid are supersonic inflow, extrapolated outflow and specularly reflecting surfaces. For each of the inflow and extrapolated outflow boundary conditions, a 'reservoir' is employed. These reservoirs are imaginary cells identical in size to the adjacent real cells. In the case of an inflow boundary, the conditions in these cells is that of the freestream. For an extrapolated outflow boundary, the conditions in these reservoirs are determined from the nearest adjacent real cell. These reservoirs are identical in principle to the technique often used in DSMC to calculate inflow or outflow [58]. Since these regions are two (or three) dimensional regions, the equations used to calculate fluxes of mass, momentum and energy are the standard TDEFM flux expressions which integrate over both velocity space and physical space.

8.3.2 Boundary condition implementation

The general implementation of a specularly reflective boundary condition is presented. Specularly reflecting surfaces do not require ghost cells: instead, fluxes of mass, momentum and energy leaving the computational region are individually treated. This is demonstrated by examining the process of flux calculation in a corner of specularly reflective surfaces as shown in Figure 8.9. Fluxes of mass, momentum and energy are predicted to travel from cell i, j into a region inside a specularly reflecting body (cell $i + 1, j - 1$). The fluxes are subtracted from

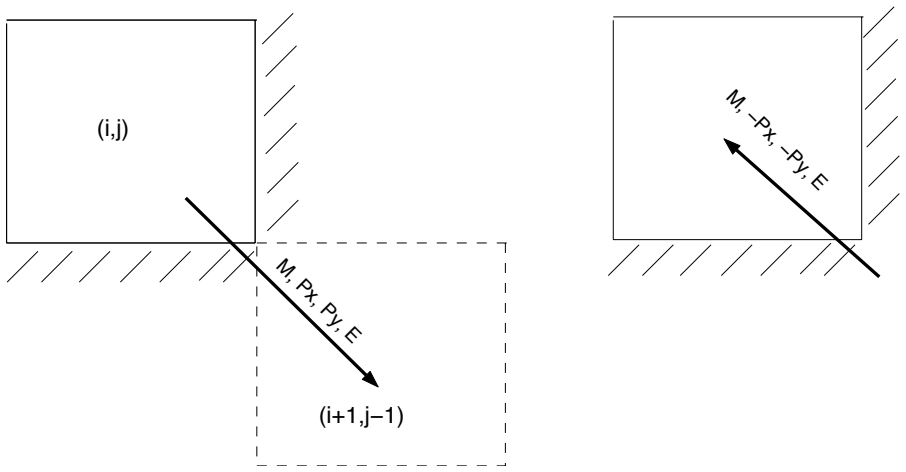

Figure 8.9: Diagram showing flux treatment with the source cell cornered by a specularly reflecting surface.

the source cell as per normal. Before the flux is inserted into the destination cell, the solver examines the physical geometry in the immediate vicinity of the source and destination cell $i+1, j-1$. First, the y-component of the momentum flux and its destination cell is treated. The solver reverses the component of momentum normal to the y coordinate and changes the destination cell to $i+1, j$. Following this, the solver checks the modified destination cell and concludes the flux still falls within a body region. The x-component of momentum and destination cell is then treated: the new destination cell is now i, j. The solver checks again to ensure the destination cell does not fall within the body and the destination flux is then added to the revised destination cell.

The simulation of flow around corners is another issue to have drawn attention in the past [47]. Flux integrals from a source cell i, j to a diagonal cell $i+1, j+1$ which is partially obscured by a body cell at $i+1, j$ possess no analytical solution to treat the partial refection of the transported fluxes expect for special cases (i.e. where the bulk velocity is zero). While a number of approximate methods exist to attempt to correctly capture this feature, this work treats the corner in a fashion consistent with existing direction decoupled techniques and reflects fluxes to cell $i, j+1$. Comparison of the results obtained using this treatment against allowing the flux to travel undisturbed to cell $i+1, j+1$ reals no major influence on the simulation result, including symptoms associated with the Carbuncle phenomenon. Therefore, while it is doubtful that this treatment is responsible for such failings as the Carbuncle phenomenon, the diffusive nature of the TDEFM fluxes makes a conclusive treatment impossible.

The treatment is fundamentally different if ghost cells are employed. Fluxes entering into ghost cells are not individually treated like previously demonstrated in Figure 8.9. Indeed, any fluxes leaving either real cell or a ghost and entering a ghost cell are ignored completely. These fluxes of mass, momentum and energy are subtracted from the source cell and then destroyed. The fluxes which would ideally be reflected back into the source cell are generated from the ghost cell by controlling the conditions. Referring to figure 8.9, if a ghost cell was employed it

would possess the same density and temperature as cell i, j. However, while the magnitudes of V_x and V_y would be identical, the directions would both be opposite. The ghost cell in cell $i+1, j$ would only have the direction V_x reversed while the ghost cell in cell $i, j-1$ would have the direction of V_y reversed. This ensures the same final result as the individual treatment covered above.

The diffusely reflecting boundary condition has no relevance in TDEFM as a Euler solver. Any attempt to use a diffusely reflective condition will result in a boundary layer thickness determined from the artificial diffusion which is a strong function of cell size [65, 1, 2]. However, by controlling this artificial diffusion through adaptive mesh refinement, the local cell size can be related to the local mean free path thereby approximating the artificial diffusion as the correct physical dissipation [97]. A diffusely reflective boundary condition may be meaningfully implemented as part of this proposed approximate viscous/rarefied solver.

8.3.3 Implementation on an adaptive grid

The implementation of TDEFM using Adaptive Mesh Refinement (AMR) is more involved than the implementation on a uniform cartesian grid. The general flow of the program, shown in Figure 8.10, has not changed significantly. The current implementation of TDEFM on an adaptively refined mesh does not currently have the ability to calculate conventional direction decoupled fluxes. Due to the unstructured nature of the problem, information contained in computational cells is contained in a one dimensional array. The address of each cell is maintained in a single cell index. Therefore, while the conceptual flow of the program is almost identical, the implementation is quite different. The calculation of flows across boundaries is handled through the use of ghost cells which are discussed later.

TDEFM flux calculation

Due to the employment of adaptive mesh refinement knowledge of neighbouring cell size and location cannot be incorporated easily into the flux expressions themselves. While a set of situation specific flux expressions could be created and employed for different possible source cell - destination cell combinations, it is far easier (although computationally expensive) to simply calculate the full TDEFM flux. Referring to Figure 8.11, the net flux of mass, momentum and energy to move from the source region to the destination region is:

$$
\begin{aligned}
M &= M_0 \mathbf{f}_{\mathrm{M}}(\mathrm{U}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \times \mathbf{f}_{\mathrm{M}}(\mathrm{V}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{y_R}, \mathrm{y_L}, \mathrm{y_l}, \mathrm{y_r}) \\
P_x &= M_0 \mathbf{f}_{\mathrm{P}}(\mathrm{U}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \times \mathbf{f}_{\mathrm{M}}(\mathrm{V}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{y_R}, \mathrm{y_L}, \mathrm{y_l}, \mathrm{y_r}) \\
P_y &= M_0 \mathbf{f}_{\mathrm{M}}(\mathrm{U}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \times \mathbf{f}_{\mathrm{P}}(\mathrm{V}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{y_R}, \mathrm{y_L}, \mathrm{y_l}, \mathrm{y_r}) \\
E_x &= M_0 \mathbf{f}_{\mathrm{E}}(\mathrm{U}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \times \mathbf{f}_{\mathrm{M}}(\mathrm{V}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{y_R}, \mathrm{y_L}, \mathrm{y_l}, \mathrm{y_r}) \\
E_y &= M_0 \mathbf{f}_{\mathrm{M}}(\mathrm{U}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \times \mathbf{f}_{\mathrm{E}}(\mathrm{V}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{y_R}, \mathrm{y_L}, \mathrm{y_l}, \mathrm{y_r}) \\
E &= E_x + E_y
\end{aligned}
$$

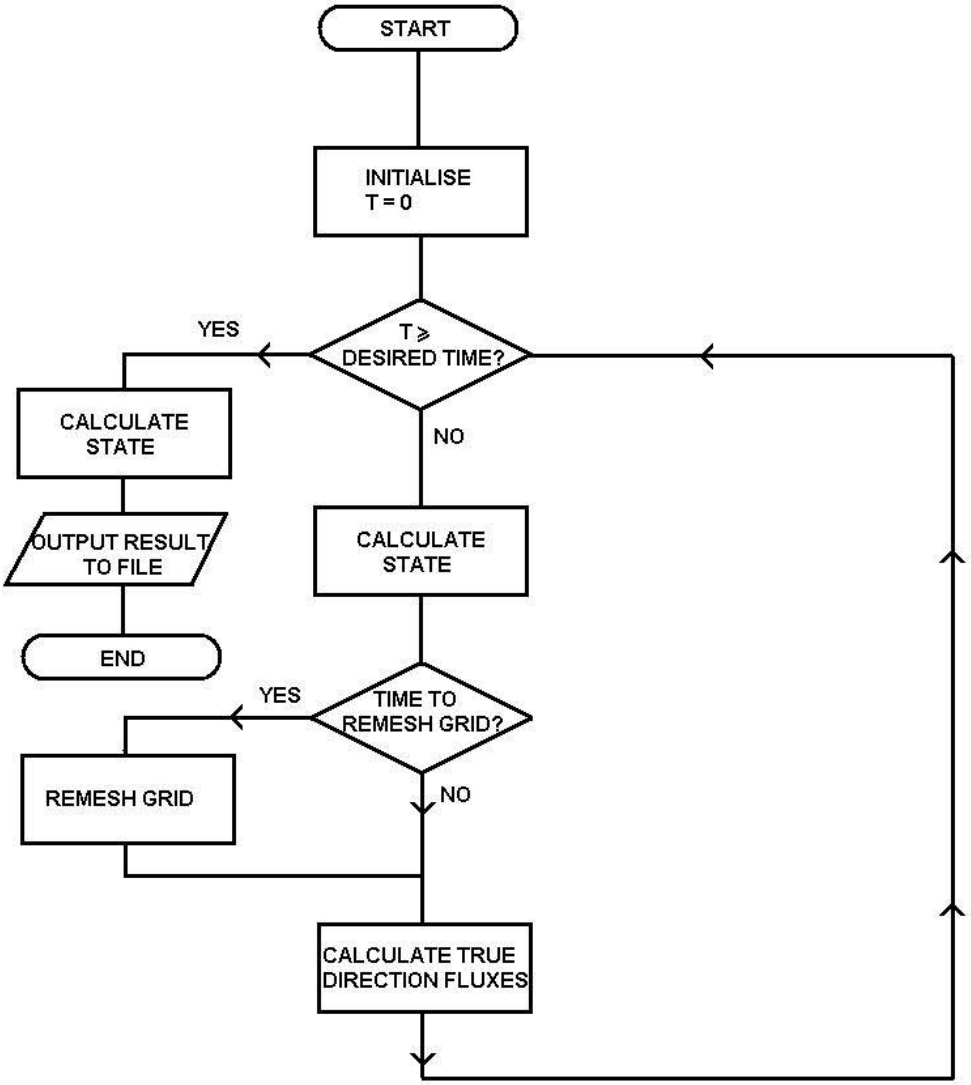

Figure 8.10: Program flowchart for adaptively refined computational grids.

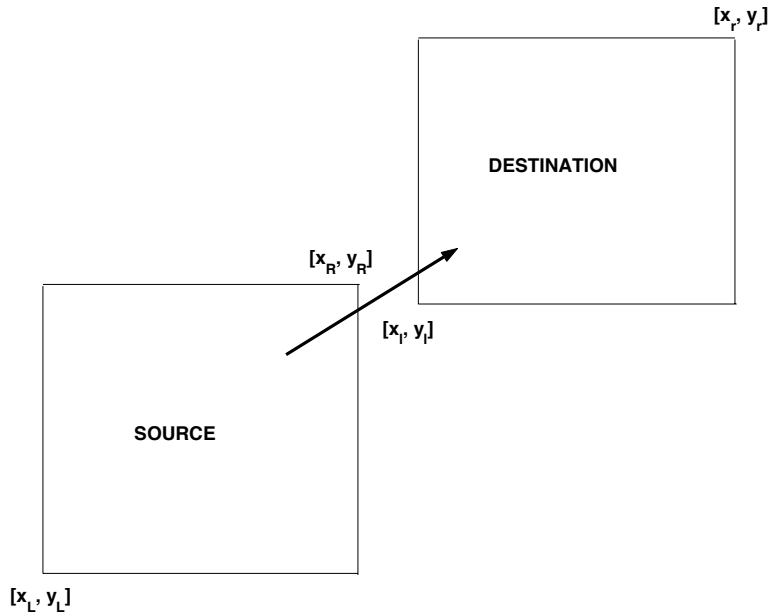

Figure 8.11: Sample source and destination cell geometry in 2D. The source cell is bounded by the coordinates $(x_L, y_L) - (x_R, y_R)$. The destination cell is bounded by the coordinates $(x_l, y_l) - (x_r, y_r)$.

where M, P and E are the net mass, momentum and energy fluxes respectively, M_0 is the initial mass in the source region, and $([x_L, y_L], [x_R, y_R])$ give the size and location of the rectangular source region, $([x_l, y_l], [x_r, y_r])$ describe the size and location of the destination region, U is the X velocity, V is the Y velocity, M is the net mass flux, P_x and P_y are the X and Y momentum fluxes and E is the energy flux. These fluxes of mass, momentum and energy represent the analytical fluxes where molecules belonging to a gas in thermal equilibrium are moved in free molecular flight. The destination region can be located anywhere in space and is not required to be adjacent to the source region.

For the extension to 3D, the process is very simple. The fluxes of mass, momentum and

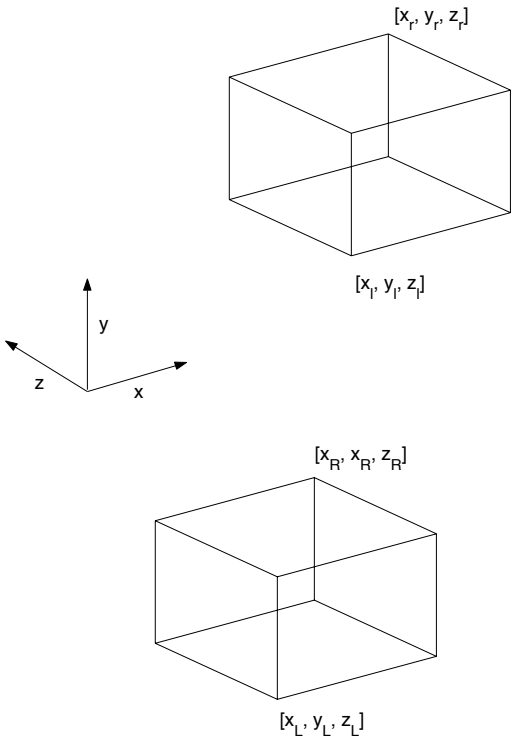

Figure 8.12: Sample source and destination cell geometry in 3D. The source cell is bounded by the coordinates $(x_L, y_L, z_L) - (x_R, y_R, z_R)$. The destination cell is bounded by the coordinates $(x_l, y_l, z_l) - (x_r, y_r, z_r)$.

energy from the source cell to the destination cell, shown in Figure 8.12, is:

$$
\begin{aligned}
M &= M_0 \mathbf{f}_{\mathrm{M}}(\mathrm{U}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \times \mathbf{f}_{\mathrm{M}}(\mathrm{V}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{y_R}, \mathrm{y_L}, \mathrm{y_l}, \mathrm{y_r}) \\
&\quad \times \mathbf{f}_{\mathrm{M}}(\mathrm{Z}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{z_R}, \mathrm{z_L}, \mathrm{z_l}, \mathrm{z_r}) \\
P_x &= M_0 \mathbf{f}_{\mathrm{P}}(\mathrm{U}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \times \mathbf{f}_{\mathrm{M}}(\mathrm{V}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{y_R}, \mathrm{y_L}, \mathrm{y_l}, \mathrm{y_r}) \\
&\quad \times \mathbf{f}_{\mathrm{M}}(\mathrm{Z}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{z_R}, \mathrm{z_L}, \mathrm{z_l}, \mathrm{z_r}) \\
P_y &= M_0 \mathbf{f}_{\mathrm{M}}(\mathrm{U}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \times \mathbf{f}_{\mathrm{P}}(\mathrm{V}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{y_R}, \mathrm{y_L}, \mathrm{y_l}, \mathrm{y_r}) \\
&\quad \times \mathbf{f}_{\mathrm{M}}(\mathrm{Z}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{z_R}, \mathrm{z_L}, \mathrm{z_l}, \mathrm{z_r}) \\
P_z &= M_0 \mathbf{f}_{\mathrm{M}}(\mathrm{U}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \times \mathbf{f}_{\mathrm{M}}(\mathrm{V}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{y_R}, \mathrm{y_L}, \mathrm{y_l}, \mathrm{y_r}) \\
&\quad \times \mathbf{f}_{\mathrm{P}}(\mathrm{Z}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{z_R}, \mathrm{z_L}, \mathrm{z_l}, \mathrm{z_r}) \\
E_x &= M_0 \mathbf{f}_{\mathrm{E}}(\mathrm{U}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \times \mathbf{f}_{\mathrm{M}}(\mathrm{V}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{y_R}, \mathrm{y_L}, \mathrm{y_l}, \mathrm{y_r}) \\
&\quad \times \mathbf{f}_{\mathrm{M}}(\mathrm{Z}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{z_R}, \mathrm{z_L}, \mathrm{z_l}, \mathrm{z_r}) \\
E_y &= M_0 \mathbf{f}_{\mathrm{M}}(\mathrm{U}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \times \mathbf{f}_{\mathrm{E}}(\mathrm{V}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{y_R}, \mathrm{y_L}, \mathrm{y_l}, \mathrm{y_r}) \\
&\quad \times \mathbf{f}_{\mathrm{M}}(\mathrm{Z}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{z_R}, \mathrm{z_L}, \mathrm{z_l}, \mathrm{z_r}) \\
E_z &= M_0 \mathbf{f}_{\mathrm{M}}(\mathrm{U}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \times \mathbf{f}_{\mathrm{M}}(\mathrm{V}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{y_R}, \mathrm{y_L}, \mathrm{y_l}, \mathrm{y_r}) \\
&\quad \times \mathbf{f}_{\mathrm{E}}(\mathrm{Z}, \sqrt{\mathrm{RT}}, \Delta\mathrm{t}, \mathrm{z_R}, \mathrm{z_L}, \mathrm{z_l}, \mathrm{z_r}) \\
E &= E_x + E_y + E_z
\end{aligned}
$$

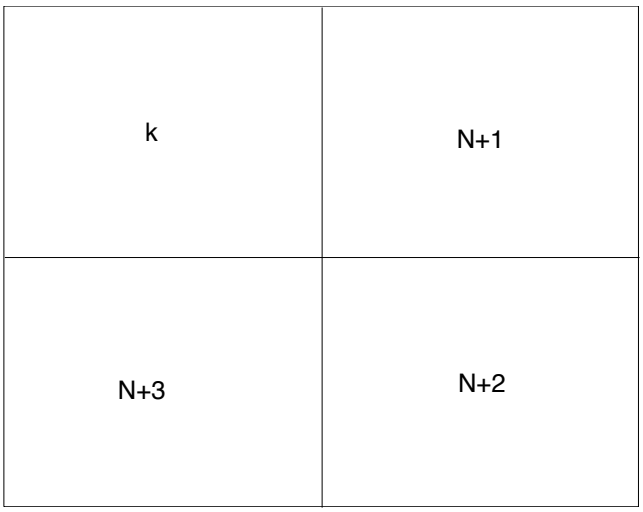

Figure 8.13: Placement of new cells in a isotropically split cell. Cell k is the original cell to be split, while cells $N+1, N+2$ and $N+3$ are newly created. N is the previous total number of cells.

Adaptive mesh refinement implementation

The use of Adaptive Mesh Refinement requires that a list of neighbouring cells be maintained. This information is kept in a two dimensional array. This list is updated while the mesh is adaptively refined. During the simulation initialisation, neighbours of source cells are found through the following routine:

1. Select a source cell

2. Search though destination cells (i.e. all cells except the source cell) and calculate distance between cell centers $R = [(c_{xd} - c_{xs})^2 + (c_{yd} - c_{ys})^2]^{0.5}$.

3. Compare to a desired neighbour radius R_d. If cells which are immediately adjacent (including diagonal cells) are desired then $R_d = [0.25((\Delta x_s - \Delta x_d)^2 + (\Delta y_s - \Delta y_d)^2)]^{0.5}$. R_d can also be a function of flow speed and radial location (if desired), though in this study only immediate neighbours are considered.

This is a lengthy procedure, and is performed during the initialisation of the simulation only. Fluxes of mass, momentum and energy are then calculated from each source cell to every neighbouring destination cell. At regular intervals, specified by the user, the qualities in each cell are examined and the decision on whether or not to split (or combine) a cell is reached. The procedure for isotropically splitting a cell with index k is:

1. Create 3 new cells with indexes of $N+1, N+2$ and $N+3$, where N was the previous number of existing cells (including ghost cells). The physical location of each new cell is fixed and shown in Figure 8.13.

2. Evenly distribute the mass, momentum and energy amoung cells k, $N+1, N+2$ and $N+3$.

3. Calculate the state in the newly created cells and regenerate the local neighbour list. A complete reconstruction of the neighbour list is not required - just a reconstruction of the cells which were previously neighbours of cell k.

4. Search through all neighbours of cell k. If the neighbour is a ghost cell and adjacent to cell k, it should also be split.

5. Update the total number of cells.

The procedure for combining cells is more complicated. However, this task is made easier by the strict rules used for splitting cells. Where a cell is deemed to be too small it is flagged for combination with other neighbouring cells. The procedure used when cell k is flagged for combination is:

1. Search through the neighbours of cell k. Locate cell N, the neighbour located directly adjacent and to the right of cell k.

2. Assign cells N, $N+1$ and $N+2$ as candidate cells for combination.

3. Search through the neighbours of cell k again, and make sure that all of cells N, $N+1$ and $N+2$ are amoung them. If not, then the cell cannot be combined with any of the surrounding neighbours.

4. Sum the mass, momentum and energy in cells k and $N, N+1, N+2$ and assign it to cell k. Adjust the cell location and volume accordingly.

5. Adjust the local neighbour lists of cells k and $N, N+1, N+2$.

6. Flag cells $N, N+1$ and $N+2$ for deletion.

When all recombinations are complete, all of the cells flagged for deletion are removed from the array. The above procedure ensures that cells are combined in the order that they were split. By doing so, this ensures that the cells do not suffer the staircasing (or bricklaying) effect discussed by Ham *et. al* [37]. This is demonstrated in Figure 8.14 showing cell 1100 and its neighbours following a mesh reconstruction. While cells 488, 1094 and 700 were re-combined properly, cells 461 and 1099 incorrectly chose neighbours for reconstruction and thus are staggered. Once local staggering occurs, the local cells cannot combine further with other surrounding cells due to their lack of alignment.

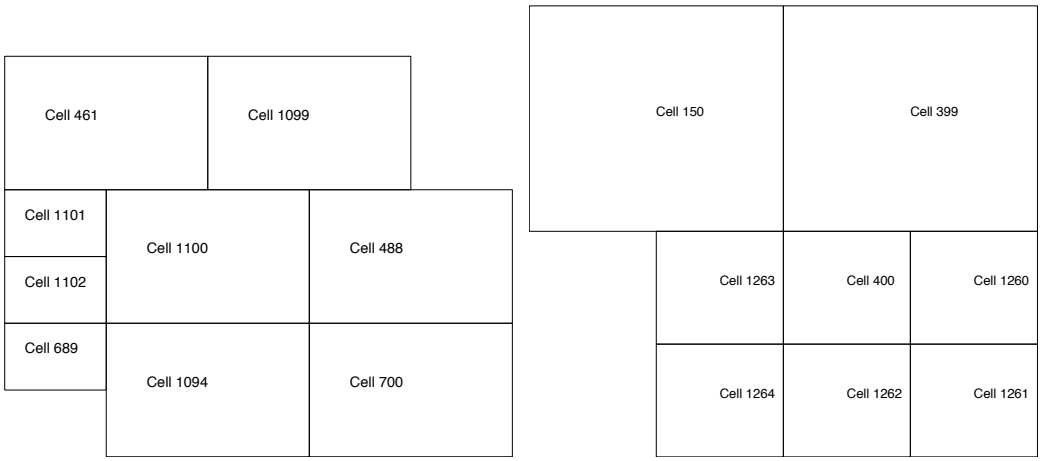

Figure 8.14: [Left] Example of a poorly reconstructed grid. While cells 1100, 488, 1094 and 700 were recombined correctly, cells 461 and 1099 were recombined incorrectly resulting in a staggering of the computational grid.[Right] Correctly reconstructed grid showing cell 400 and its neighbours.

Boundary condition implementation

Boundary conditions are managed through the use of ghost cells. These ghost cells behave differently depending on what type of surface they represent, and thus are further categorised. The types of boundaries used in this study in conjunction with adaptive mesh refinement are specularly and diffusely reflecting surfaces, inflow and extrapolated outflow. The inflow and outflow boundary ghost cells are identical in principle to the ghost cells used in the uniform grid implementation - the ghost cells behave as 'reservoirs' from which fluxes of mass, momentum and energy are calculated. Any fluxes calculated into these ghost cells are disregarded and not used. Since these ghost cells (reservoirs) are two or three dimensional, the complete TDEFM flux expressions are used to calculate fluxes into the flow region. This procedure is outlined in the flowchart presented in Figure 8.15.

The treatment of diffusely reflective surfaces is fundamentally different to the previous surface implementations. The distribution function for diffusely reflected molecules is not the same as the equilibrium distribution function used in the derivation of the conventional TDEFM flux expressions. In addition, the diffusely reflected fluxes result from integration of a velocity distribution function from a surface rather than a volume. Therefore, regions which are diffusely reflective are not considered as 'reservoirs' as the outflow and specularly reflective ghost cells are.

Instead, the diffusely reflective ghost cells only serve to hold information regarding the mass passed from real (internal cells) through the diffusely reflective surfaces of the simulation region. This approach is only approximate - alternatively, conventional direction split fluxes could be used to calculate the flux instead. However, for small values of Δt these fluxes has been shown to be equivalent [93]. The momentum and energy (per unit mass) reflected from the surface is known (Equations 8.34), so the mass collected in these ghost regions is used to calculate

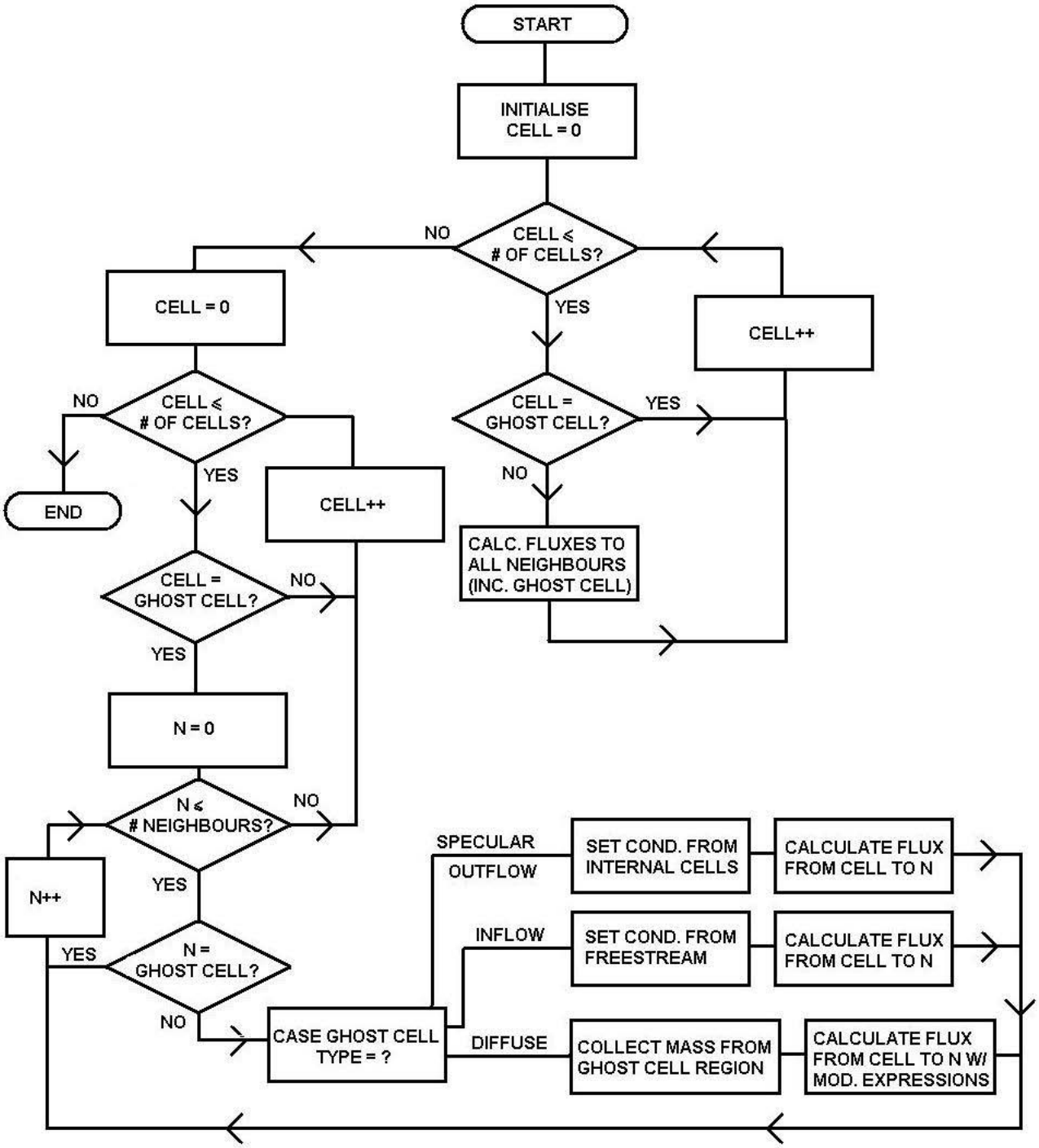

Figure 8.15: Program flowchart for treatment of boundary conditions in adaptively refined meshes.

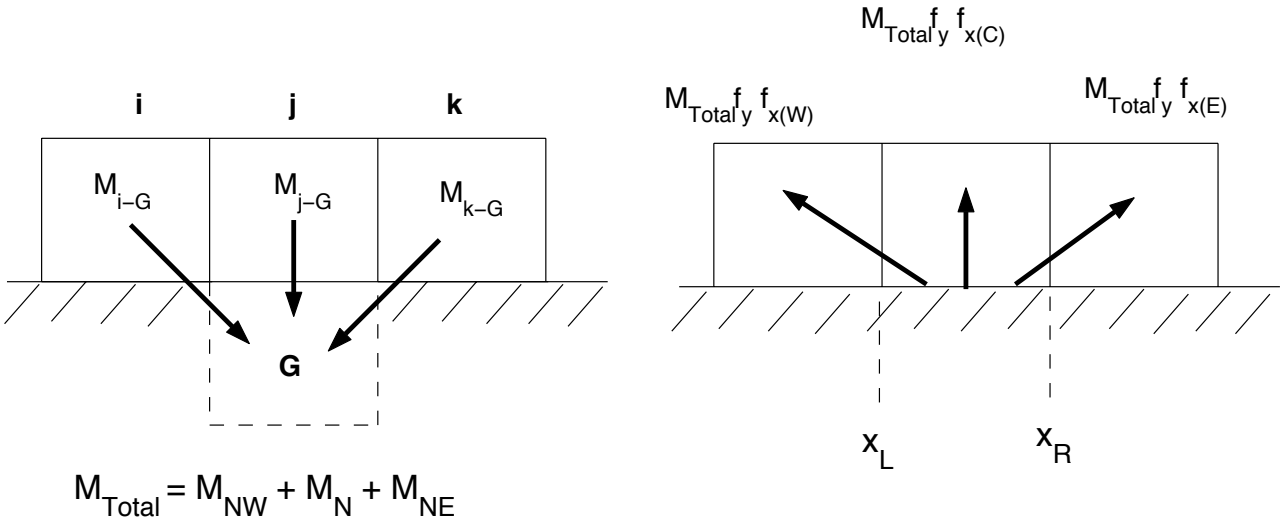

Figure 8.16: Example of a diffusely reflecting surface with fluxes calculated to its nearest neighbours.

the total momentum and energy transfered. Following this, all of the mass, momentum and energy is returned to the flow. Using this approach there is no chance of losing mass through a diffusely reflective surface.

This is explicitly demonstrated using the cell layout shown in Figure 8.16. First, fluxes from real cells i, j and k into the ghost cell G are calculated using standard TDEFM flux expressions. These masses, M_{i-G}, M_{j-G} and M_{k-G} are totaled to provide the total mass transfer into cell G M_{Total}. This mass is estimated as the total mass to pass through the surface separating cells G and j, an approximation only valid in the limit of a zero time step. The reflected mass flux from this surface to cell i is given by $M_{Total} f_y f_{x(W)} = M_{Total} f_M(i)$. The equation for $f_M(i)$ is given in Equation 8.34 under the assumption that $f_y = 1$. The fraction of reflected flux to remain above the surface $f_{x(C)} = 1 - (f_{x(W)} + f_{x(W)} = 1 - (f_{M(i)} + f_{M(k)})$ such that the sum of reflected fluxes is equal to the net mass flux M_{Total}, thereby ensuring that no mass is lost through the diffusely reflective surface. Fluxes of momentum and energy are carried out in similar fashion.

8.3.4 Implementation with arbitrarily shaped boundaries

Engineering problems which require CFD analysis are rarely restricted to bodies of rectangular shape. Therefore, it is important for a CFD solver to have the capacity to simulate flows around bodies or arbitrary shape. A common strategy is to employ an unstructured triangular mesh which allows simulation of complex shapes. Unfortunately, the use of a non-rectangular destination region means that the integrals used to evaluate the TDEFM flux expressions posses no closed solution. A number of approximations could be made which enable the flux expressions to be evaluated: however, these result in a reduced accuracy and a greatly increased computational expense. These alternatives lie outside of the scope of this thesis.

Fortunately, an increasing amount of effort has been spent investigating cartesian meshing

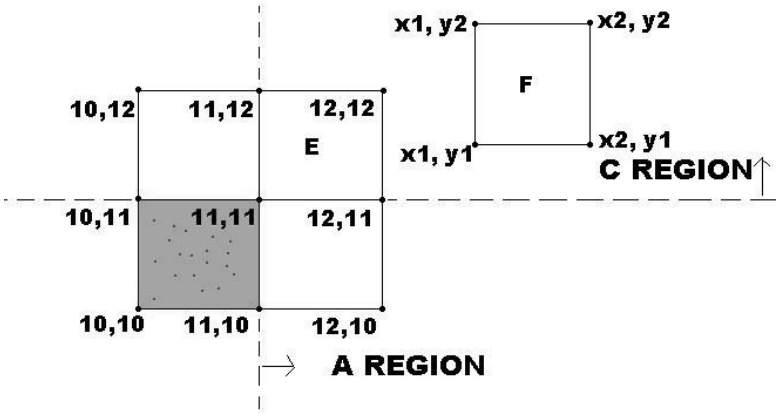

Figure 8.17: Setup for the numerical validation of the TDEFM fluxes. Region A is defined as the entire region where $x > 11$, while region C is $y > 11$. The bulk velocities in the x and y direction are mx and my respectively. Region E is the cell diagonally adjacent from the central, shaded cell. Region F is located at any point in space defined by x_1, x_2, y_1 and y_2.

methods around non-rectangular bodies [49, 50]. Since the computational grid of TDEFM is not required to be aligned with the flow the use of domain mapping onto the grid is easily implemented. Any computational cells that fall within a body are designated as body cells and the specular reflection routine discussed in Section 8.3.2 can be employed. The consequence of such a method is that angles surfaces and curved surfaces are resolved in steps and consequently the quality of resolution at the surface is reliant upon cell size. The use of an adaptively refined grid, refined near the surface of each body, improves resolution at the boundary and reduces errors due to stair casing of curved or angled boundaries.

8.4 Validation of Flux Expressions

Direct simulations run in MATLAB were used to verify the TDEFM fluxes. n simulation particles with unit mass were placed a region of unit volume. The mean velocities m_x and m_y in the central region were set to unity as was the time step. Particles were moved through a single time step, and no particle interactions were allowed to occur. Figure 8.17 shows the central (shaded) region located between $10 < x < 11$ and $10 < y < 11$. The fractions of mass, energy and momentum were calculated for regions A, C, E and for the general region F, which can exist anywhere in the flow field and does not need to share a common interface with the shaded region for mass, momentum or energy transfer to occur. As the number n of simulation particles increases, the fluxes calculated through direct simulation are expected to rapidly approach the TDEFM fluxes derived. Defining region F by setting $x_1 = 13, x_2 = 14, y_1 = 13$ and $y_2 = 14$, we can simulate a cell diagonally adjacent to the shaded region, although we could have chosen any region in the flow with equally successful results. If the mass flux fraction into region F calculated by TDEFM is f_{MF}^T and the fraction obtained from direct simulation is f_{MF}^S, then $\lim_{n\to\infty}\left(f_{MF}^T - f_{MF}^S\right)$ should equal zero. This can be shown in Table 8.1. The mean

n	mean($f^T_{MF} - f^S_{MF}$)	var($f^T_{MF} - f^S_{MF}$)
100	2.985e-3	2.944e-4
1000	4.414e-3	8.493e-5
10,000	7.043e-4	4.392e-6
100,000	1.762e-5	3.374e-7

Table 8.1: The absolute value of mean difference and the variance of the difference between analytically calculated and directly simulated mass flux fractions. Both the mean and the variance of the difference, defined as $f^T_{MF} - f^S_{MF}$, can be seen to approach zero as the number of simulation particles increases. f^T_{MF} is the mass flux fraction calculated by TDEFM, f^T_{MF} represents the simulated mass flux fraction. Subscript MF represents the mass flux fraction into region F.

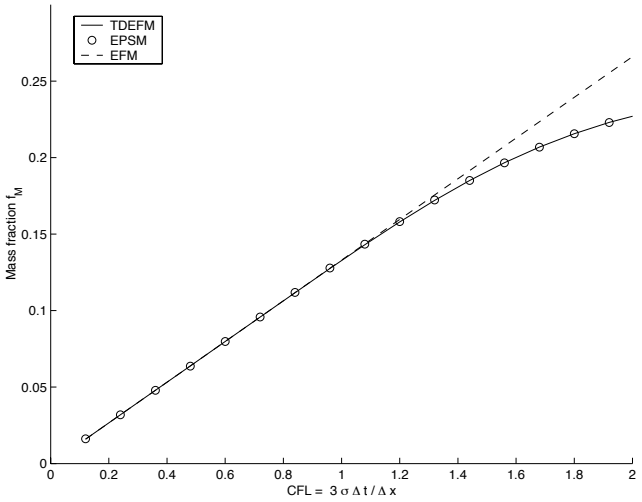

Figure 8.18: Mass flux (per unit source mass) calculated in the numerical validation of the TDEFM fluxes. The gas in the source region is at rest.

difference and variance between these fluxes are both decreasing as the number of simulated particles increases, implying that the fluxes calculated by TDEFM are correct.

The accuracy of the one dimensional TDEFM flux expressions with varying kinetic CFL numbers was also tested. A large number of simulation particles (500,000) were used to transfer mass, momentum and energy from a region of width Δx to an adjacent region of the same width. After each test, the kinetic CFL number was increased and the resulting fluxes recorded. The resulting mass fraction as a function of kinetic CFL number are shown in Figure 8.18. As the CFL number increases, the fraction of particles which are able to pass over the destination region increase, resulting in a decreased fraction of mass captured in the adjacent region. The EFM fluxes are constant with time since they assume complete capture by the destination region. The results from EPSM closely match those from TDEFM.

Chapter 9

FASTWAVE - A rapid response to blast wave threats

The effect of blast waves on buildings and structures has emerged as an important application of Computational Fluid Dynamics (CFD). Traditionally, these simulations are carried out by an expert user. These sophisticated calculation often involve the manual creation of a computational grid, followed by the use of high power computational facilities and manual post processing of results. The total time required, even when performed by an expert user on the most modern facilities, can easily extend into hours or even days. Therefore, the application of existing computational techniques and software to simulate blast waves resulting from an immediate threat or emergency is simply impossible.

The use of existing commercially CFD packages can be quite involved and time consuming. The flexibility and features offered by such modern packages, while making advanced analysis of flow problems possible, have undesired side effects. Expert knowledge is required by the user to decide, among other things, upon (i) solver selection, (ii) computational grid design, (iii) boundary conditions, and (iv) initial conditions. Each of these has a significant impact on the accuracy of the solution.

Traditionally, the procedure for performing a computational fluid dynamics calculation for simulation of blast waves around structures is:

1. Create the computational grid. This can be a slow and sophisticated process because the flow accuracy partially relies on the style and design of the grid. This step involves

   (a) Creating nodes for all important surfaces or regions. These include physical surfaces such as buildings in addition to non-physical regions such as the source region for the explosion.

   (b) Defining lines connecting all nodes and spatially discretising them.

   (c) Creation of computational blocks in which the initial conditions of the flow problem can be defined.

   (d) Creating the computational grid in each block. This can be done using unstructured or structured cells.

2. Definition of the initial conditions used for the simulations, including
   (a) Explosive regions, or the source of the blast wave,
   (b) Undisturbed regions of gas, including the interior of the buildings (if applicable).
3. Definition and selection of boundary conditions for each surface, i.e.:
   (a) Encapsulating boundaries,
   (b) Building-Fluid surfaces, and
   (c) Fluid-Fluid surfaces (often done automatically).
4. Selection of a CFD solver, and
5. Graphing of results for analysis.

Depending on the detail and type of simulation, the actual solution of the governing fluid flow equations may only require a fraction of the total time required. The development of the computational grid can be a lengthy and complicated process, even for simple flow problems. As an example, a typical computational grid used for such problems is shown in Figure 9.1. In this simple two dimensional simulation, there are a total of nine (9) simulated buildings and one high pressure source region emulating the effects of an explosion source. In this case the computational grid includes 160 surfaces or boundaries and 40 computational blocks. In a structured grid, each block consists of $x \times y$ cells and each neighbouring block must share a common number of cells along the connecting surface. The expert user then needs to attribute each of these surfaces with the appropriate boundary condition. Following this, the user then needs to apply the correct initial conditions to each computational block. After the selection of the solver, the equations governing the behaviour of the select fluid are solved. Finally, the results are graphed and analysis is performed. The entire procedure often requires the use of multiple pieces of software, requiring the user to close and open various programs during the development of the solution.

FASTWAVE is designed to encapsulate all of the above steps in a single application. Using this software, the average user can perform the above tasks in a fraction of the time without leaving the main program screen. The tedious tasks of edge discretisation and block creation are removed completely as the solver uses a single block encapsulating the entire flow region. The interfaces between fluid and structures are detected and handled in the solution engine itself, so no user intervention is required. The errors associated with the use of regular cartesian grids are reduced through the optional use of the true direction solver TDEFM (True Direction Equilibrium Flux Method). This family of solvers represent the analytical solution to a direct solver in the equilibrium limit using Godunov's monotone scheme, i.e. uniform conditions over a cell volume. While conventional finite volume solvers calculate fluxes across cell interfaces, TDEFM (and its higher order equivalents DTDEFM and VTDEFM) calculate the fluxes of

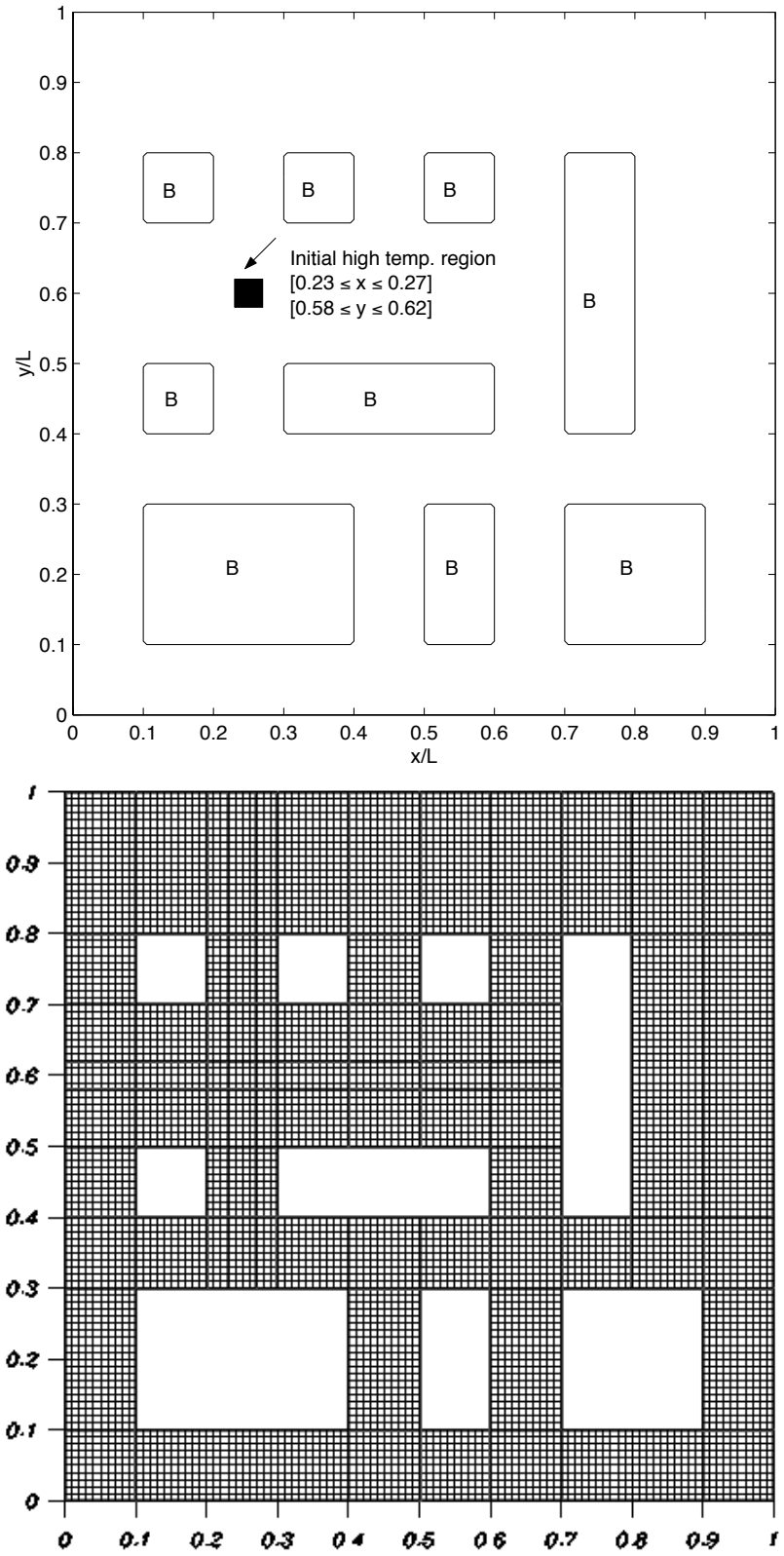

Figure 9.1: A typical computational grid used by the conventional commercially available CFD solver CFD-FASTRAN [57].

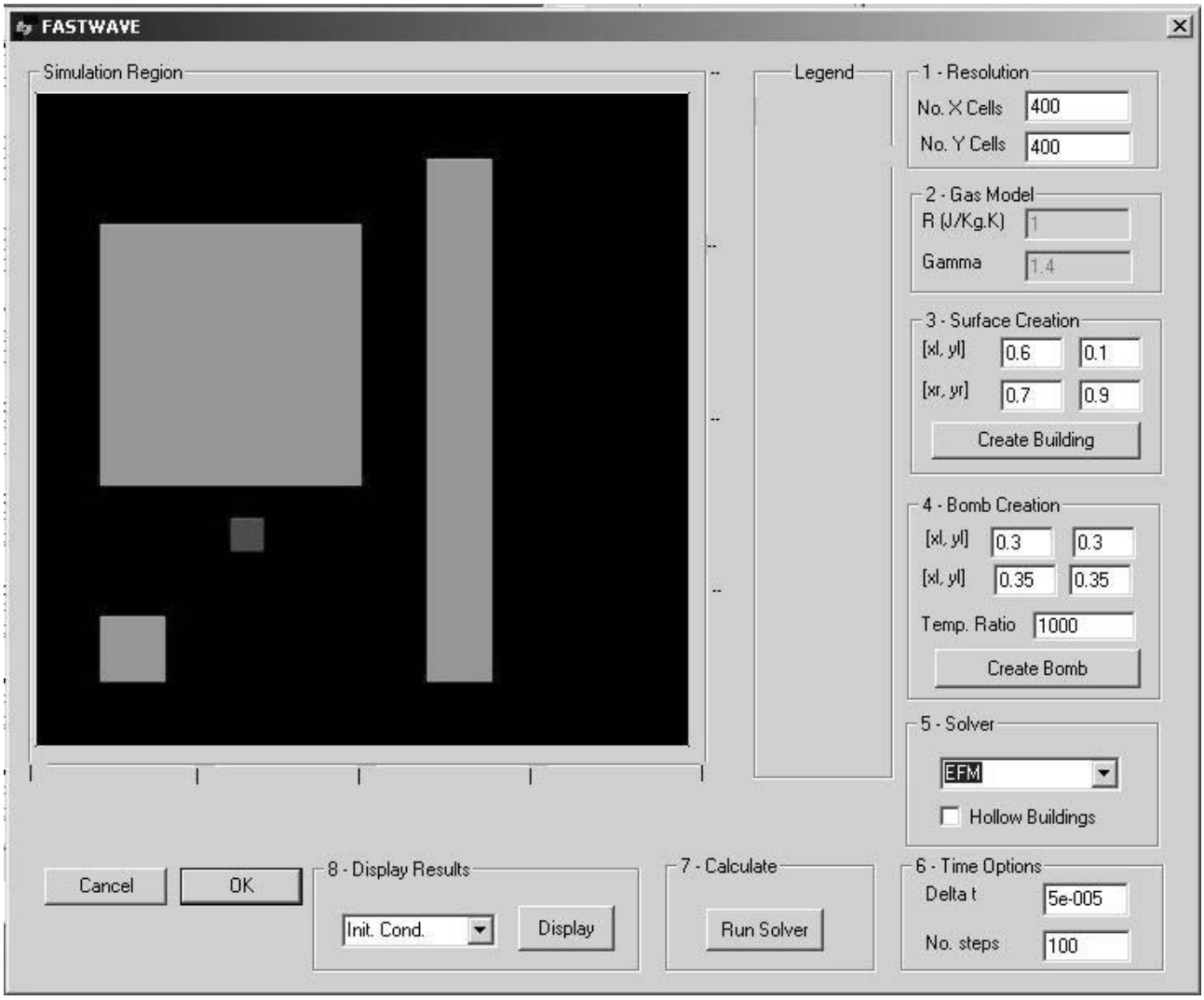

Figure 9.2: Layout of the FASTWAVE interface. The simulation region displays information regarding structure and explosion geometry. The control panels located on the right side of the interface contain variable settings for grid resolution, solvers and tools for entering structure and explosions. The lower section of the interface contains the display options, where different flow properties (Mach number, density ratio) are selected and graphed using filled colour contours.

mass, momentum and energy to all surrounding cells, not just those sharing an adjacent interface. When compared with conventional finite volume solvers, TDEFM has been shown to provide more accurate solutions on regular cartesian grids where flow is not aligned with the grid.

A screen shot of FASTWAVE is shown in Figure 9.2. The simulation region occupies a large fraction of the interface - it is here that the computational mesh, building and blast geometry and results can be viewed. To the right of this region are the control panels. The general procedure for solving a problem using FASTWAVE is demonstrated by stepping in order through the control panels:

1. Resolution - Enter the number of cells in the x and y coordinates.

Solver	Mesh Generation	Solution Initiation	Solving	Total
CFD-FASTRAN	42 mins	12 mins	28 mins	82 mins
FASTWAVE	1 min	30 sec	2 mins	3 mins 30 sec

Table 9.1: Times required to complete a blast wave simulation using FASTWAVE and CFD-FASTRAN [57]. All simulations were conducted on the same desktop computer.

2. Structure placement - enter the coordinates of any buildings involved in the simulation.

3. Explosive placement - enter the coordinates and the strength of the explosion. The default strength is 1000, meaning the temperature in the heated gas is 1000 times that of the surrounding ambient gas.

4. Solver selection - choose from Pullin's EFM, Jacobs approximate Riemann solver, Macrossan's PFM, or Smith's TDEFM, VTDEFM and DTDEFM. The default is EFM.

5. Time control - select the time step and number of steps for the solver the use.

6. Calculate - run the selected solver for the number of selected steps.

7. Display Results - load the results from file and display on the simulation region. The results will update at intervals of 20 percent of the total simulation time.

The results are written in tab delimited form to the local directory and can be viewed using MATLAB. The domain in FASTWAVE consists of a single block with the number of cells specified. The structures in FASTWAVE are constructed from normal cells and flagged as solid bodies. Fluxes are not calculated for flagged cells, and when a fluid cell is connected (or nearby, for true direction fluxes) and interaction is predicted, specularly reflective boundary conditions are employed. The specified body dimensions are used as a guide for the flagging of body cells. If any part of the cell occupies a region specified by the user as a building, the cell is flagged as a body. Therefore, the level of detail available for building definition is limited by the computational grid resolution. For the purpose of rapid calculation a coarse mesh of less than a million cells is acceptable.

The solvers available for use by FASTWAVE are Pullin's Equilibrium Flux Method (EFM) [80], Jacob's approximate Riemann solver [44], the True Direction Equilibrium Flux Method (TDEFM) and its derivatives DTDEFM and VTDEFM [93, 98, 95] and finally the Particle Flux Method (PFM) by Macrossan *et al* [70].

The times required to run a blast wave simulation using the city geometry in Figure 9.1 are detailed in Table 9. The grid resolution used by both CFD-FASTRAN and FASTWAVE was identical. CFD-FASTRAN is a popular commercially availble CFD package [57]. The CFD-FASTRAN solver used Runga-Kutta time stepping with a single stage (simple Euler). Spatial

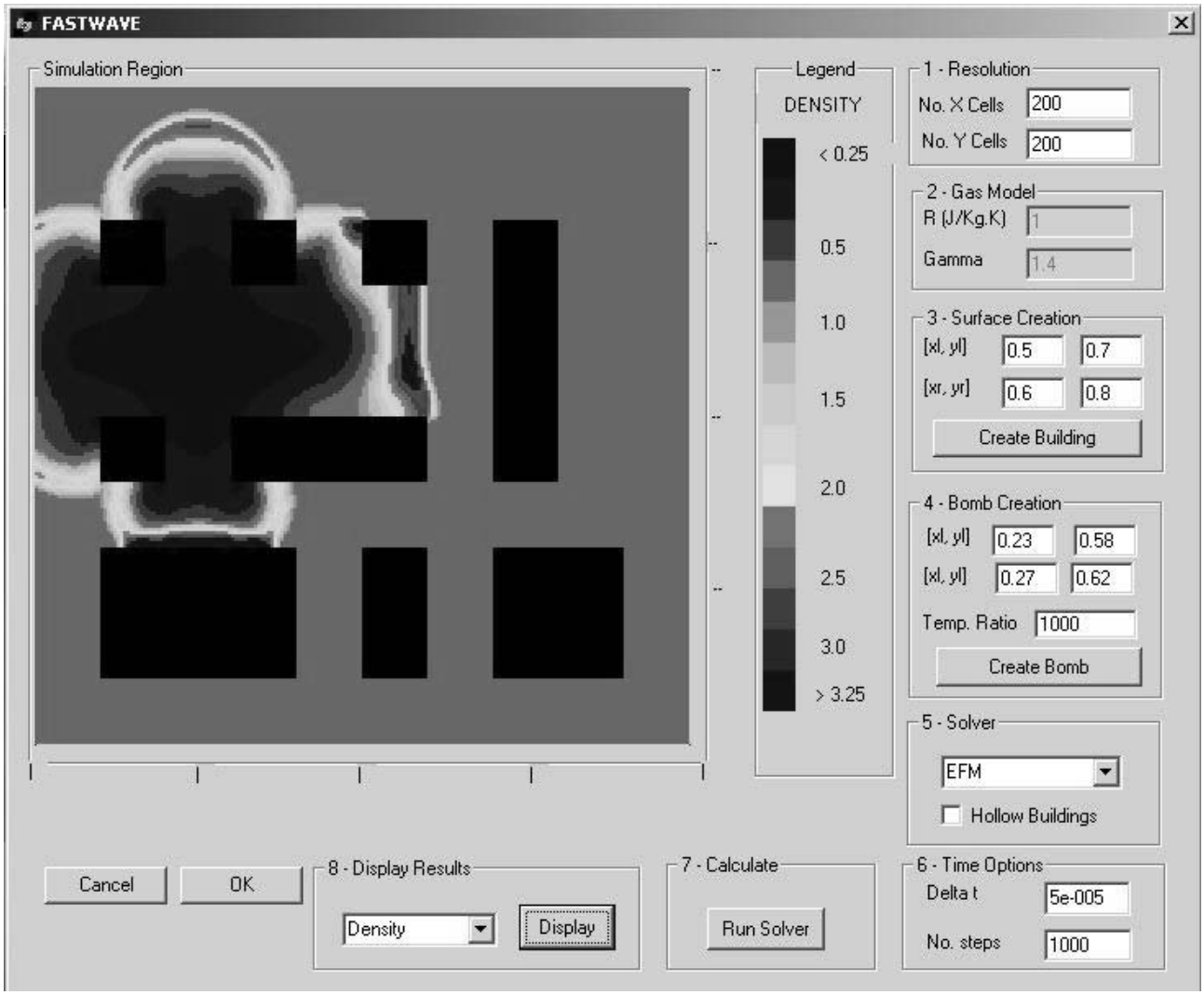

Figure 9.3: FASTWAVE interface showing the blast wave results for the initial conditions shown in Figure 9.1. Colour contours of density are shown in the simulation region. The display can be refreshed while the simulation is running to display the development of the flow. These results are saved as unique files at intervals of 20 percent of the total simulation time.

accuracy was kept at first order to increase calculation speed. The fluxes at cell interfaces were calculated using the Roe solver included with CFD-FASTRAN. An example of the graphical output capability of FASTWAVE is shown in Figure 9.3. The shown colour contours are of density and compare to that provided by the CFD-FASTRAN solution.

The current implementation of FASTWAVE uses an interface and source code capable of simulation in two dimensions only. This is a drawback due to the three dimensional nature of blasts. The extension to three dimensions provides a relieving effect on the gas thus restriction to two dimensions has the effect of overpredicting the pressure of the blasted gas as it flows through the city. However, as a preliminary tool for the estimation of the effect of a blast in a city environment, this overprediction may be acceptable. Extension to higher dimensions is possible in both the interface and the TDEFM solver itself, though falls outside the scope of this work.

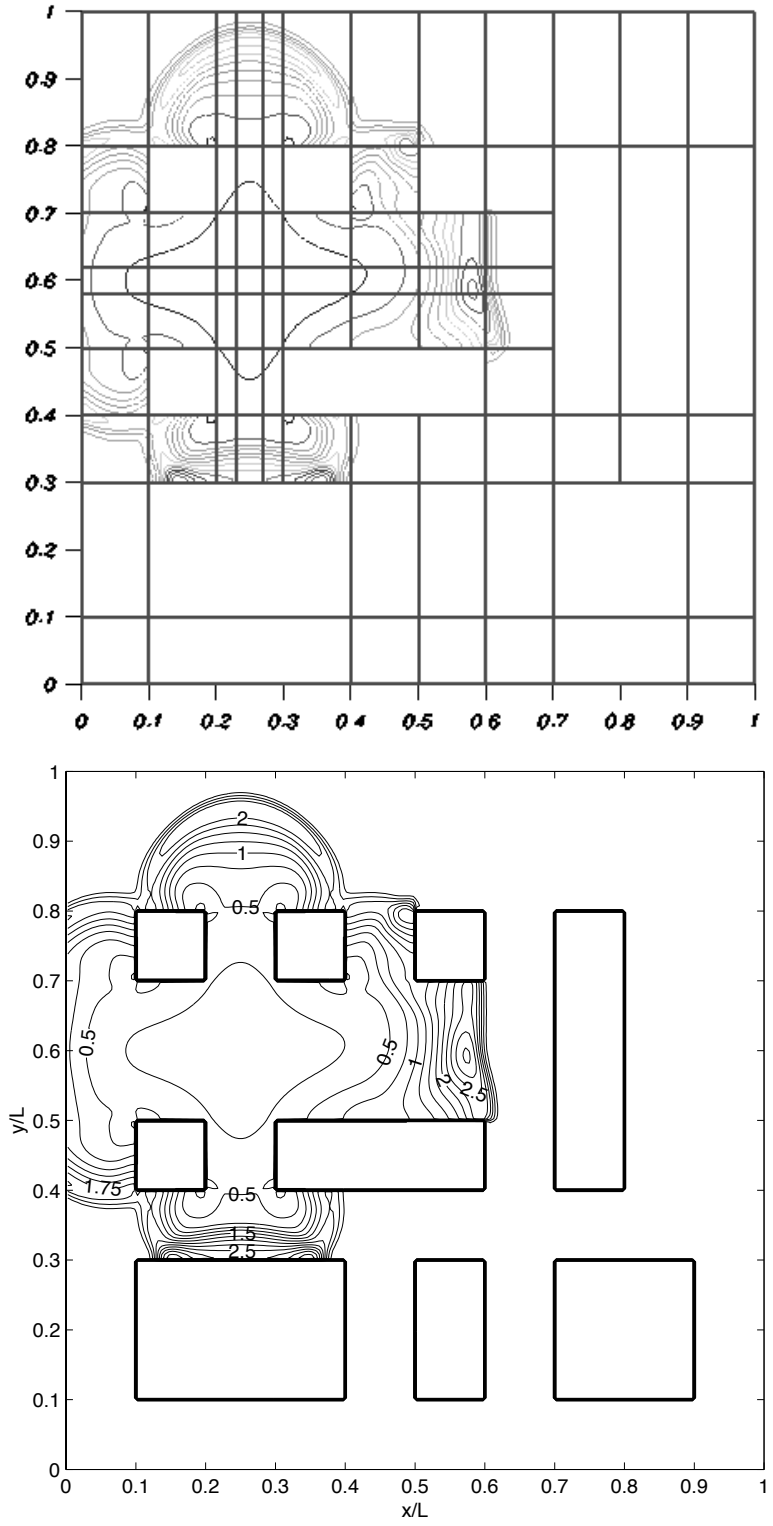

Figure 9.4: (Top) Density contours from the CFD-FASTRAN results and exported by CFD-VIEW. (Bottom) Density contours from the FASTWAVE results using the True Direction Equilibrium Flux Method (TDEFM). Both results use identical mesh density, time step size and initial conditions. All building surfaces are treated as reflective with no losses from friction. The gas is ideal with $gamma = 1.4$.

Chapter 10

Results

10.1 1D Shock Tube Problem

Results are presented for a standard 1D shock tube problem. A propagating shock is created through a pressure ratio across a diaphragm, which is removed at time $t = 0$. There is no temperature difference across the diaphragm, and the gas used is inviscid with $\gamma = 5/3$. After the diaphragm is removed, a propagating shock waves is expected, followed by the contact discontinuity and finally an expansion wave. The simulation is run using 200 time steps to a total time of $0.1L/(RT)^{0.5}$ on a uniform grid of 200 cells. Figure 10.1 shows density and temperature profiles from various solvers, with all results showing general agreement.

Figure 10.2 show a closer view of the propagating shock wave at $x/L = 0.9$. As expected, the results obtained from TDEFM closely match that of EPSM with the forced redistribution of mass. It is clear to see that the forced redistribution of mass in a cell causes the shock thickness to increase. Density TDEFM also more closely matches the density and temperature profiles of EPSM than does TDEFM. However, the simple inclusion of a linear variation of density is insufficient to fully recreate the results obtained from EPSM, although the method offers a significant improvement upon the existing TDEFM. This indicates that higher order distributions of density are present in the EPSM solution, meaning that the placement of flow features can be captured in regions of size less than a cell width.

Figure 10.3 shows the results from the same 1D shock tube problem using the proposed hybrid TDEFM-BGK method. The simulation is run using 200 time steps to a total time of $0.1L/(RT)^{0.5}$ on a uniform grid of 200 cells. The gas is ideal and monatomic with a power law viscosity using $\omega = 0.75$ The Knudsen number in the tube using the initial conditions on the low density side was $Kn = 0.02$ with the length of the tube used as the characteristic length.

Figure 10.4 shows the X Velocity distributions at varying locations through the propagating shock for the same shock tube problem. The mean values of each velocity distribution shifts at each location, indicating that the flow is accelerating through the shock wave. The variance of each distribution function is also increasing, meaning the temperature of the gas is increasing.

Figure 10.5 shows results from the 1D shock tube problem used by Li and Zhang [54]. The initial conditions are similar to that of Sod's 1D shock tube with an imaginary diaphragm separating gases of different conditions, specified below:

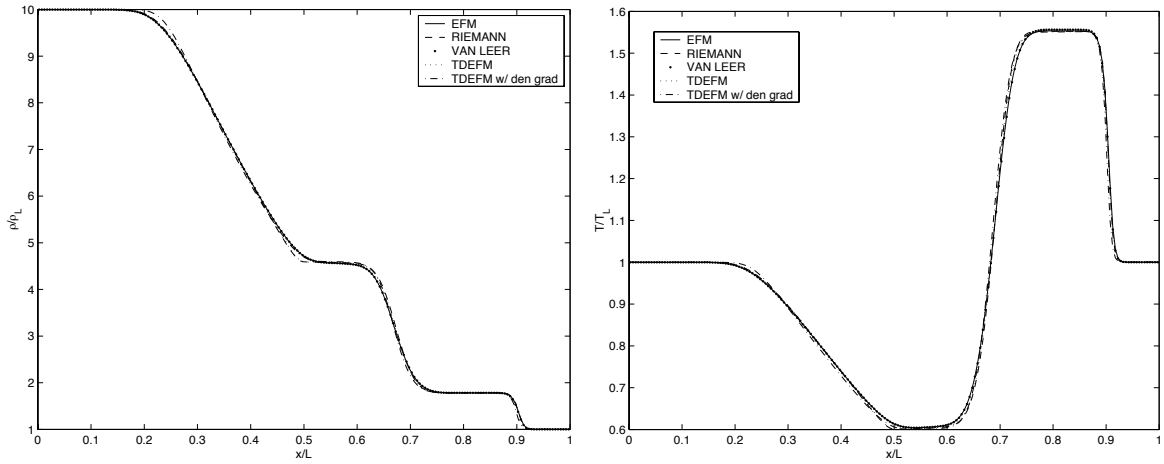

Figure 10.1: Normalised density (left) and temperature (right) profiles for the 1D shock tube problem from various continuum solvers. The gas is inviscid with $\gamma = 5/3$, and an initial density ratio of 10. The simulation is run using 200 time steps to a total time of $0.1L/(RT)^{0.5}$. 200 cells are used in all methods.

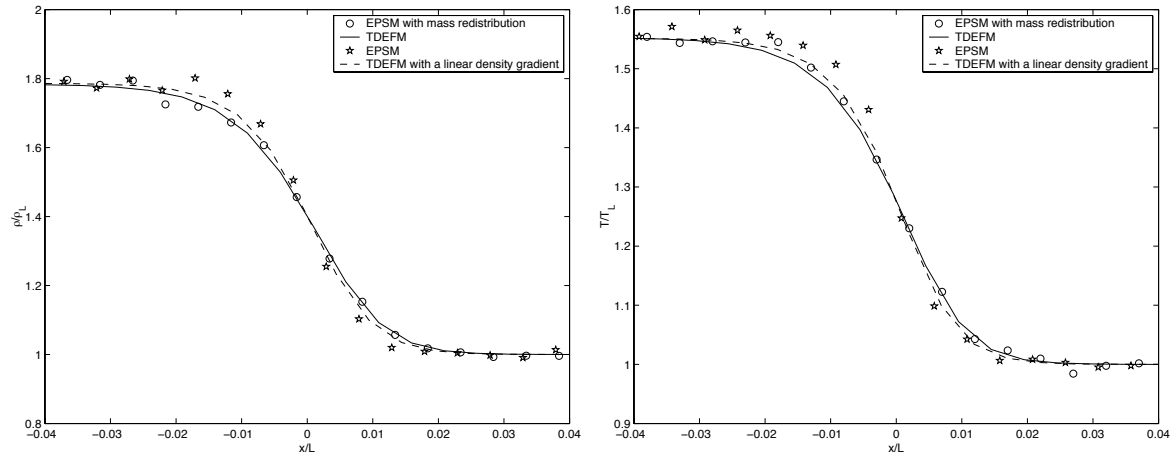

Figure 10.2: Comparison of density (left) and temperature (right) profiles from the 1D shock tube results using various direct and true direction continuum solvers. The gas is inviscid with $\gamma = 5/3$, and an initial density ratio of 10. The simulation is run using 1000 time steps to a total time of $0.1L/(RT)^{0.5}$. 200 cells are used in all methods. The EPSM solution uses 550,000 simulation particles with 20 runs to help reduce statistical scatter.

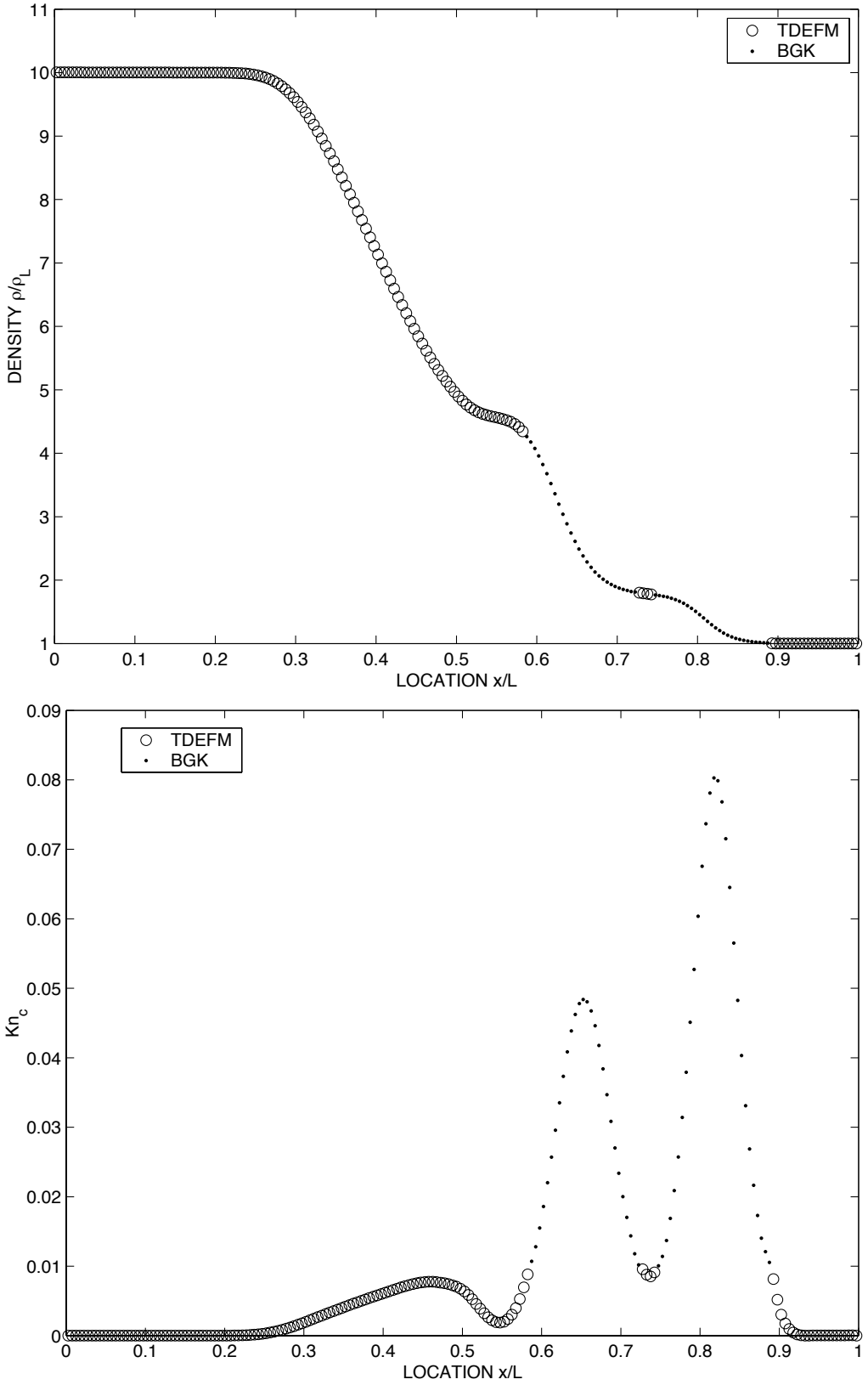

Figure 10.3: Normalised density (top) and local gradient length Knudsen number (bottom) for the 1D shock tube problem.

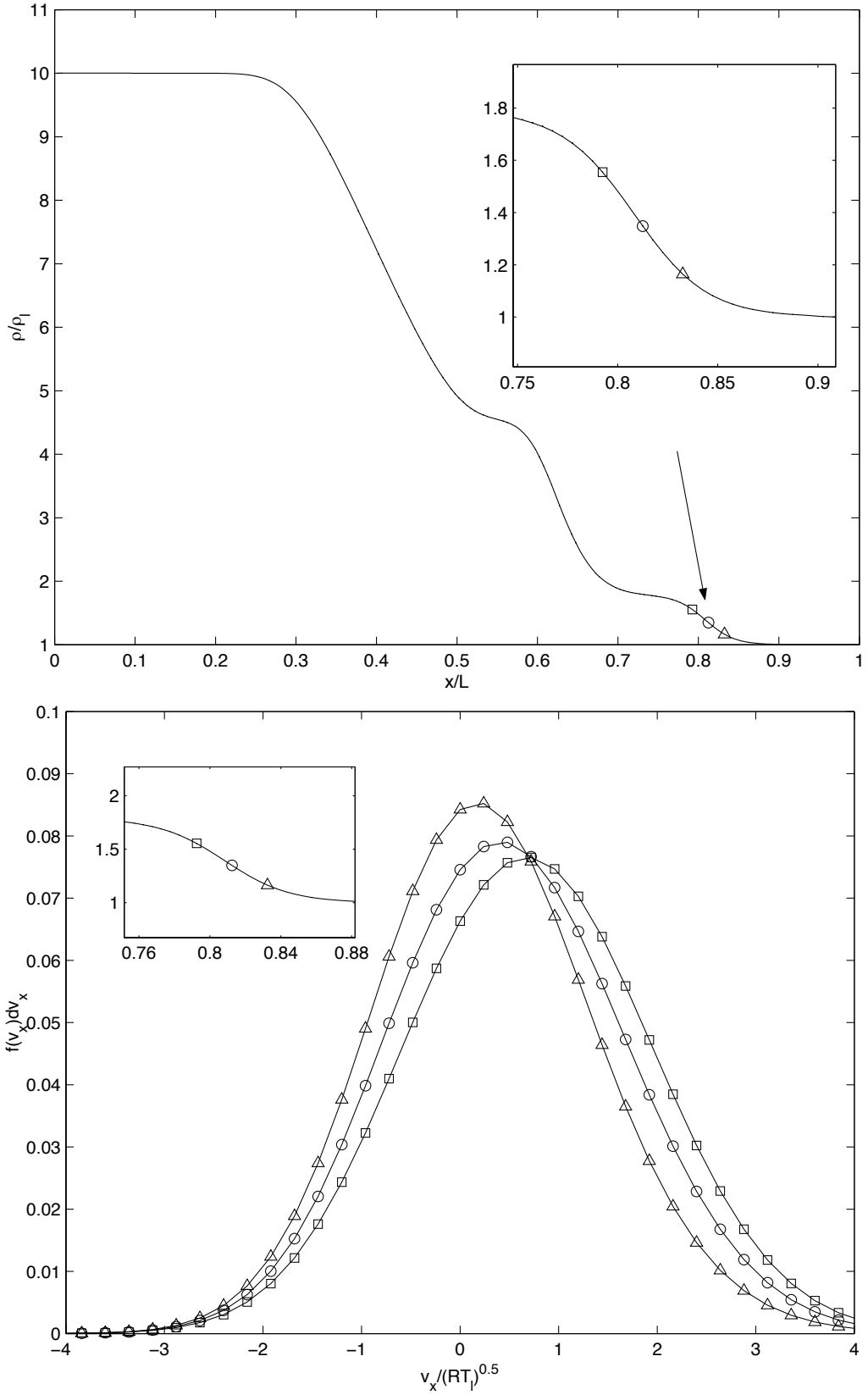

Figure 10.4: (Top) Density profile calculated using the TDEFM-BGK hybrid solver, (Bottom) X Velocity distribution functions calculated by the TDEFM-BGK solver at varying locations through the propagating shock wave.

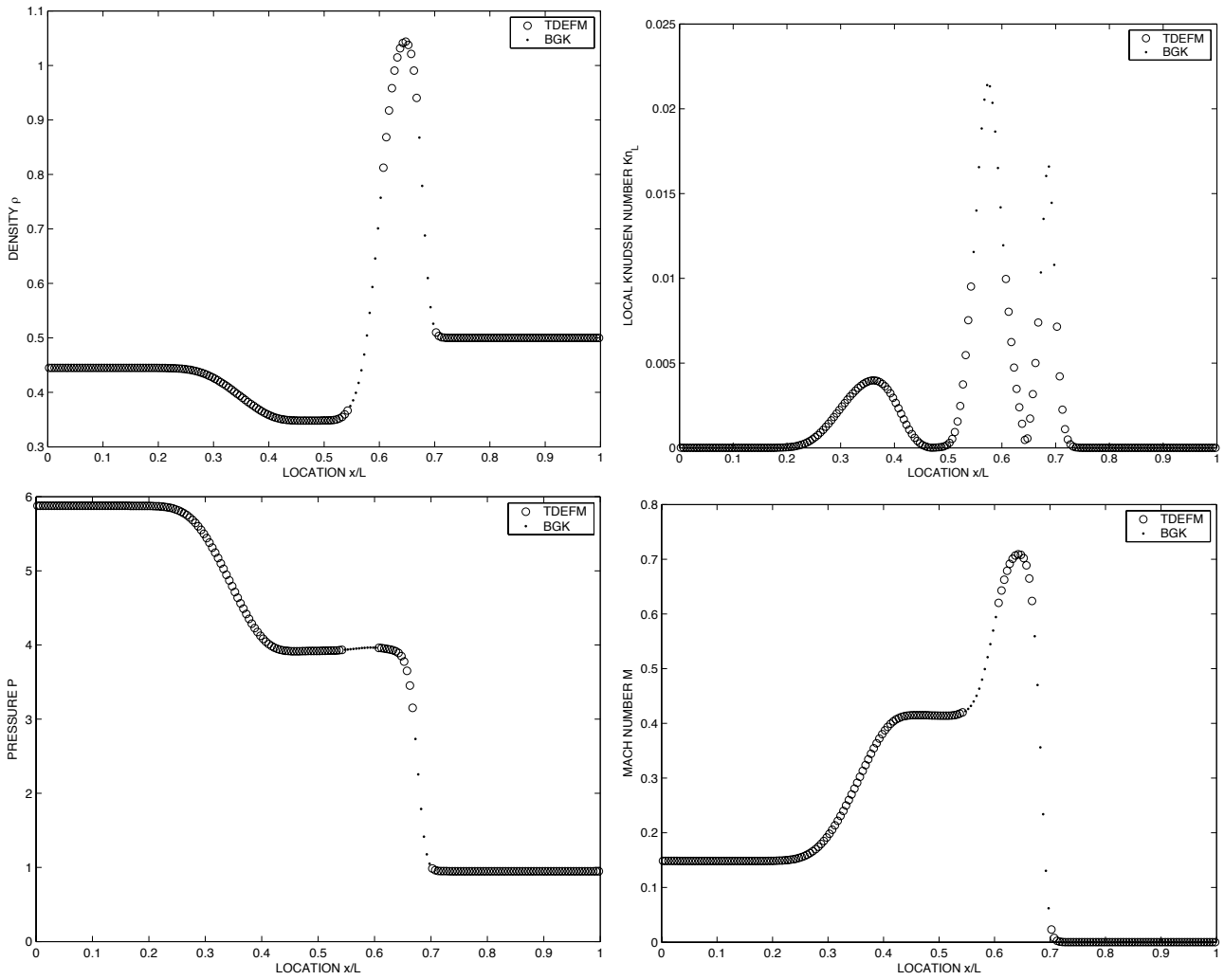

Figure 10.5: Normalised density (top) and local gradient length Knudsen number (bottom).

$$\rho = \begin{cases} 0.445, \mathrm{x} < \mathrm{L}/2 \\ 0.5, \mathrm{x} > \mathrm{L}/2 \end{cases} \tag{10.1}$$

$$T = \begin{cases} 13.21, \mathrm{x} < \mathrm{L}/2 \\ 1.9, \mathrm{x} > \mathrm{L}/2 \end{cases} \tag{10.2}$$

$$u = \begin{cases} 0.698, \mathrm{x} < \mathrm{L}/2 \\ 0.0, \mathrm{x} > \mathrm{L}/2 \end{cases} \tag{10.3}$$

Figure 10.5 shows the density, Mach number and temperature using a breakdown parameter of 0.01. Wherever the gradient length Knudsen number is less than this value, the BGK solution approach is used while TDEFM is used in all other regions. The CFL is restricted to ensure that the fastest moving velocity buckets from the BGK method can move no further than the adjacent cell. The results agree with the use of BGK throughout the entire region, while requiring only a fraction of the time.

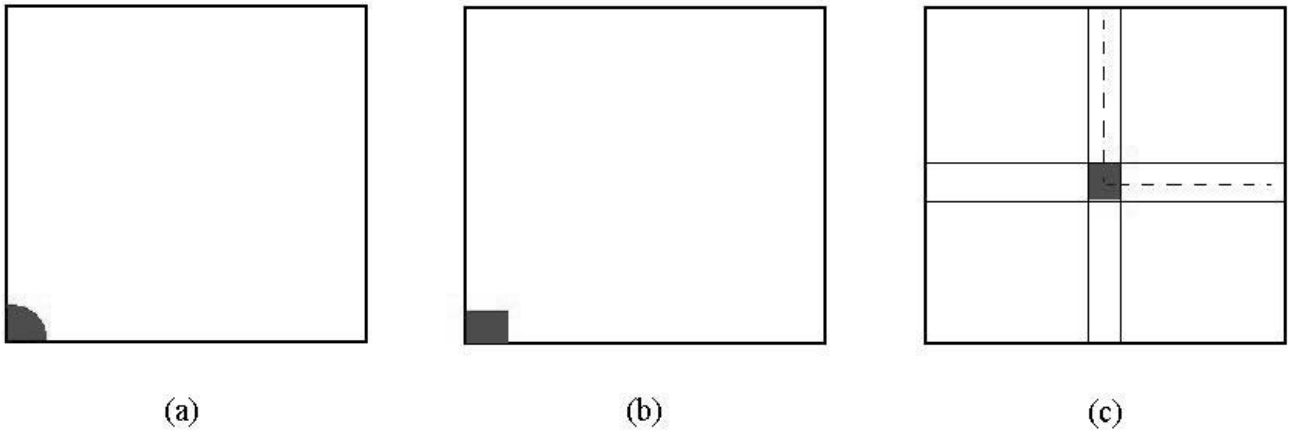

Figure 10.6: Blast wave geometry. (a) Ideal initial condition and geometry. (b) Geometry used by the solvers with 50x50 cells. (c) Geometry used by solvers with 100x100 cells. The symmetry boundary condition (specular reflection) was applied at all boundaries. Perfect gas with ratio of specific heats $\gamma = 5/3$. Initial conditions: $T_H/T_L = 1000$, $\rho_H/\rho_L = 1$ (pressure ratio $P_H/P_L = 1000$). Radius of high pressure region is r.

10.2 2D Blast Wave Problem

The flow field contains a two dimensional 'blast wave' caused by an initial small region with a temperature higher than the surrounding gas. One quarter of a square plane of unit width with symmetry condition applied on all four walls is used. The length of computational domain is $50r$ in each direction, where r is the radius of the high temperature region. The initial conditions are:

$$\begin{aligned} \rho_H/\rho_L &= 1 \\ T &= \chi T_H \\ \chi &= f + (1-f)\tfrac{T_L}{T_H} \\ U = V &= 0 \\ \gamma &= \tfrac{5}{3} \end{aligned} \tag{10.4}$$

where ρ_0 is the density, T_H is the temperature inside the ideal circular initial condition, T_L is the temperature outside. The fraction of the area of each cell inside the high temperature region is given by f, and is demonstrated in Figure 10.6. The ratio χ is used to ensure that, regardless of mesh density, the initial computational domain possesses the same total energy. This initial high temperature (and hence pressure) in the one cell simulates a sudden 'explosion' centered on the origin. Ideally, the resulting flow is radially symmetric. The unsteady simulation is run to time $t\sqrt{RT_L}/r = 0.00196$ where the expanding shock wave has traveled to just beyond $22r$. Although the method disregards viscous effects, the same numerical viscosity present in EFM is present in TDEFM.

The benchmark result is obtained from a 1D-EFM solution using the initial condition described in Figure 10.6(a). The length of the circular region was divided radially into 800 cells

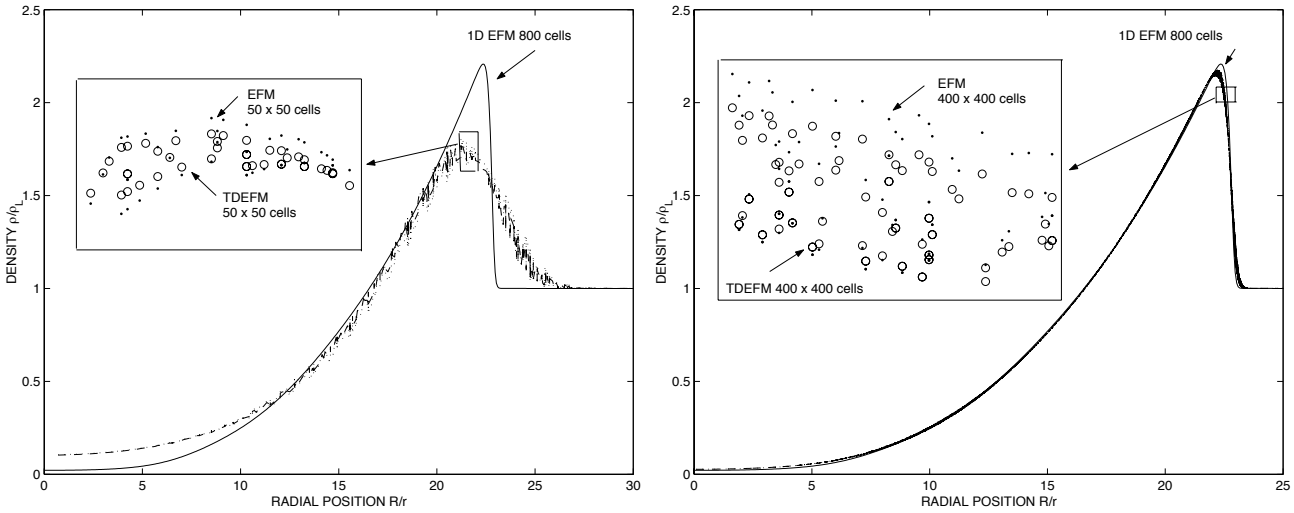

Figure 10.7: 2D solutions of the blast wave problem showing normalised density using 2D-TDEFM (∘) and 2D-EFM (·) using a 50x50 mesh (left) and a 400x400 mesh (right). The initial conditions as shown in Figure 10.6(b). The results from all individual cells are shown and should collapse into a single line. The solid line shows 1D results with 1x800 cells. Simulations are run up to $t\sqrt{RT}/r = 0.00196$. Sections of the results have been enlarged to better demonstrate the scatter present in the results.

and the simulation run up to $t\sqrt{RT_o}/r = 0.00196$ using 1000 time steps. The benchmark results are represented as solid lines in Figure 10.7. Representations of the initial circular starting condition are shown in Figure 10.6(a). Figure 10.7 shows the normalised density for the 2D-TDEFM and 2D-EFM results for a mesh using 50 x 50 cells and 400 x 400 cells. The expected features of this flow are present in both results - an increase in Mach number, density and temperature occur through the radially expanding shock. The flow is smeared due to the inability of the solvers to accurately capture the flow on a coarse mesh, though this smearing diminishes as the mesh density increases. Since the flow is expected to display radial symmetry there is a single correct value for temperature, density and Mach number at any given radius. It can be seen that this is not true for the numerical solution - indeed, the degree of scatter in these profiles is an indication of the error of the solution and has been used as such previously [71].

Figure 10.8 shows the Mach Number and Density contours for TDEFM. Due to the radially symmetric nature of the problem, the resulting Mach number and density contours should also be radially symmetric. However, asymmetric features are present in both contours. These features include a slight necking in the contours of the Mach number at $M = 0.8$ and a single pressure 'bubble' in the pressure contours at $P/P_L = 4$. These two features are also present in the results obtained from EPSM when mass redistribution is employed. Figure 10.9 clearly shows the same necking of the Mach number contours, as well as a pressure bubble occurring at $P/P_L = 4$.

Figure 10.10 shows the Mach Number and density contours obtained using DTDEFM. These results can be seen to have a higher degree of radial symmetry, indicating a higher

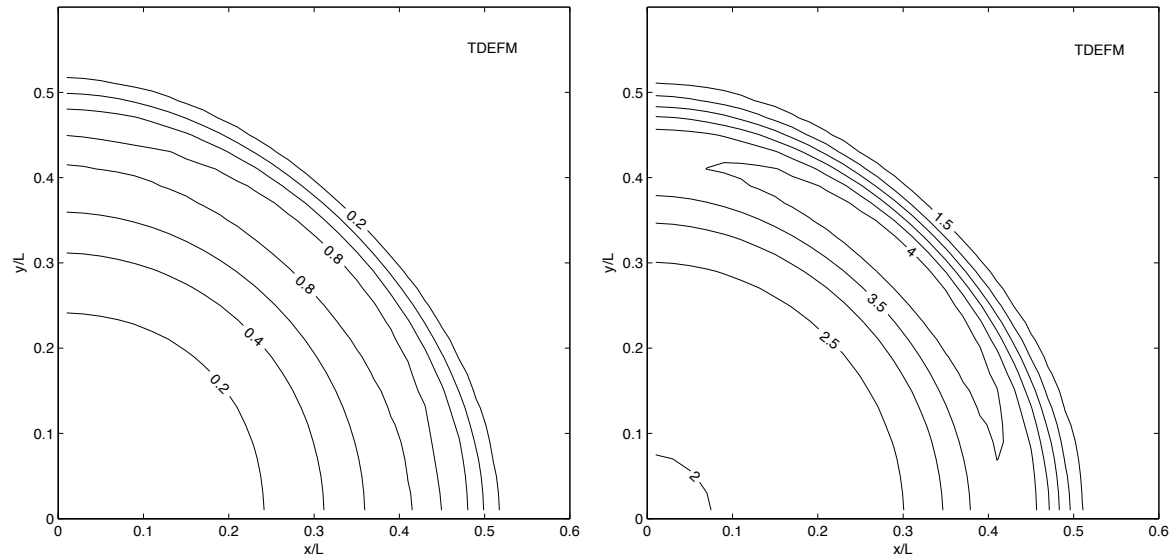

Figure 10.8: Contours of (Left) Mach number and (Right) normalised pressure from TDEFM at time $t\sqrt{RT_o}/r = 0.00196$. Contours of Mach number are of in steps of 0.2 up to 1.0, while normalised pressure contours are in steps of 0.5 up to 4.0.

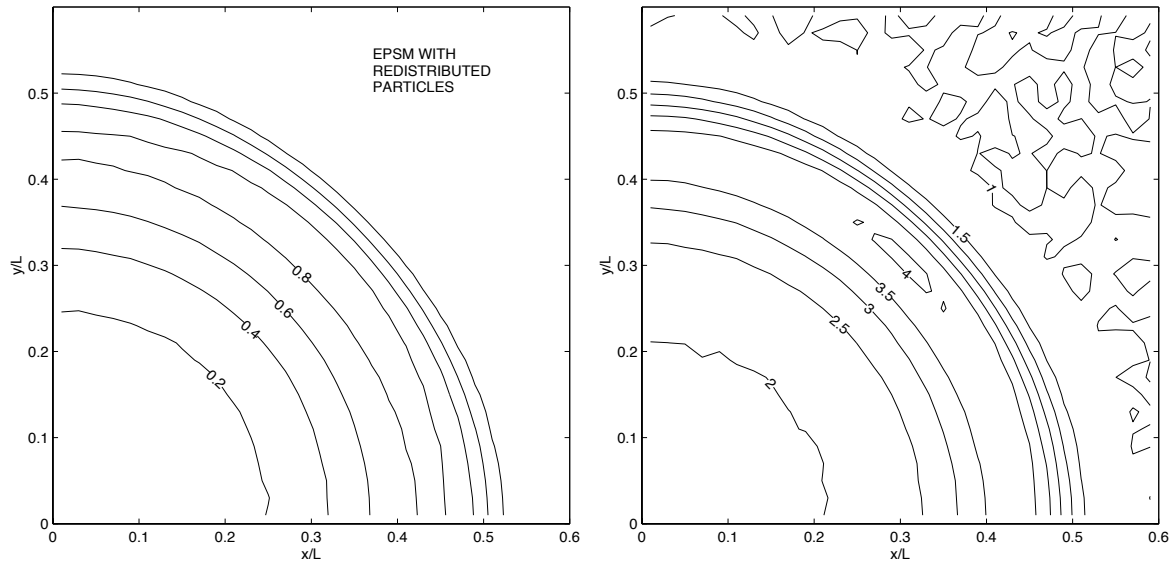

Figure 10.9: Contours of (Left) Mach number and (Right) normalised pressure from modified EPSM at time $t\sqrt{RT_o}/r = 0.00196$. Contours of Mach number are of in steps of 0.2 up to 1.0, while normalised pressure contours are in steps of 0.5 up to 4.0.

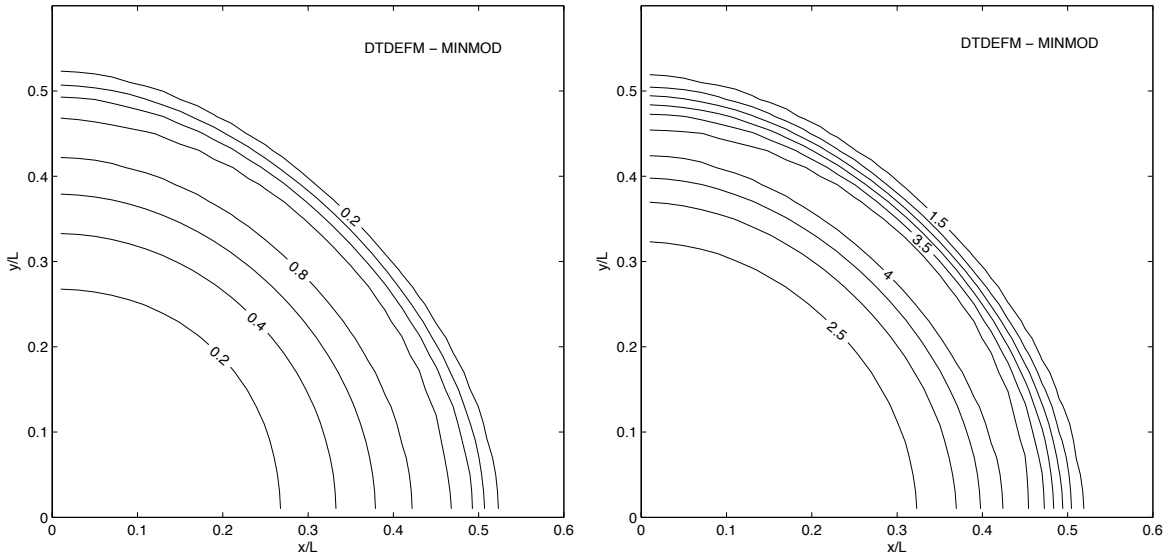

Figure 10.10: Contours of (Left) Mach number and (Right) normalised pressure for DTDEFM using MINMOD for gradient calculation at time $t\sqrt{RT_o}/r = 0.00196$. Contours of Mach number are of in steps of 0.2 up to 1.0, while normalised pressure contours are in steps of 0.5 up to 4.0.

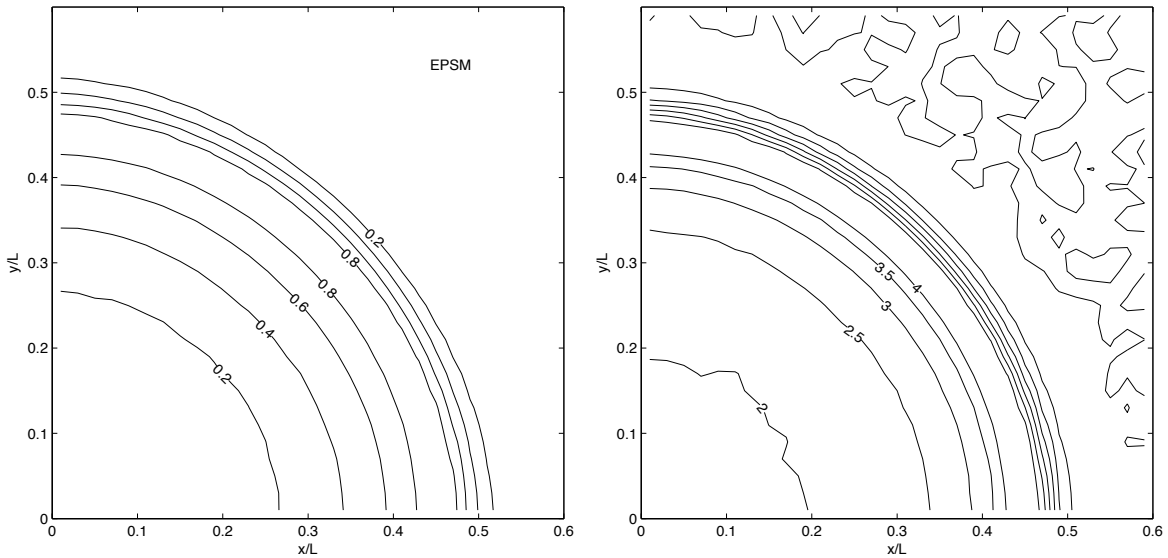

Figure 10.11: Contours of (Left) Mach number and (Right) normalised pressure from modified EPSM at time $t\sqrt{RT_o}/r = 0.00196$. Contours of Mach number are of in steps of 0.2 up to 1.0, while normalised pressure contours are in steps of 0.5 up to 4.0.

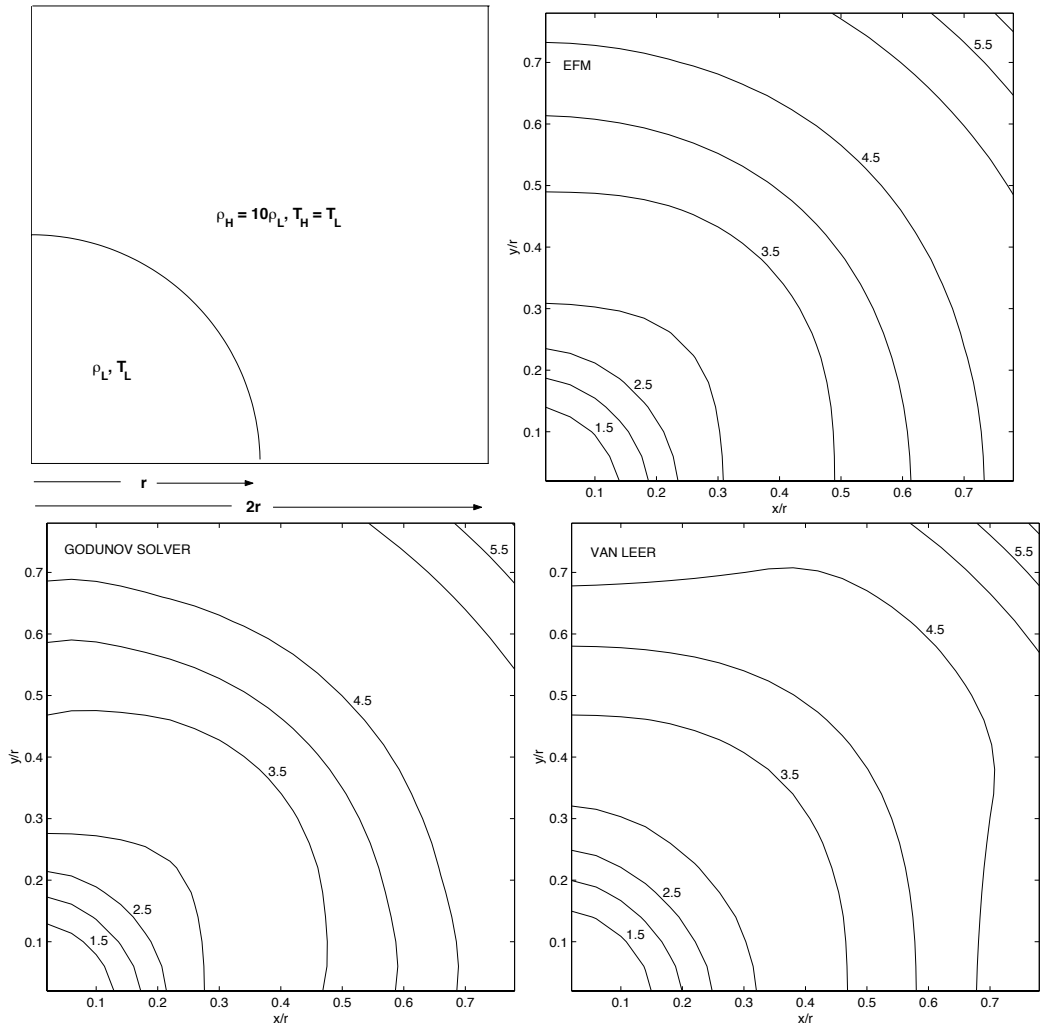

Figure 10.12: Direction decoupled 2D solutions to the implosion problem using a 50x50 mesh.(Top Left) Initial condition ; (Top Right) EFM; (Lower Left) Godunov Method [44]; (Lower Right) Van Leer [105]. Contours are of density (ρ/ρ_L) with contours every 0.5. Flow is shown at $t\sqrt{RT_L}/r = 0.098$ after 100 time steps. Initial conditions are $\gamma = 9/7$, $T_H/T_L = 1.0$, $\rho_H/\rho_L = 10$. Computational domain lies in the square region $0 < x/r < 2$.

degree of accuracy. There is no necking of the Mach number contours present, and the pressure 'bubble' has vanished, insteading expanding to a uniform, radially symmetric pressure band. Therefore, these effects cannot be due to direction decoupling or mesh resolution, but instead exist because of the forced mass redistribution across cells. These results are confirmed in Figure 10.11, showing Mach number and pressure contours obtained using ordinary EPSM.

10.3 2D Implosion Problem

TDEFM has been compared to EFM in a 2D implosion problem with the aim of demonstrating the problems associated with direction splitting. The implosion problem is shown in the

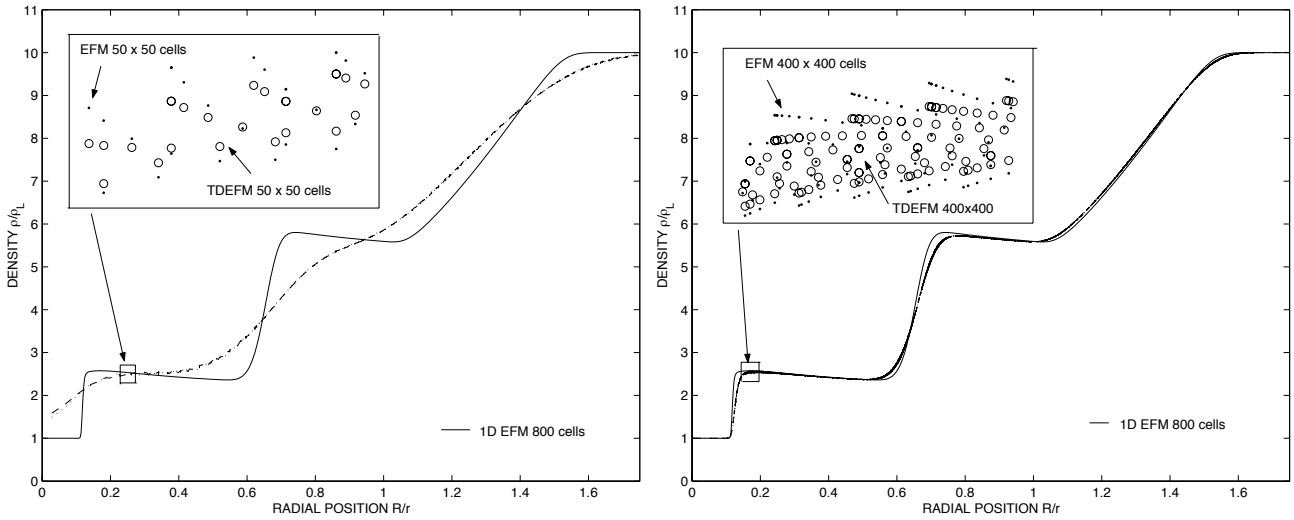

Figure 10.13: 2D solutions to the implosion problem showing normalised density using 2D-TDEFM ($\circ$) and 2D-EFM ($\cdot$) using a 50x50 mesh (left) and a 400x400 mesh (right). The solid line shows 1D results with 1x800 cells. Simulations are run up to $t\sqrt{RT}/r = 0.098$.

introduction in Figure 10.12. The initial conditions are as follows:

$$\begin{aligned} \rho &= \chi\rho_H \\ \chi &= f + (1-f)\frac{\rho_L}{\rho_H} \\ T_H/T_L &= 1 \\ U = V &= 0 \\ \gamma &= \tfrac{5}{3} \end{aligned} \tag{10.5}$$

where f is the fraction of the cell falling outside radius r. The results from 2D-TDEFM and 2D-EFM using a 50 x 50 and 400 x 400 mesh are shown in Figure 10.13. As expected, the fine mesh results more closely match the 1D results.

Shown in Figure 10.14 is a comparison of density contours between 2D-EFM and 2D-TDEFM for the same initial conditions used to obtain the results in Figure 10.12. The 2D-TDEFM contours are closer to being radially symmetric than the 2D-EFM contours, confirming the result obtained through the analysis of the angle of deviation. The time step used was small enough to justify the simplification of the primary TDEFM flux expressions in Equations 2-4 to the original EFM expressions. At this time step, the direction coupled EFM provided identical results (differences of less than 1e-13 percent) to the complete TDEFM expressions while performing the same number of exponential and error function evaluations as direction decoupled EFM.

Figure 10.15 shows the implosion results for DTDEFM and VTDEFM compared to TDEFM. There is significant variation in the methods as a result of the inclusion of the density and velocity gradients.

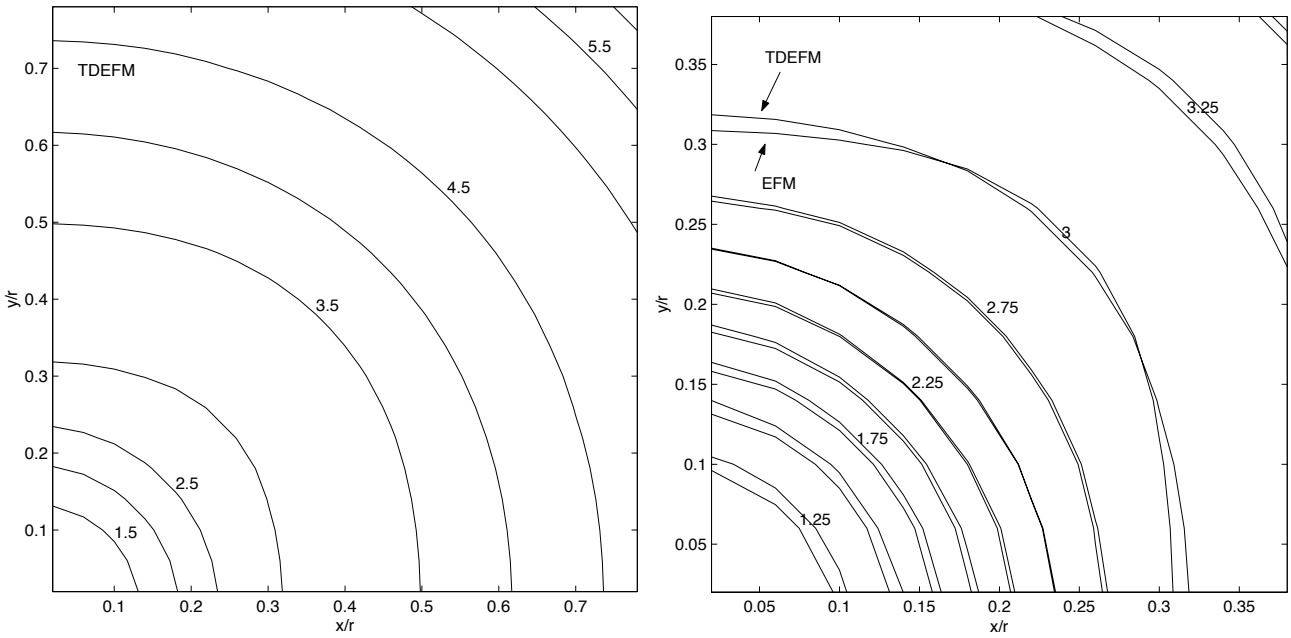

Figure 10.14: (Left) Contours of density for the implosion problem shown in Figure 10.12 using 2D-TDEFM, (Right) Enlarged (2x) comparison between contours of density from TDEFM and EFM using the same initial conditions showing the effects of direction decoupling.

10.4 Hypersonic flow over a rectangular body

The previous examples dealt with predominately low speed, unsteady flows in a square region. The results for steady hypersonic flow over the rectangular body shown in Figure 10.16 are shown here. The flow conditions are $M_\infty = 20$, $\rho_\infty = 1$ and $T_\infty = 1$. The flow is progressed until $t\sqrt{RT_\infty}/H = 3$. The gas is ideal with $\gamma = 7/5$. Density contours of the result obtained using TDEFM is shown in Figure 10.17. The top and right hand side boundaries are extrapolated outflow. The lower boundary and the body surfaces are reflective boundaries which are appropriate for this inviscid calculation. As expected, a detached bow shock has formed, with the density increasing through the bow shock and decreasing as the flow expands around the corner of the rectangular body. There are no bumps or other spurious oscillations present in the bow shock.

The computational region extends from $-2.5 \geq x/H \geq 1.5$. and $0.0 \geq y/H \geq 4.0$ with varying computational grid densities. The rectangular body has height H, is placed at $x/H = 0.0$ and extends to $x/H = 1.5$. The initial conditions throughout the flow field are of uniform Mach Number M, varying from $M = 5$ to $M = 20$. The gas is an ideal gas with $\gamma = 7/5$. No viscous effects are included in the calculations, although a numerical viscosity is present in all methods. All reflections off the body are considered specular. The flow is symmetric around $y/H = 0.0$.

The temperature and density profiles alone line A-A' (shown in Figure 10.16) are shown in Figure 10.18. Here, we can see that even for steady flow problems there is a distinct difference in the solutions. The location at which the detached bow shock crosses the line A-A' differs

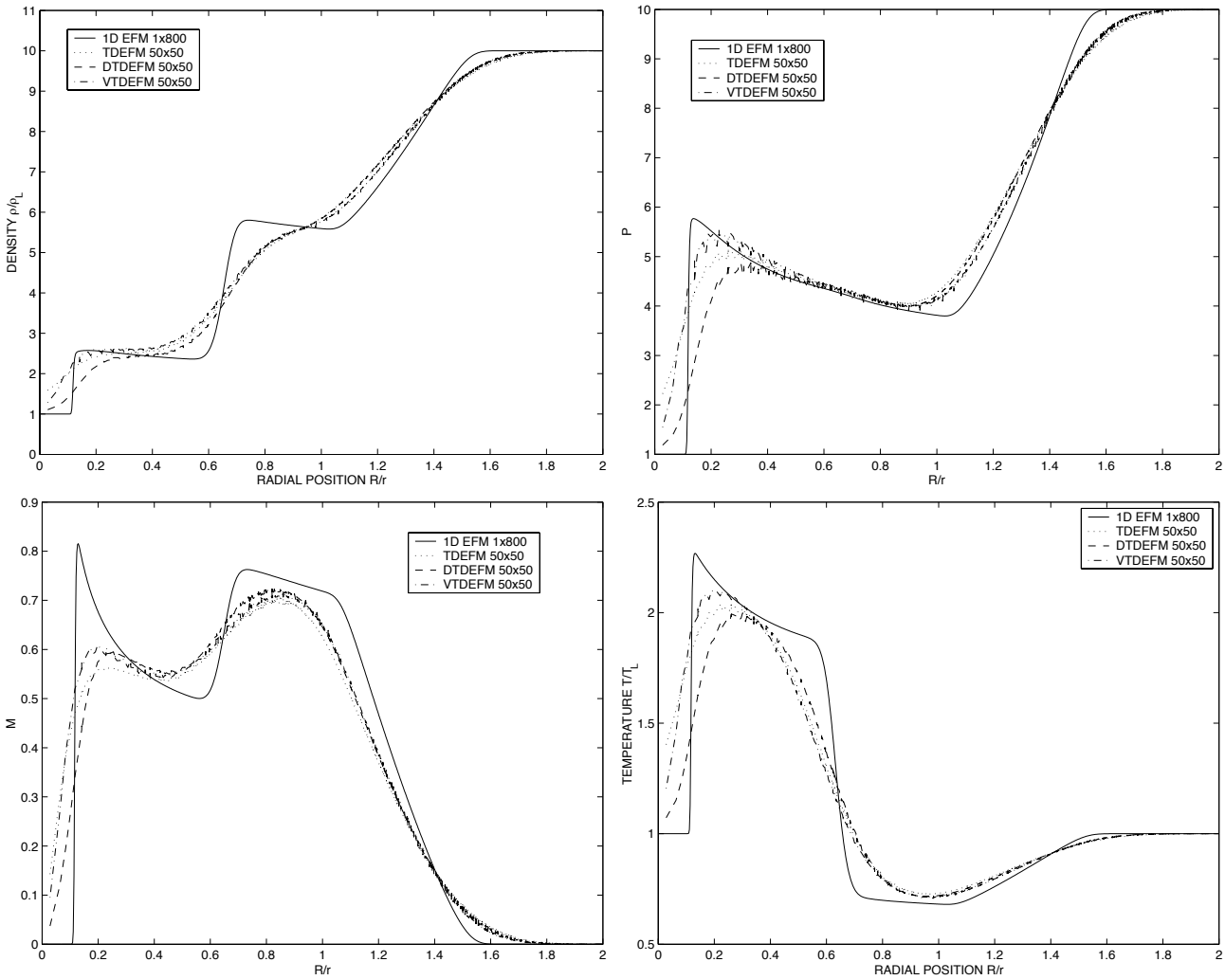

Figure 10.15: 2D solutions to the implosion problem showing normalised density using 2D-TDEFM, 2D-DTDEFM and 2D-VTDEFM using a 50x50 mesh (left). The solid line shows 1D results with 1x800 cells. Simulations are run up to $t\sqrt{RT}/r = 0.098$.

for true direction and direction decoupled fluxes. This is true regardless of mesh density. As shown by the density profile in Figure 10.18, when the number of cells is increased by more than 400 percent there is still a noticeable difference in the location of the bow shock. The effect of direction decoupling here is quite severe as the flow is not aligned with the grid. The temperature profile in Figure 10.18 extends from $1 \geq y/H \geq 2.5$ (along the line A-A' shown in Figure 10.16) where the flow is closer to the body and better aligned with the computational grid. The results demonstrate that the difference between the methods decreases where the flow is better aligned with the grid. As the flow direction diverges from grid alignment, i.e. as the distance y/H increases along line A-A', the difference between the results is shown to increase.

As may be expected, the shock stand off distance is also affected. Presented in Table 10.1 are the shock standoff distances using TDEFM and EFM with varying mesh densities. The

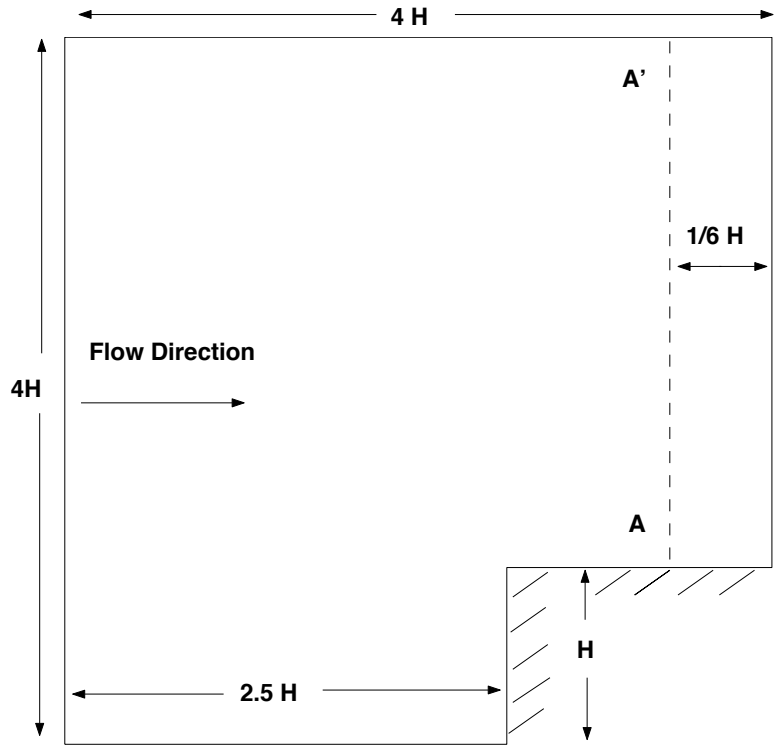

Figure 10.16: The computational domain used for the hypersonic flow example over a rectangular body of height H. Flow is at Mach 20 with $\gamma = 1.4$. Initial conditions are $\rho = \rho_\infty, M = M_\infty$ and $T = T_\infty$. The simulations are progressed in time to $t\sqrt{RT_\infty}/H = 3$.

Method	Number of cells	Standoff Distance Δ/H	Relative Shock Standoff Distance
TDEFM	3255	1.118	1
EFM	3255	1.185	1.06
EFM	3596	1.163	1.04
TDEFM	13050	1.0195	1
EFM	13050	1.05	1.03

Table 10.1: Shock standoff distances for varying computational grids.

shock standoff distance is defined here as the location along $y = 0$ where the Mach number equals unity. Since TDEFM (in its simplified form) is typically 10 percent computationally slower than EFM, tests were performed using EFM with a correspondingly larger number of cells. While the results improve slightly, the difference between the results is still significant. In terms of the shock standoff distance, increasing the number of cells from 3522 to 3596 decreased the difference in normalised shock standoff distance from 6 percent to 4 percent. Similar trends were shown with increasing mesh densities. Therefore, we conclude that the benefits of direction coupling outweigh the slight increase in computational expense.

Figure 10.19 shows normalised temperature contours of Mach 20 flow over the rectangular body specified including the results from DTDEFM and EPSM. The EPSM and DTDEFM results are more closely matched than the TDEFM results. All of the features expected are

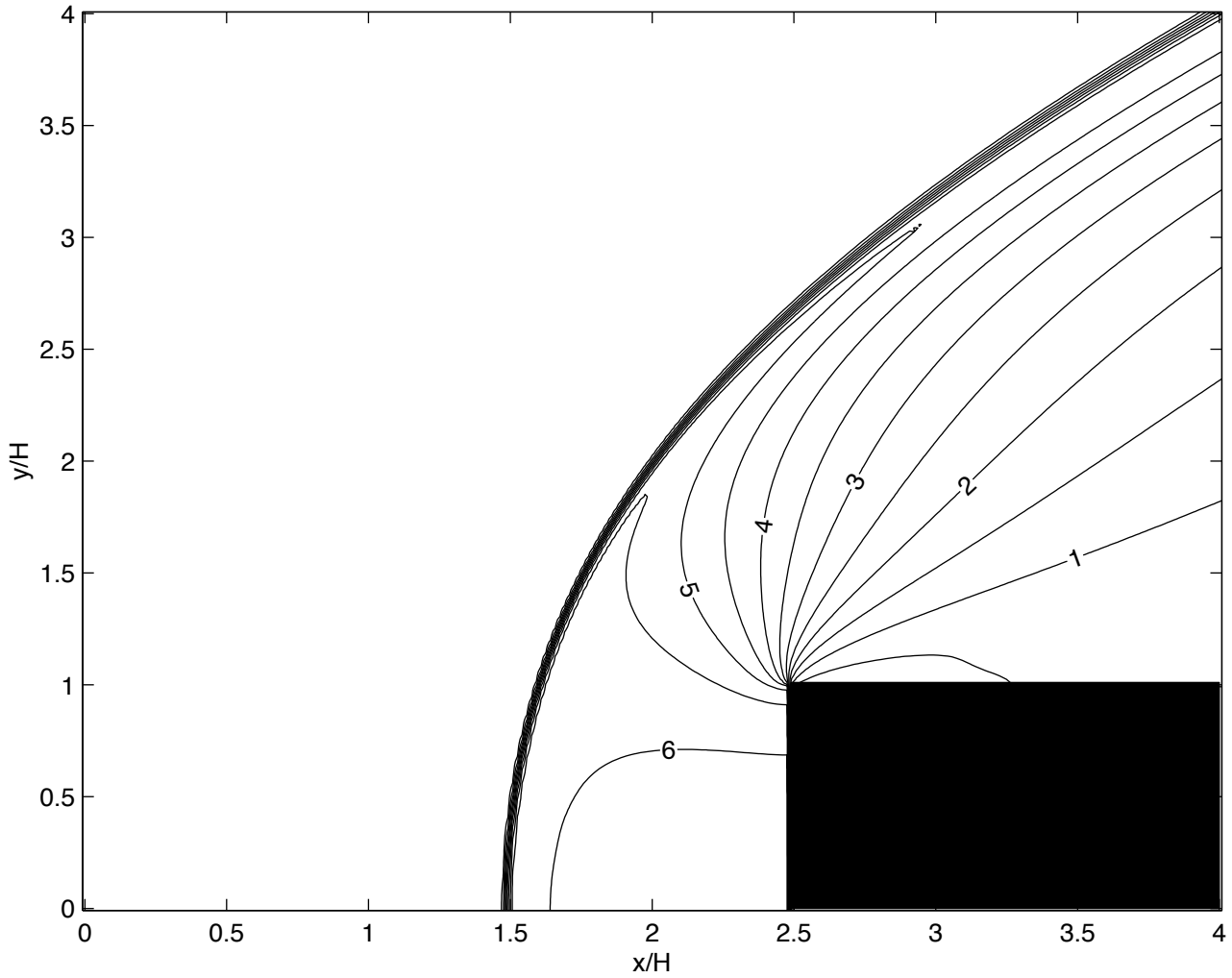

Figure 10.17: Colour contours of density for hypersonic flow over a rectangular body of height H using TDEFM. The computational region extends to $4H$, with the front of the body located at $[2.5H, 0]$. Flow is at Mach 20 with $\gamma = 1.4$. Initial conditions are $\rho = \rho_\infty, M = M_\infty$ and $T = T_\infty$. The simulations are progressed in time to $t\sqrt{RT_\infty}/H = 3$.

present - the flow is compressed and heated through the bow shock and then cools as the flow expands around the body. Figure 10.20 shows the vertical temperature profiles at $x/H = 0.85$ of Mach 20 flow over the rectangular body. Results taken from EFM and a Godunov Solver [44] are also presented. In the temperature profile near the top surface, the DTDEFM solution almost exactly matches the EPSM solution, while the TDEFM, EFM and Godunov solutions all vary. In the density profile through the bow shock, the DTDEFM solution best attempts to capture the shock when compared to the other continuum methods. This result is also shown in Figure 13.1, with DTDEFM consistently capturing shocks at thicknesses closer to those obtained by EPSM. Due to the nature of the Godunov solver, and the fact that viscosity has been disregarded in these simulations, it is natural that the Godunov solver posses the thinnest shock thickness. Figure 13.1 also shows shock standoff distance, with EPSM, DTDEFM and TDEFM show similar shock placements varying no more than 3%.

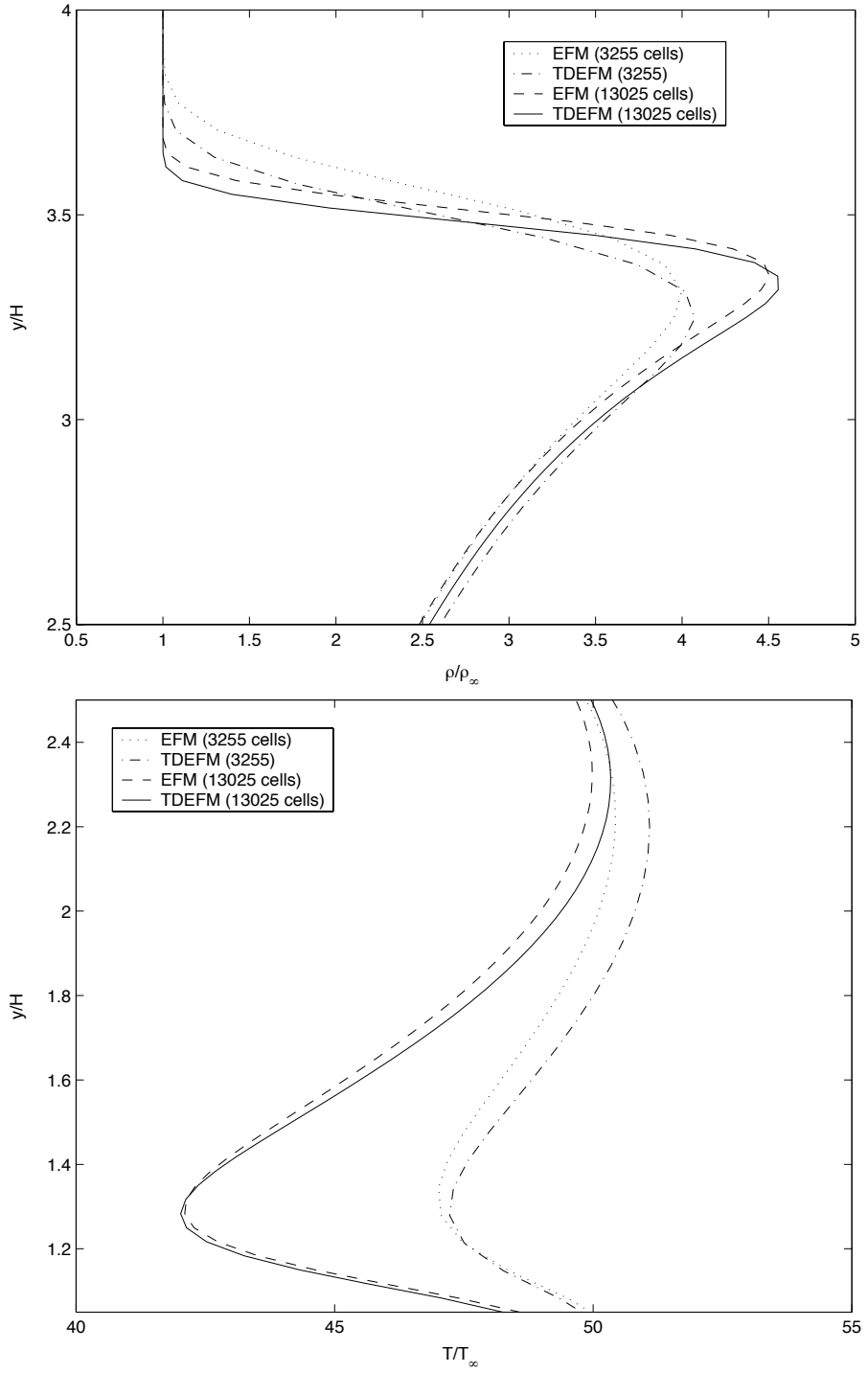

Figure 10.18: Density profiles (top) and temperature profiles (bottom) from EFM and TDEFM solutions using various mesh densities. The results shown are alone line A-A' as shown in Figure 10.16. The temperature profile is in the region ($1 \geq y/H \geq 2.5$). The density profile is in the region ($2.5 \geq y/H \geq 4$). Flow is at Mach 20 with $\gamma = 1.4$. Initial conditions are $\rho = \rho_\infty, M = M_\infty$ and $T = T_\infty$. The simulations are progressed in time to $t\sqrt{RT_\infty}/H = 3$.

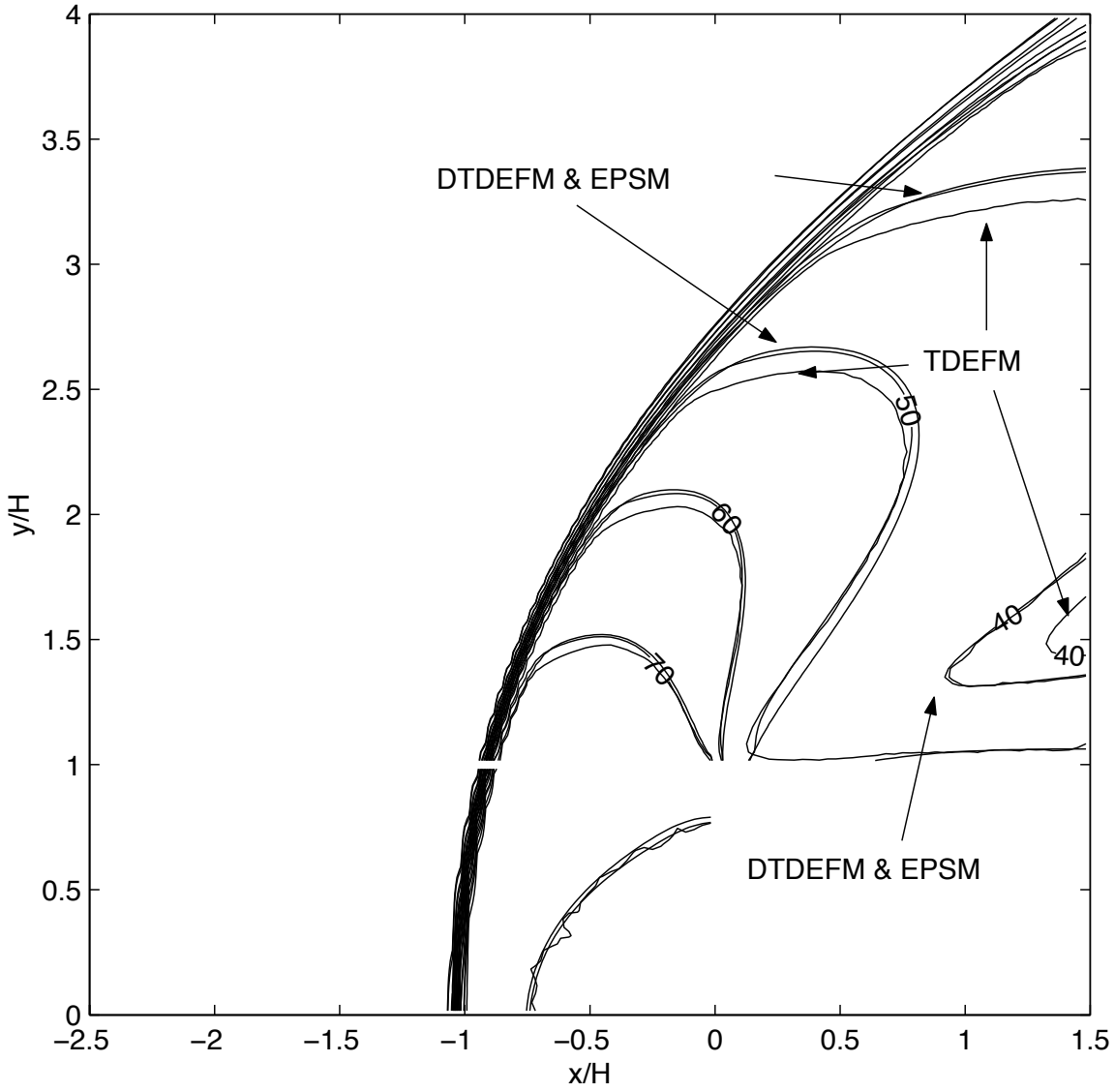

Figure 10.19: Contours of steady state temperature showing results from EPSM, Density TDEFM (DTDEFM) and TDEFM. $M_\infty = 20$, $\gamma = 7/5$. $\Delta x = \Delta y = H/30$. Contours are taken every $T/T_\infty = 10$. DTDEFM resulted obtained using MINMOD to calculate density gradients in the source cells.

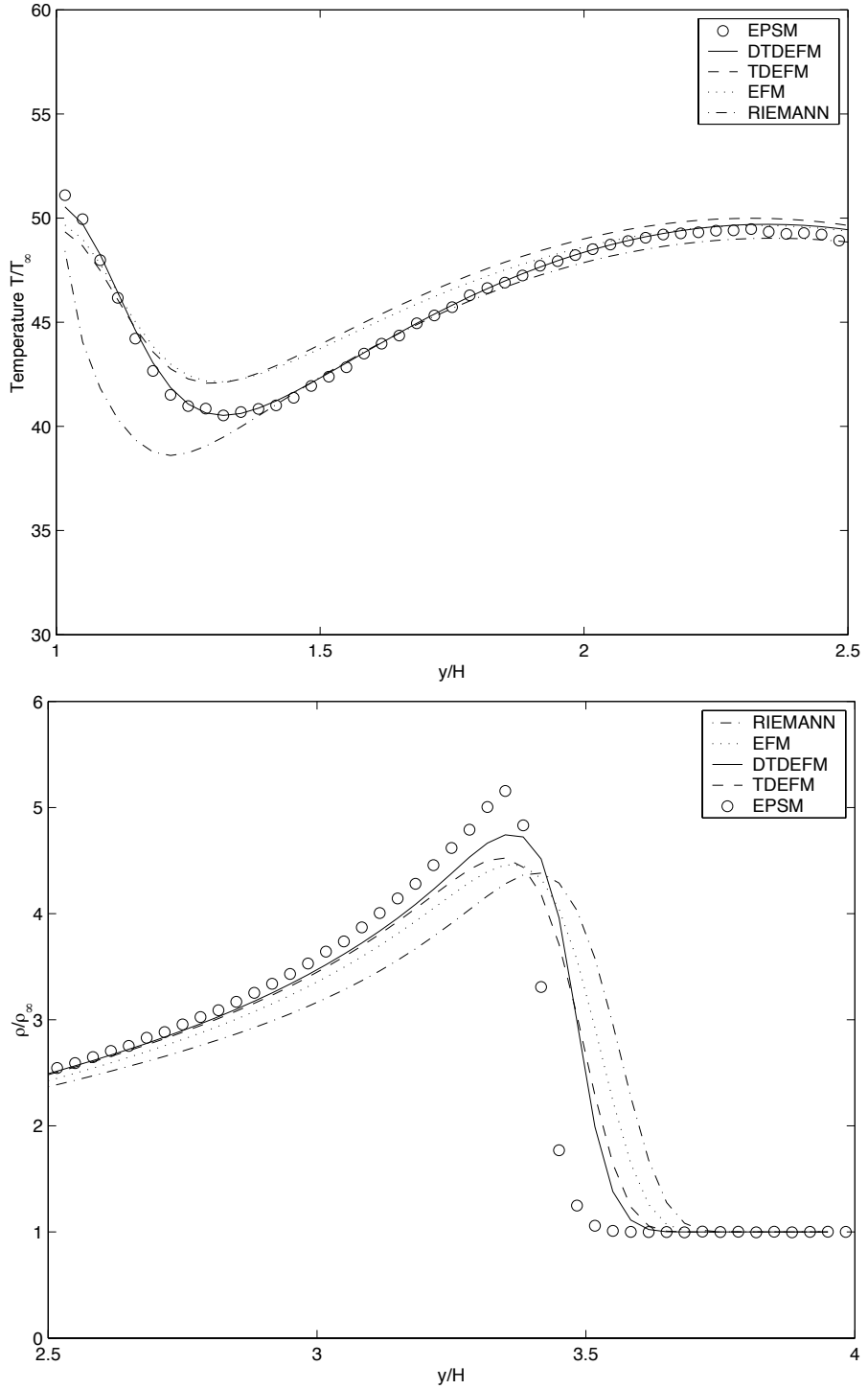

Figure 10.20: Temperature and Density profiles at $x/H = 0.85$ showing results from EPSM, DTDEFM and TDEFM, EFM and a Godunov Solver. (Top) Temperature Profiles extending from $1.0 \geq y/H \geq 2.5$. (Bottom) Density Profiles extending from $2.5 \geq y/H \geq 4.0$. $M_\infty = 20$, $\gamma = 7/5$. $\Delta x = \Delta y = H/30$. DTDEFM resulted obtained using MINMOD to calculate density gradients in the source cells.

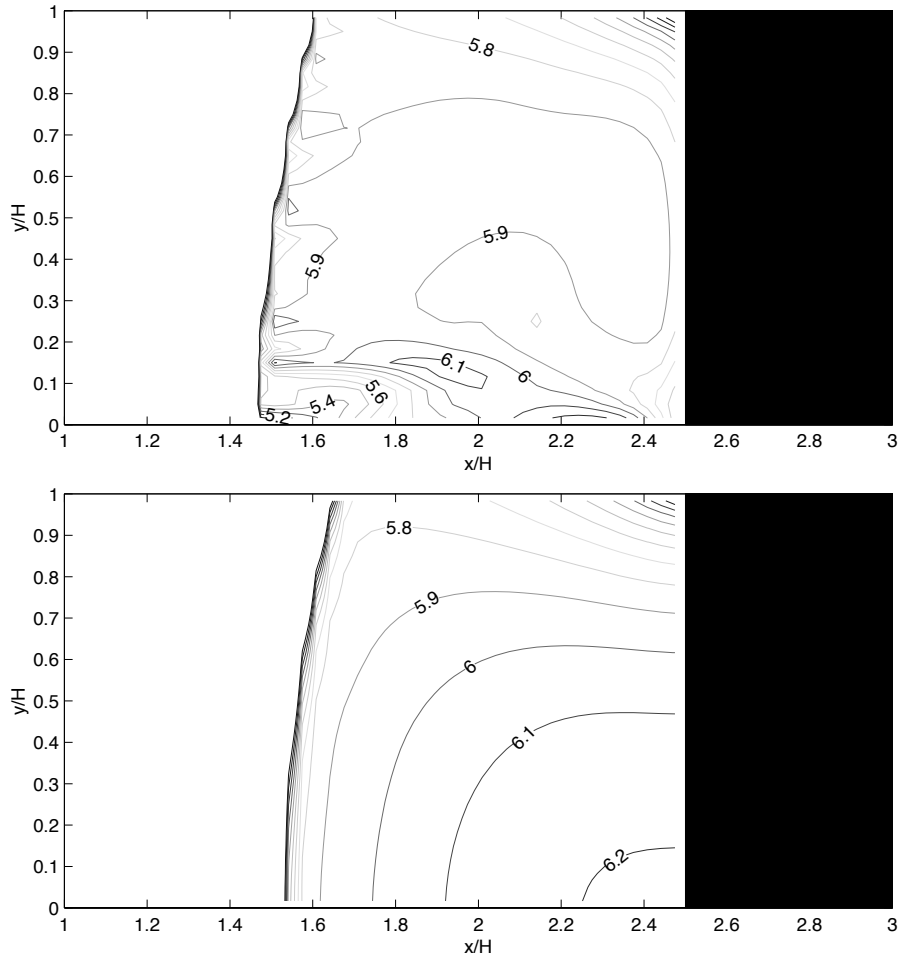

Figure 10.21: Comparison of density contours in stagnation region for hypersonic flow over a rectangular body. [Top] Riemann solver, [Bottom] TDEFM. Flow is at Mach 20 with $\gamma = 1.4$. Initial conditions are $\rho = \rho_\infty, M = M_\infty$ and $T = T_\infty$. The simulations are progressed in time to $t\sqrt{RT_\infty}/H = 3$.

Many of the symptoms associated with the Carbuncle Phenomenon (shown in Figure 4.3) can been seen in hypersonic flow over a rectangular body. This is demonstrated in Figure 10.21. Results taken from the approximate Riemann solver are seen to have spurious oscillations in the stagnation region similar to several of the solvers shown in Figure 4.3. These oscillations are not present in the TDEFM solution. However, it should be noted that the numerical diffusion present in TDEFM is equivalent to that of direction decoupled EFM, which is also seen to be immune to the Carbuncle phenomenon. Therefore it is more likely that TDEFM is immune because of its inherent numerical diffusion rather than its direction coupled fluxes.

10.5 2D Hypersonic Flow over a cylinder

Results for hypersonic flow of an ideal inviscid gas over a cylinder are presented for TDEFM. The computational grid is a regular rectangular cartesian grid. The surface of the cylinder is mapped onto the mesh using the approach discussed in Section 8.3.4. The gas is ideal with

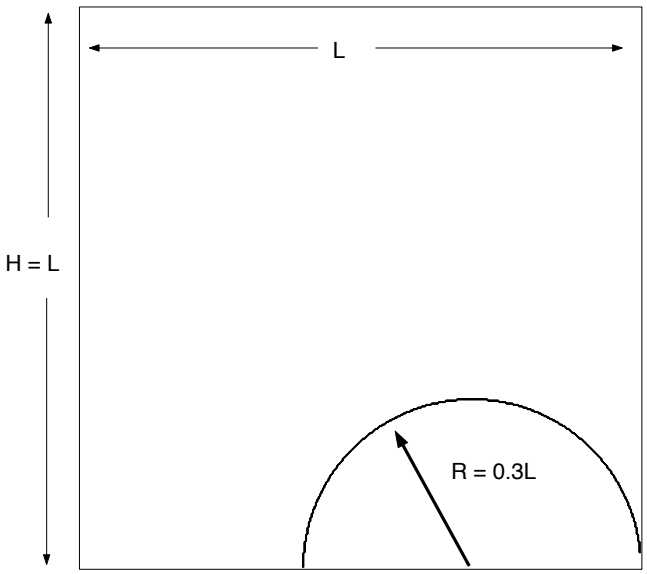

Figure 10.22: Computational domain for simulation of hypersonic flow over a cylinder. The free stream Mach number is $M_\infty = 3$. The gas is ideal with $\gamma = 7/5$.

$\gamma = 7/5$ and a freestream Mach number of $M_\infty = 3$. The surface of the cylinder is specularly reflective. The geometry used is shown in Figure 10.22. The lower boundary is a symmetry condition and the right hand side and upper boundaries are interpolated outflow conditions. The computational grid is 400 x 400 cells with the cylinder occupying 22622 cells. All regions of the flow initially have a Mach number $M = M_\infty$.

Figure 10.23 shows the unsteady flow development of density contours for the conditions specified. No spurious oscillations are present in the stagnation region. There are no obvious flaws in the density contours resulting from the use of domain mapping or an unaligned computational grid. The calculated TDEFM fluxes are spatially and temporally first order. The results are typical of a conventional, flow alligned finite volume CFD solver.

10.6 2D Hypersonic Flow over a forward facing step

Results for hypersonic flow over a forward facing step are presented for various solvers. This problem, made famous by Woodward and Collela [112] involves the unsteady development of shocked flow over a rectangular step. The initial conditions for this problem are presented in Figure 10.24. The gas is ideal with $\gamma = 5/3$. All boundaries are specularly reflective with exception made to the inflow and outflow boundaries. The inflow is fixed at the desired inflow conditions while the outflow is managed using extrapolated conditions. The procedure for this is identical to that used in the previous hypersonic flow problem.

Figure 10.25 displays density contours taken from the TDEFM solution after 1600 time steps of size $\Delta t = 0.0005$. The largest local kinetic CFL present in the conditions presented is 0.47. The results from TDEFM suitably match those obtained from existing continuum solvers.

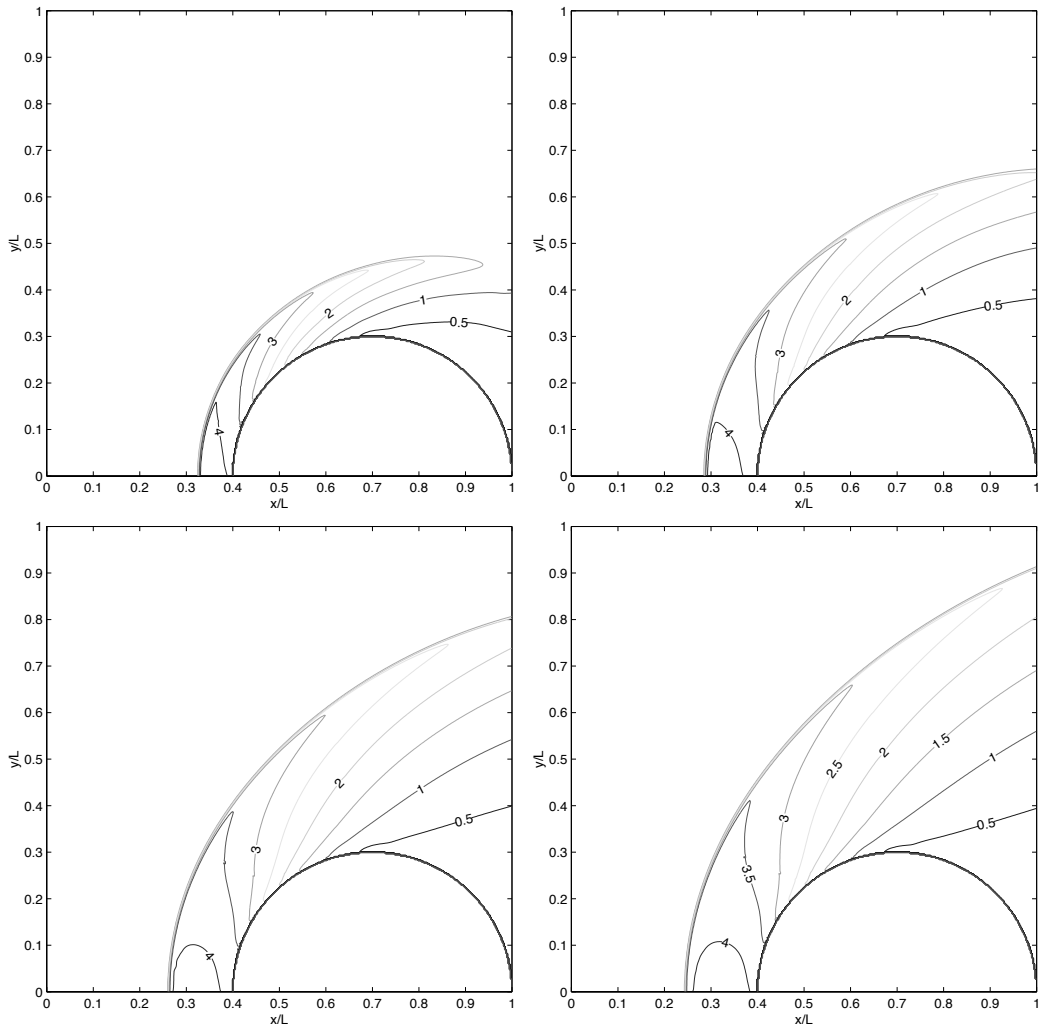

Figure 10.23: Unsteady development of hypersonic flow over a cylinder. Results are shown at $t\sqrt{RT_\infty}/L = 0.1, 0.2, 0.3 and 0.5$ The free stream Mach number is $M_\infty = 3$. The gas is ideal with $\gamma = 7/5$. The computational grid is comprised of 400 by 400 cells.

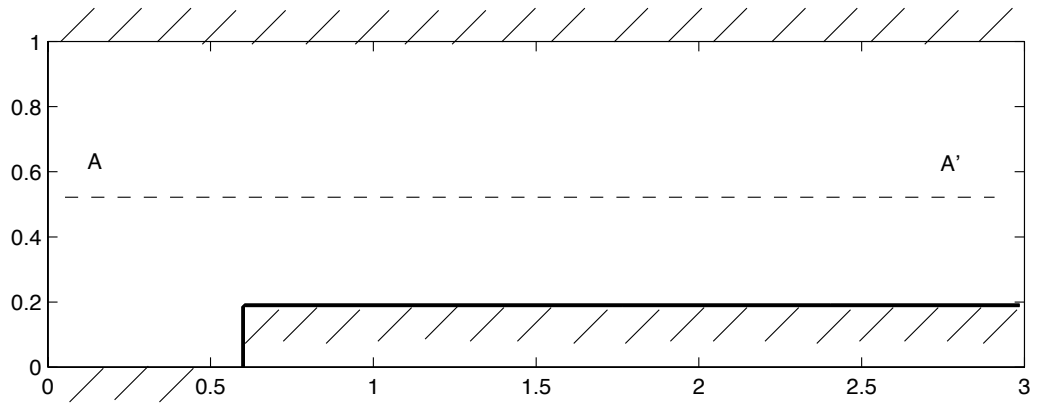

Figure 10.24: Geometry used for 2D hypersonic flow over a forward facing step. The gas simulated is ideal with $\gamma = 5/3$ with a free stream Mach number $M_\infty = 3$.

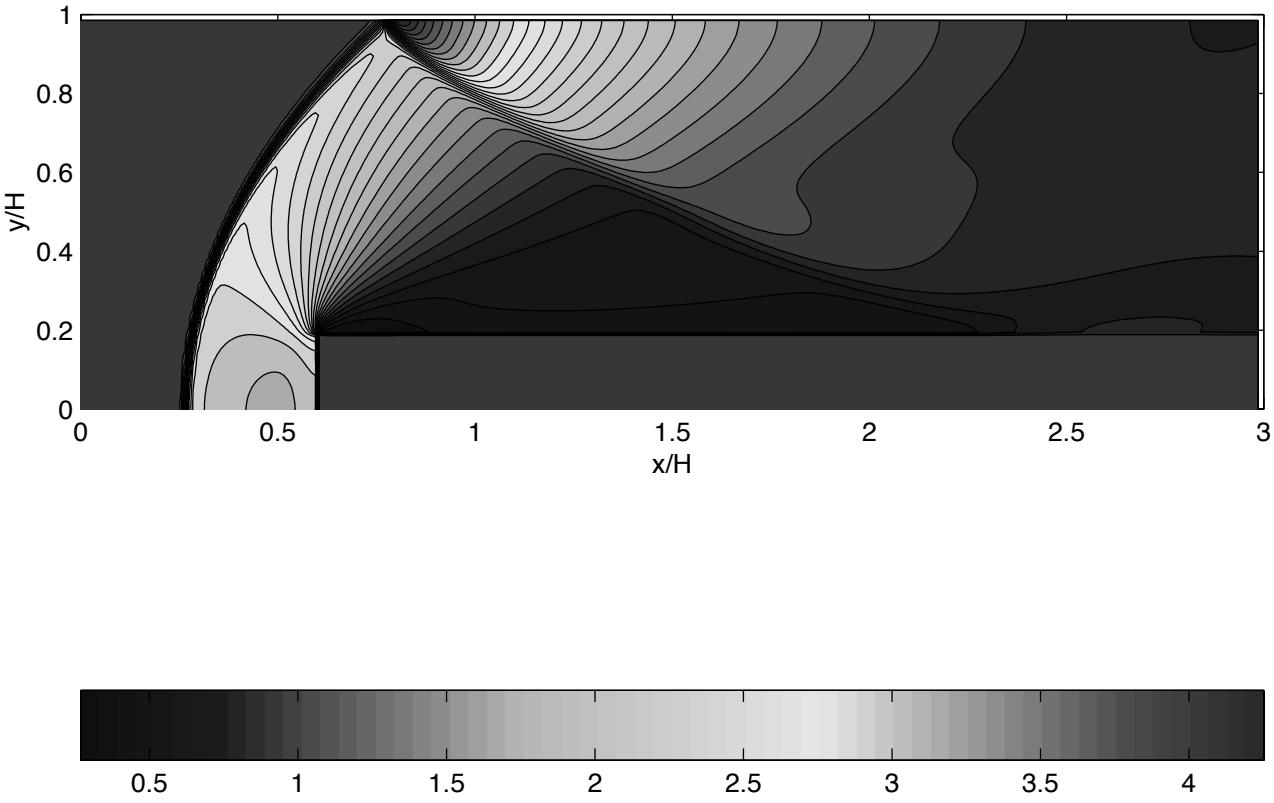

Figure 10.25: Density contours taken from the TDEFM solution after 1600 time steps of size 0.0005. The free stream Mach number is $M_\infty = 3$. The gas is ideal with $\gamma = 5/3$.

The leading bow shock is free of oscillations or wriggles.

Temperature contours for various solvers after 1600 time steps are shown in Figure 10.26. The differences between the EFM results and TDEFM results are not as significant as encountered in the previous hypersonic flow problem. Due to the encapsulating nature of the upper boundary, flow is not as free to be unaligned with the computational mesh as when an extrapolative boundary is used. Since the flow is better aligned with the mesh and the local kinetic CFL numbers are quite low, the results should be very similar. However, greater differences exist between the higher order VTDEFM and DTDEFM.

10.7 2D Planar Shock propagation over square cavity

The square cavity problem investigated by Reichenbach et. al [83] and used by Long to compare against DSMC simulations [60] is shown in Figure 7.2 and is used here for comparison to TDEFM and DTDEFM results. Presented are simulations using wave Mach numbers of 2.0. All simulations used 400x200 cells are compared qualitatively to Reichenbach's results. The gas used in the simulations is inviscid with $\gamma = 7/5$. No heat transfer or viscous effects are considered. The initial shock position was located at $x/L = 0.1125$ and 1492 time steps were taken to reach a total time of $149\mu s$. Contours of density taken from DTDEFM, TDEFM and

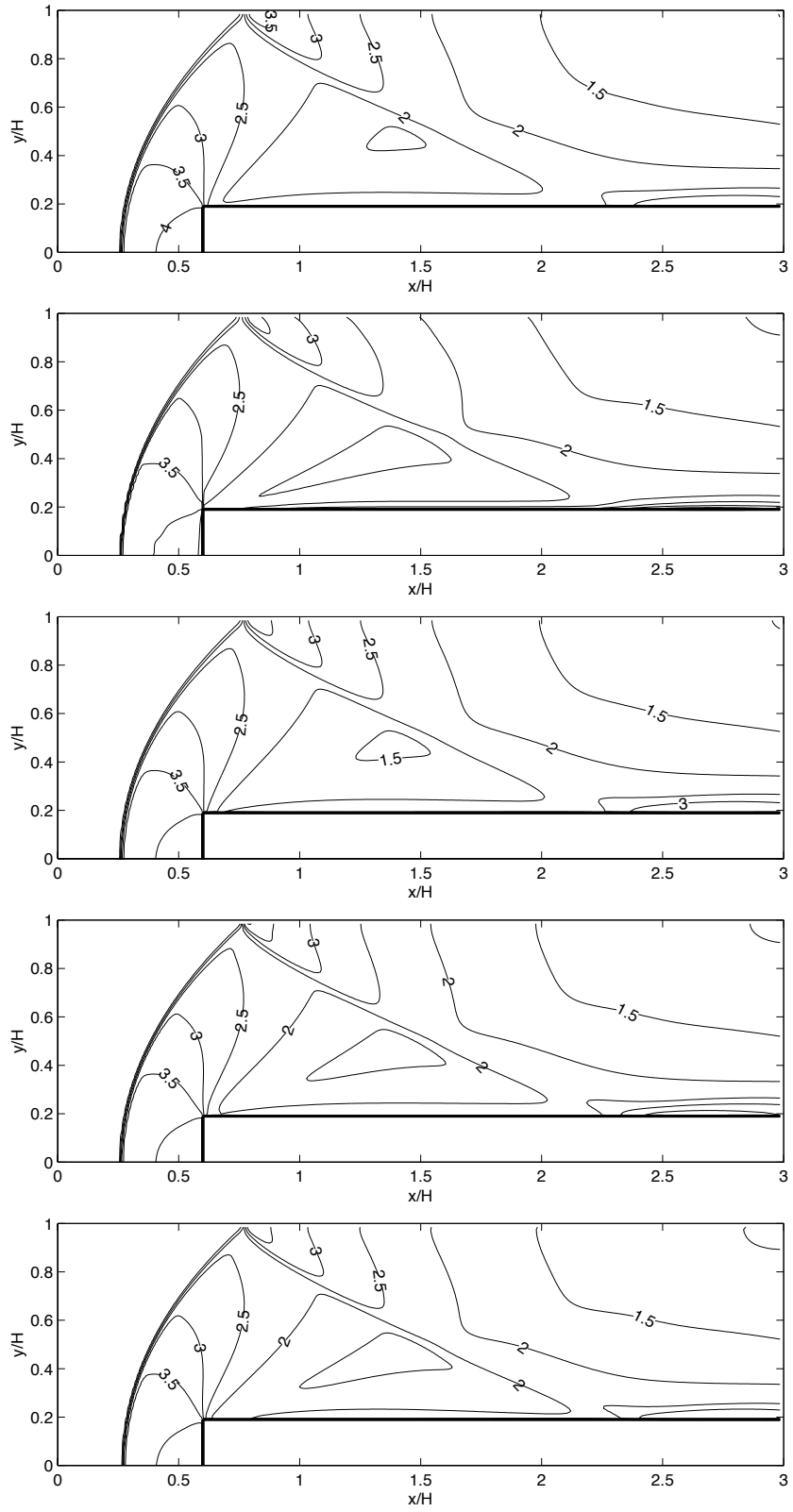

Figure 10.26: Temperature contours taken from various solvers after 1600 time steps of $\Delta t = 0.0005$. The gas is ideal with $\gamma = 5/3$. The solver results displayed are EFM (Top), a Riemann solver, TDEFM, DTDEFM and VTDEFM (Bottom). The grid resolution was 100x300 cells. The free stream flow Mach number is $M = 3$. The gas is inviscid and all boundaries are specularly reflective with exception to the inflow and outflow boundaries.

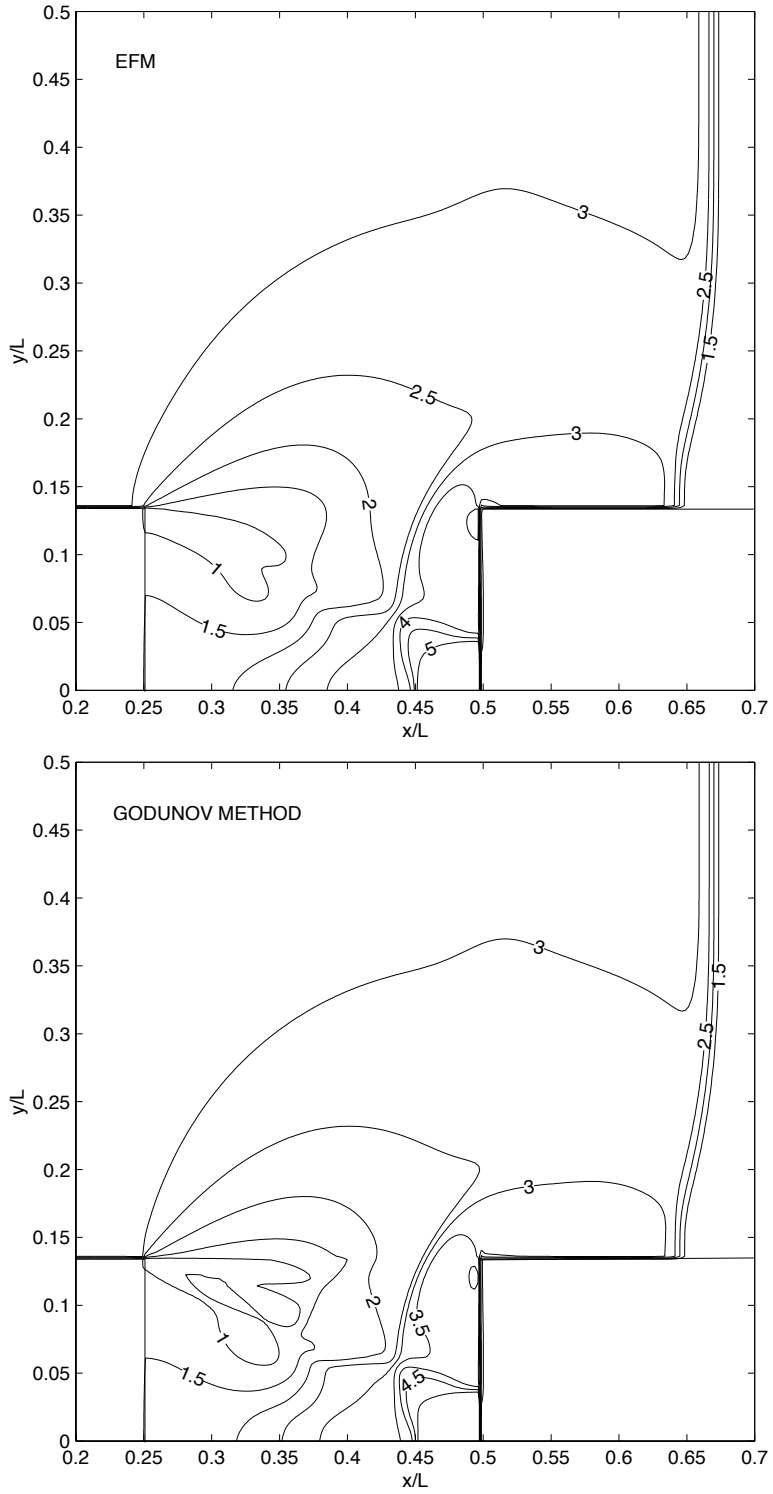

Figure 10.27: Comparison of planar shock interaction results showing EFM (left) and a Godunov solver (right) . Presented are contours of density in steps of 0.5 kgm^{-3}. The density ratio across the shock wave is 2.6667, temperature ratio = 1.687. The density in front of the shock is $1.14kgm^{-3}$. 1494 time steps are taken up to a total time of 149 μs.

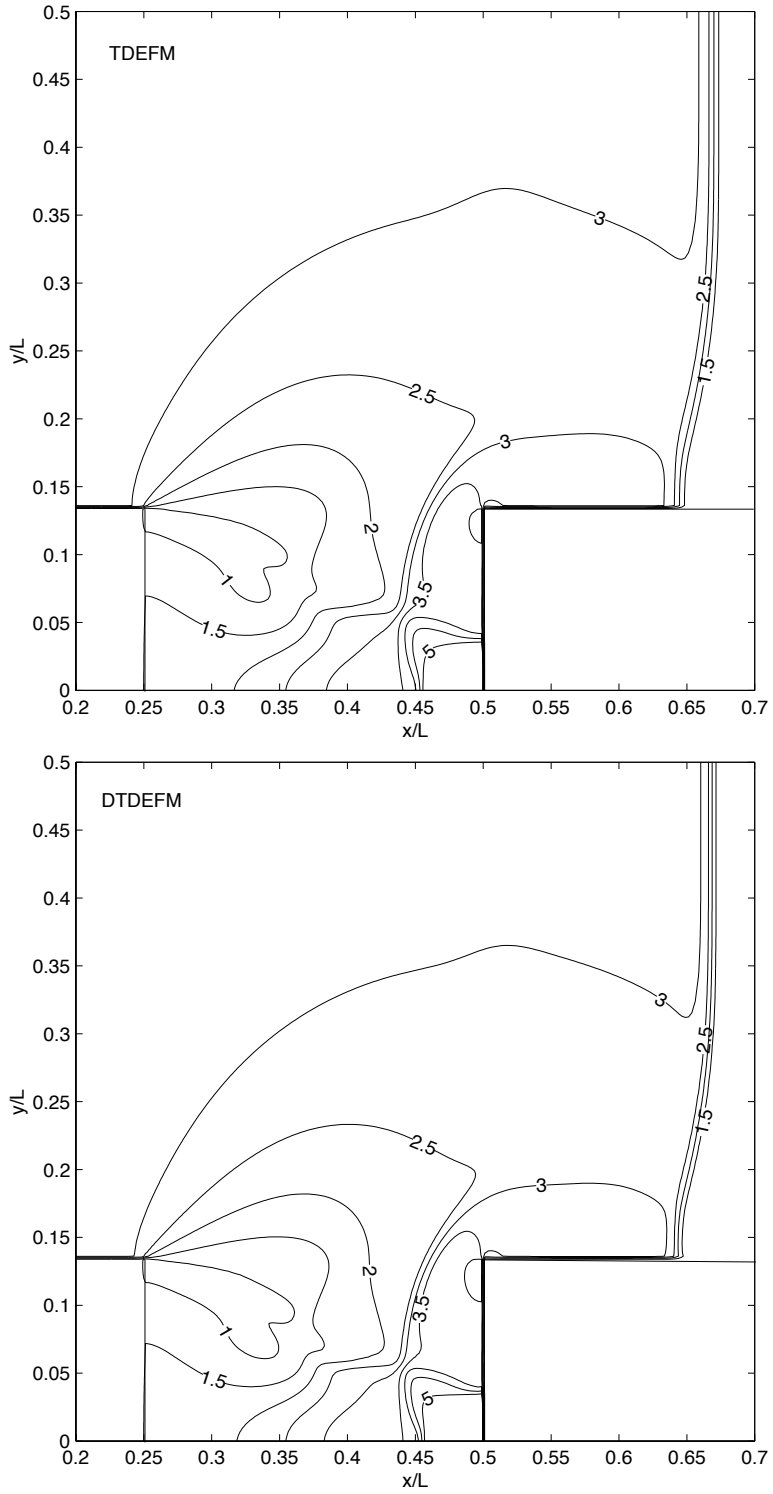

Figure 10.28: Comparison of planar shock interaction results showing TDEFM (left) and DT-DEFM (right) . Presented are contours of density in steps of 0.5 kgm^{-3}. The density ratio across the shock wave is 2.6667, temperature ratio = 1.687. The density in front of the shock is $1.14kgm^{-3}$. 1494 time steps are taken up to a total time of 149 μs.

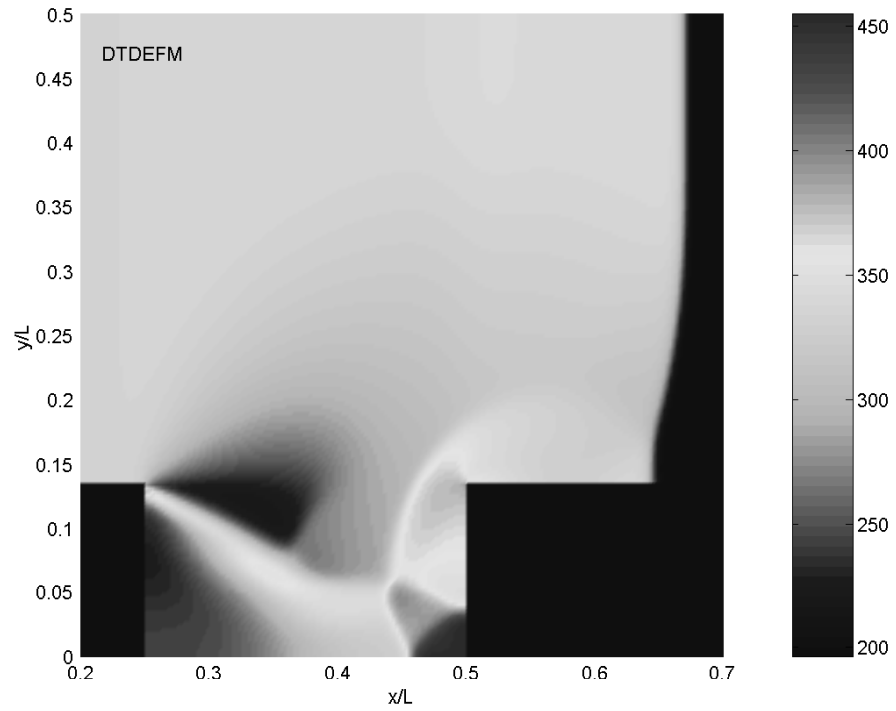

Figure 10.29: Planar shock interaction results from DTDEFM showing temperature (K). The density ratio across the shock wave is 2.6667, temperature ratio = 1.687. The density in front of the shock is $1.14 kgm^{-3}$, the temperature $196K$. 1494 time steps are taken up to a total time of 149 μs.

the Godunov solver results are presented in Figure 10.27 and Figure 10.28. A surface plot of temperature taken from DTDEFM results is shown in Figure 10.29. All of the features seen in the experimental results [83] are present.

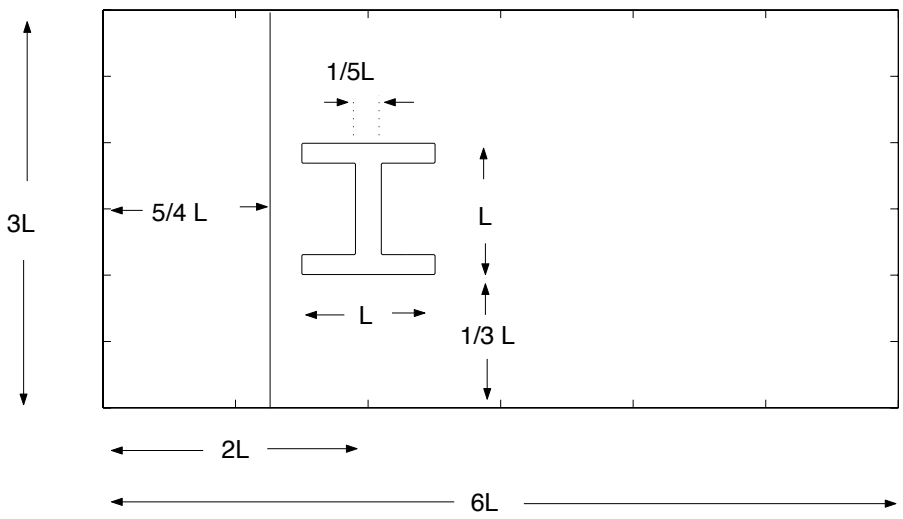

Figure 10.30: I Beam geometry used for blast wave and shock wave tests.

10.8 2D Blast waves around buildings and structures

2D Shock wave interaction with I Beam structure

Results are presented for two dimensional flow around the I-beam displayed in Figure 10.30. The geometry and conditions used here are similar to the numerical experiments performed by Long and Sharma [60]. A region of shocked gas is traveling toward the I-beam. The approaching shock creates a pressure increase of $p_H/p_L = 4.5$ and a temperature increase of $T_H/T_L = 1.687$ where subscripts H and L represent high and low density conditions respectively. The initial location of the shock wave is $(1/4)L$ away from the leading edges where L is the length and width of the I beam. The I beam itself is square with sections of thickness $(1/5)L$. The gas used in the simulation is assumed inviscid with $\gamma = 1.4$. The computational region is divided into 400 x 200 cells.

The transient development of flow calculated using TDEFM is shown in Figure 10.31 and Figure 10.32. The same features demonstrated by Long and Sharma [60] are present in the results provided by TDEFM. An important feature to note is that the conditions behind the propagating shock wave are constant, resulting in a stationary detached bow shock positioned in front of the I-beam. This feature can be seen in results from various solvers shown in Figure 10.33 and Figure 10.34. The unsteady development of flow for other various solvers is presented in the Appendix.

Presented in Figure 10.36 is the development of pressure at the various locations on the I-beam shown in Figure 10.35. As expected, the points on the leading edge of the I-beam first experience an increase in pressure as the propagating shock strikes the I-beam. As the trapped gas in between the flanges is compressed by the shock and resulting detached shocks from the flanges, the pressure on the surface of the I-beam increases significantly. As the shock travels around the I-beam, the pressure at the top and bottom surfaces increases and

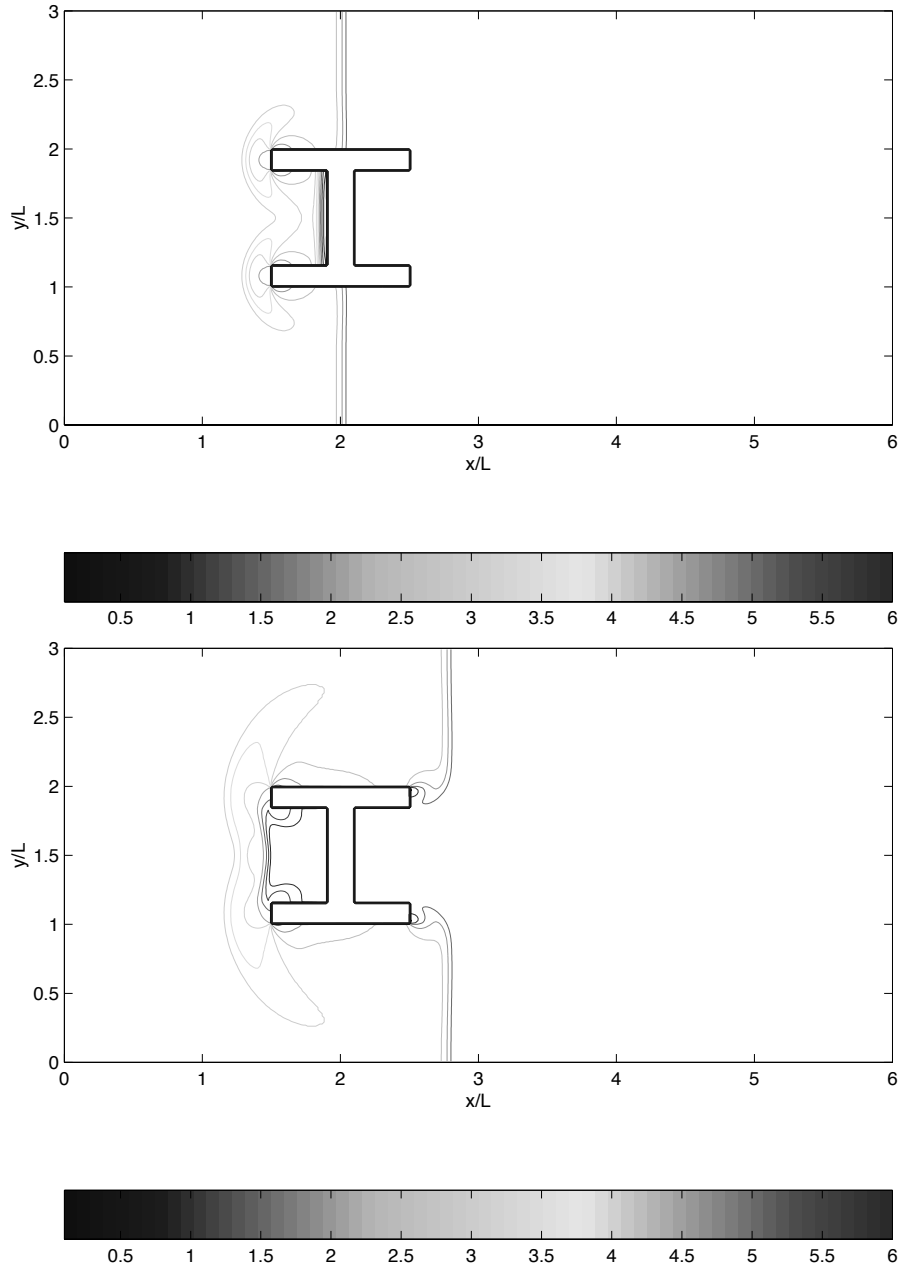

Figure 10.31: Density contours from TDEFM solution for a shock wave interaction with the I-beam shown in Figure 10.30 at times $\sqrt{RT}t/L = 0.052$ and 0.104 where t is the flow time. The computational grid employs 400x200 cells. The gas in assumed inviscid with $\gamma = 1.4$. The approaching shock creates a pressure increase of $p_H/p_L = 4.5$ and a temperature increase of $T_H/T_L = 1.687$. All surface reflections off the I-beam are treated as specular. All boundaries encapsulating the computational grid are extrapolated from internal cells.

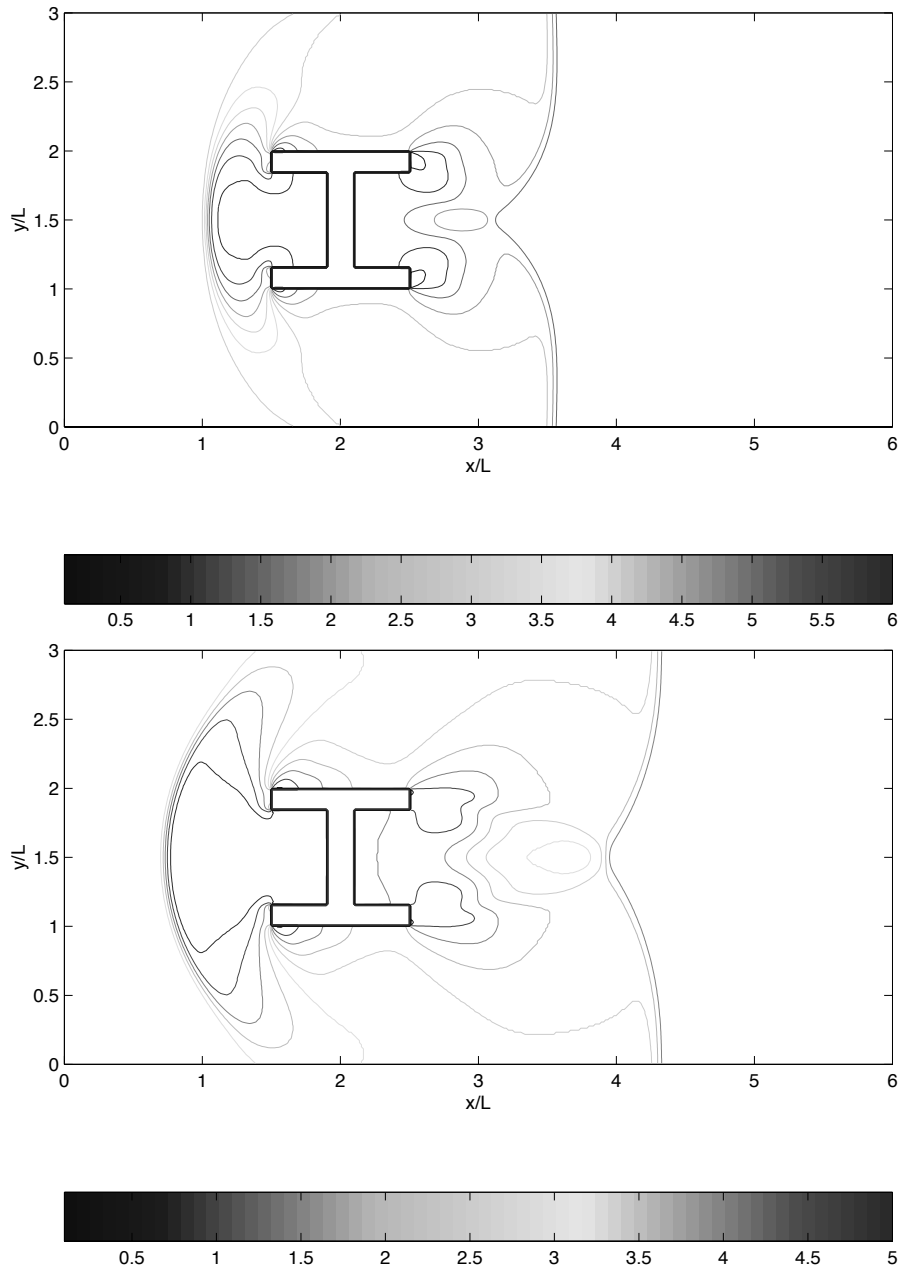

Figure 10.32: Density contours from TDEFM solution for a shock wave interaction with the I-beam shown in Figure 10.30 at times $\sqrt{RT}t/L = 0.156$ and 0.208 where t is the flow time. The computational grid employs 400x200 cells. The gas in assumed inviscid with $\gamma = 1.4$. The approaching shock creates a pressure increase of $p_H/p_L = 4.5$ and a temperature increase of $T_H/T_L = 1.687$. All surface reflections off the I-beam are treated as specular. All boundaries encapsulating the computational grid are extrapolated from internal cells.

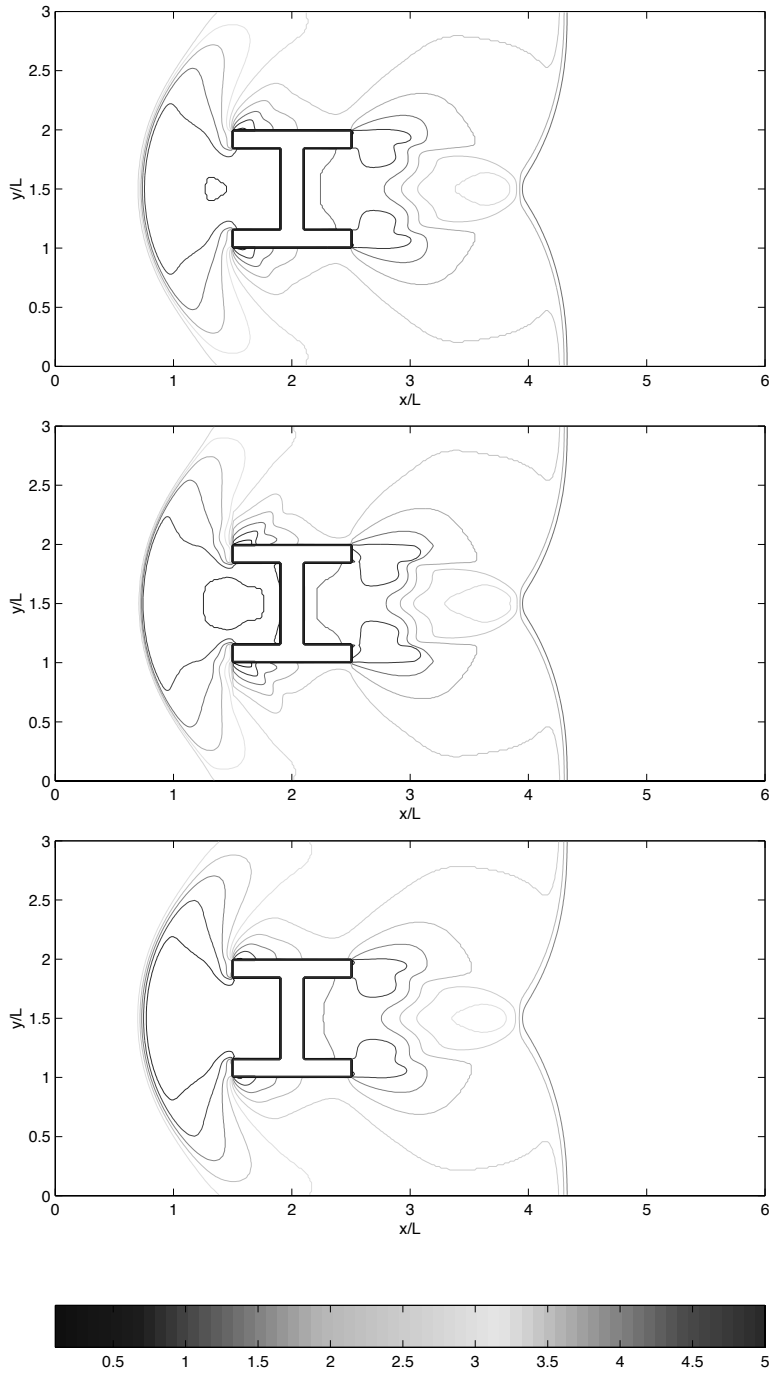

Figure 10.33: Density contours from EFM (Top), Riemann (Middle) and TDEFM (Bottom) solutions for a shock wave interaction with the I-beam shown in Figure 10.30 at time $\sqrt{RT}t/L = 0.208$ where t is the flow time. The computational grid employs 400x200 cells. The gas in assumed inviscid with $\gamma = 1.4$. The approaching shock creates a pressure increase of $p_H/p_L = 4.5$ and a temperature increase of $T_H/T_L = 1.687$. All surface reflections off the I-beam are treated as specular. All boundaries encapsulating the computational grid are extrapolated from internal cells.

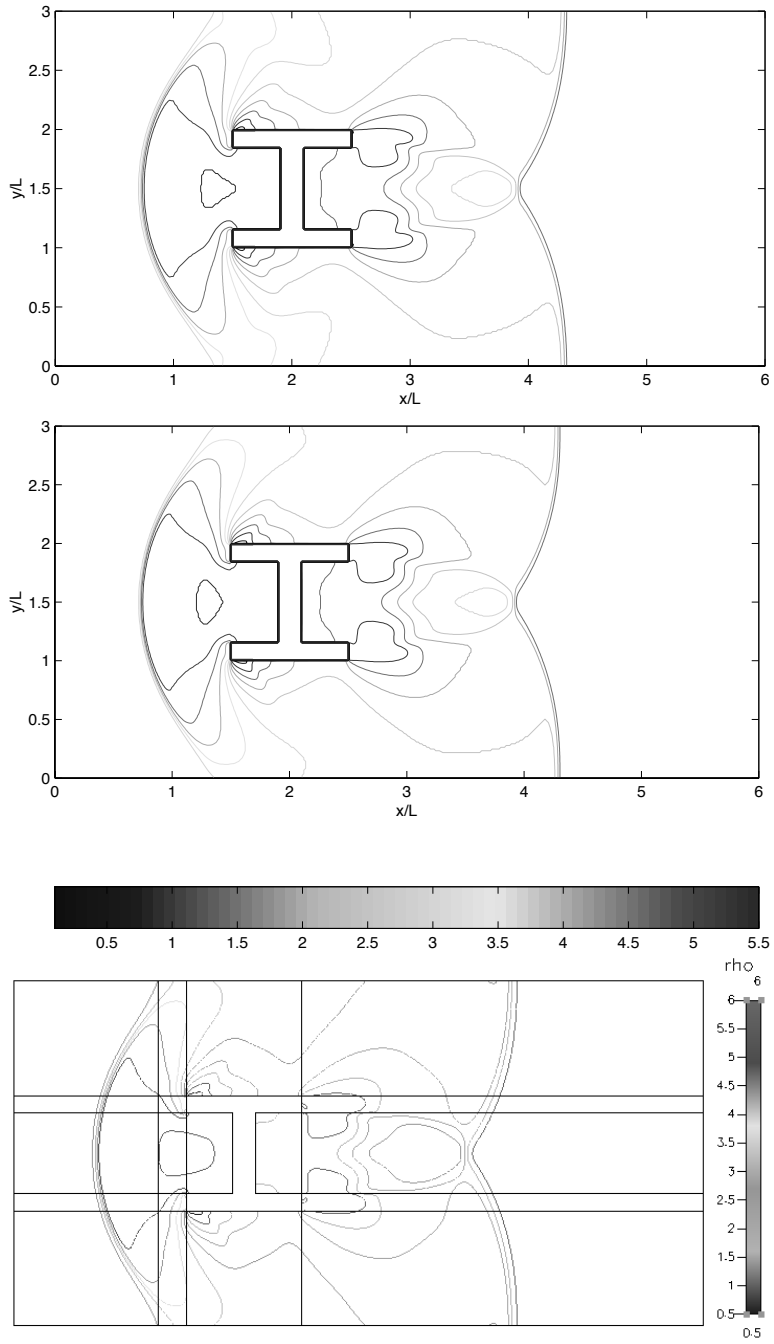

Figure 10.34: Density contours from DTDEFM (Top), VTDEFM (Middle) and CFD-FASTRAN (Bottom) solutions for a shock wave interaction with the I-beam shown in Figure 10.30 at time $\sqrt{RT}t/L = 0.208$ where t is the flow time. The computational grid employs 400x200 cells. The gas in assumed inviscid with $\gamma = 1.4$. The approaching shock creates a pressure increase of $p_H/p_L = 4.5$ and a temperature increase of $T_H/T_L = 1.687$. All surface reflections off the I-beam are treated as specular. All boundaries encapsulating the computational grid are extrapolated from internal cells.

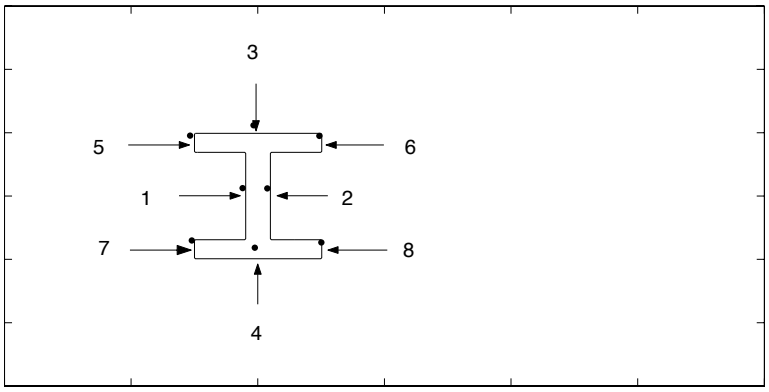

Figure 10.35: Locations of pressure measurements taken from simulations to provide a transient response resulting from blast/shock wave interaction for the I beam shown in Figure 10.30.

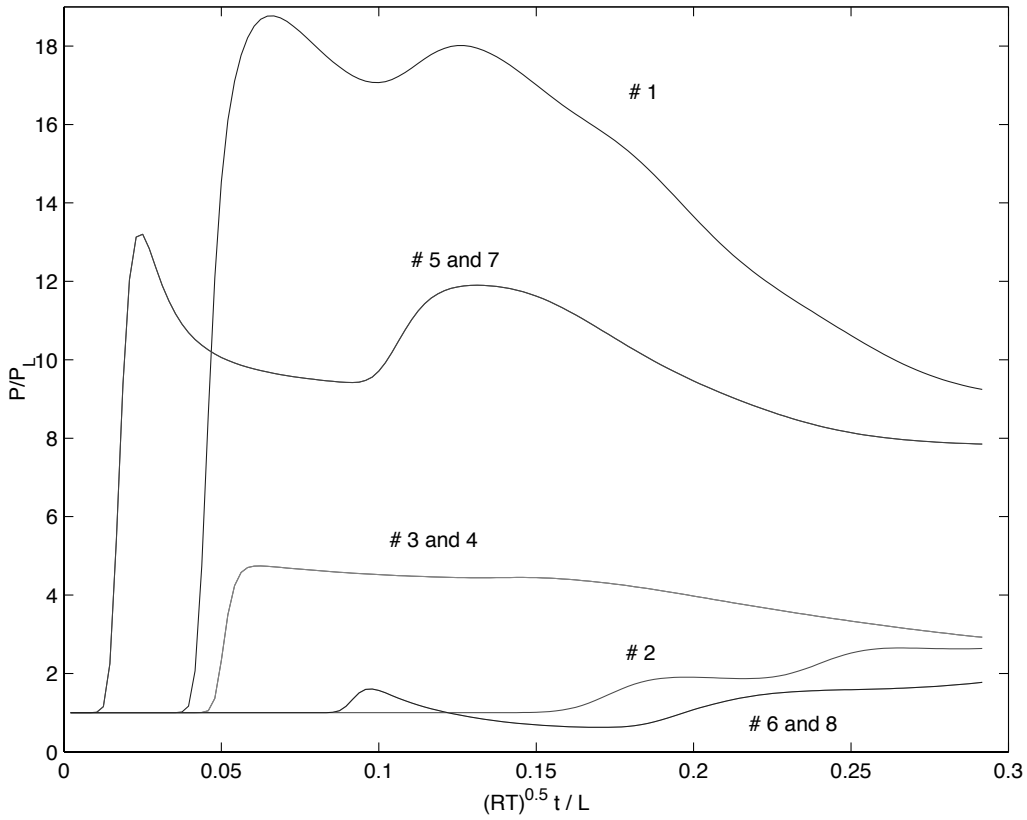

Figure 10.36: Pressure measurements taken from TDEFM simulations showing the development of pressure at the locations shown in Figure 10.35 resulting from blast/shock wave interaction.

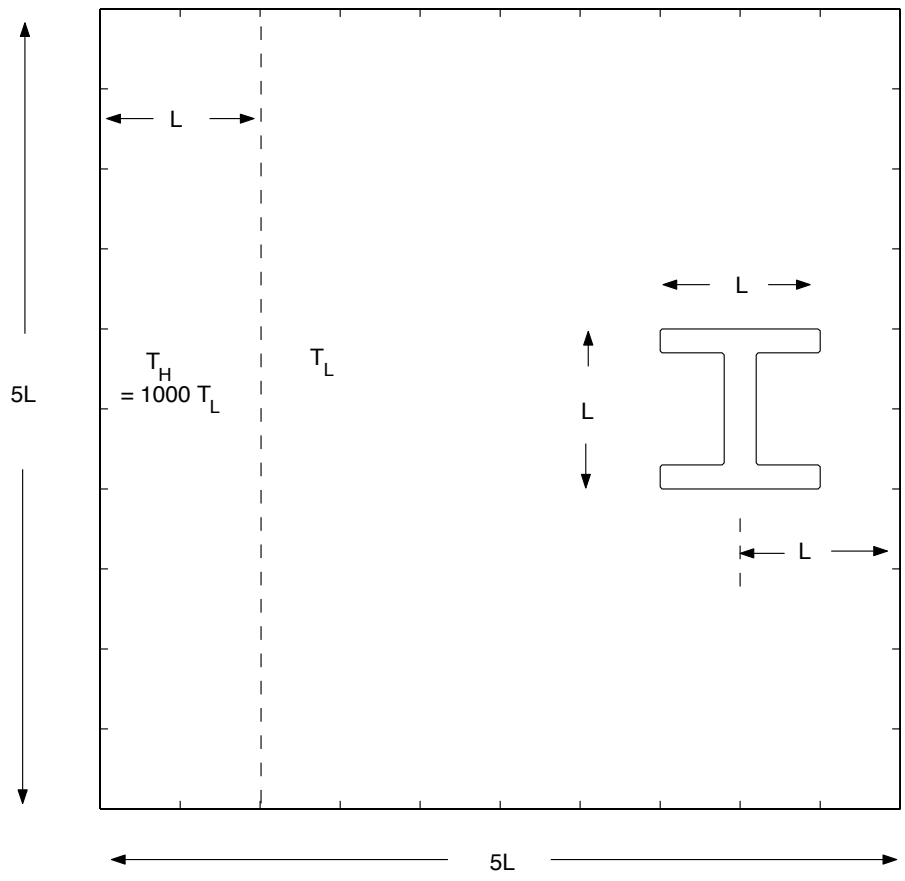

Figure 10.37: I Beam and blast region geometry used for blast wave test.

remains relatively constant. The pressure at the trailing edges of the I-beam increases slightly as the shock defracts around the edges of the I-beam, then decreases to below the free stream conditions due to the expansion of the flow around the corner.

Blast wave interaction with I beam structure

Results are presented for the interaction of a blast wave resulting from a simulated explosion on an I-beam. Rather than modeling the traveling blast wave as a shock wave with constant properties driving the gas forward, the entire processes is captured. The purpose is to demonstrate the effect of an I beam located in close proximity to the explosion source. The geometry of the blast conditions and the I-beam is displayed in Figure 10.37. A region of high temperature gas of width L is given a temperature 1000 times higher than the surrounding ambient gas. The standard I beam, shown in Figure 10.30 and Figure 10.37, is placed a distance of $3L$ (to the I-beam flange) away from the initial diaphragm separating the high and low temperature gas. The air is assumed inviscid with $\gamma = 1.4$. Upon removal of the diaphragm, a blast wave will propagate toward the I-beam. The structure of the blast wave is similar to that demonstrated in the radial blast wave problem - a significant and sudden increase in density and Mach number immediately followed by a rapid reduction in both quantities as the gas propagates outwards.

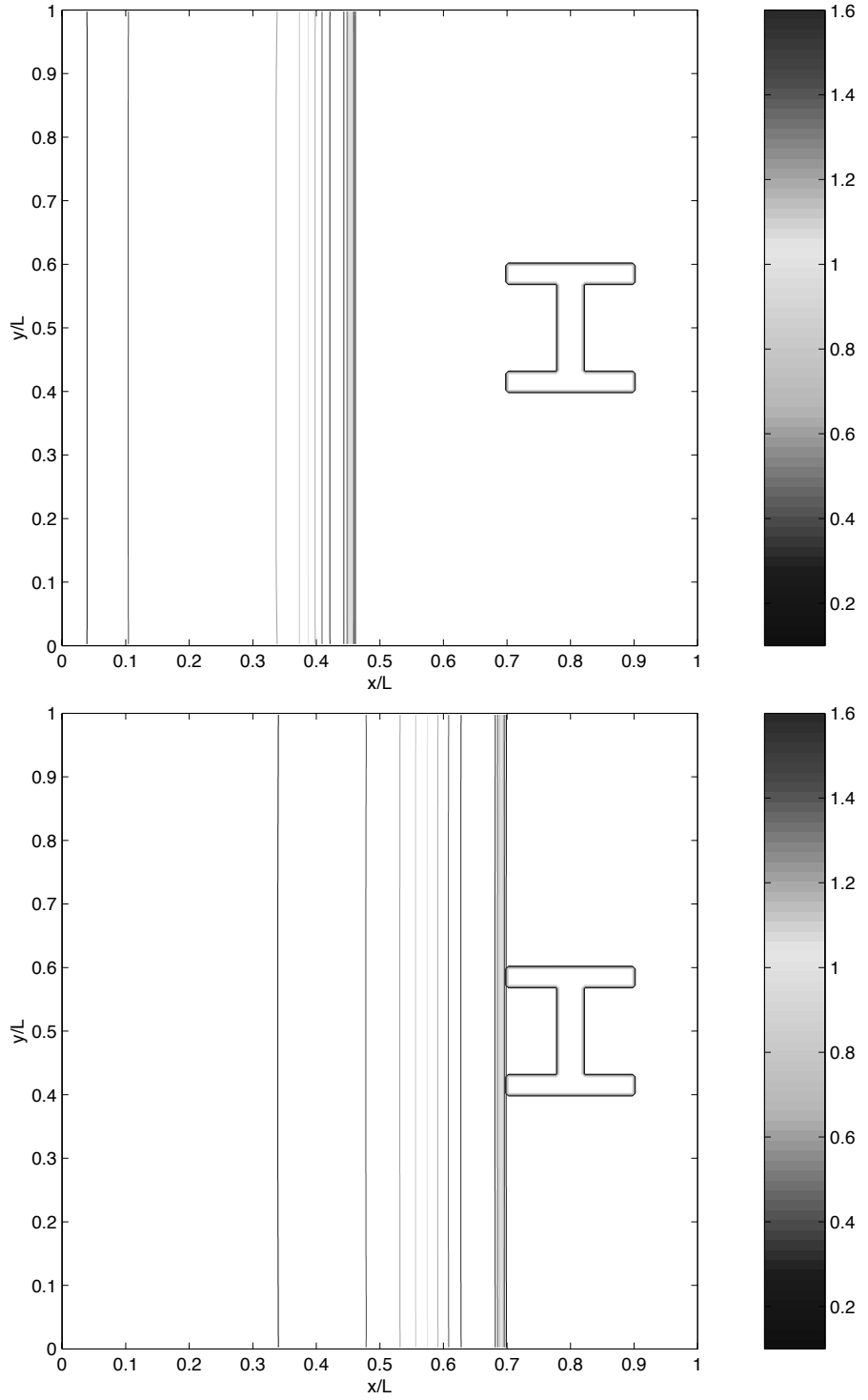

Figure 10.38: Mach number contours from the DTDEFM solution for a blast wave interaction with a the I-beam shown in Figure 10.30 at times $\sqrt{RT}t/L = 0.1$ and 0.2 where t is the flow time. The computational grid employs 200x200 cells. The gas in assumed inviscid with $\gamma = 1.4$. The approaching blast was generated by a region of gas with a temperature 1000 times higher than that of the surrounding fluid. All surface reflections off the I-beam are treated as specular. All boundaries encapsulating the computational grid are extrapolated from internal cells.

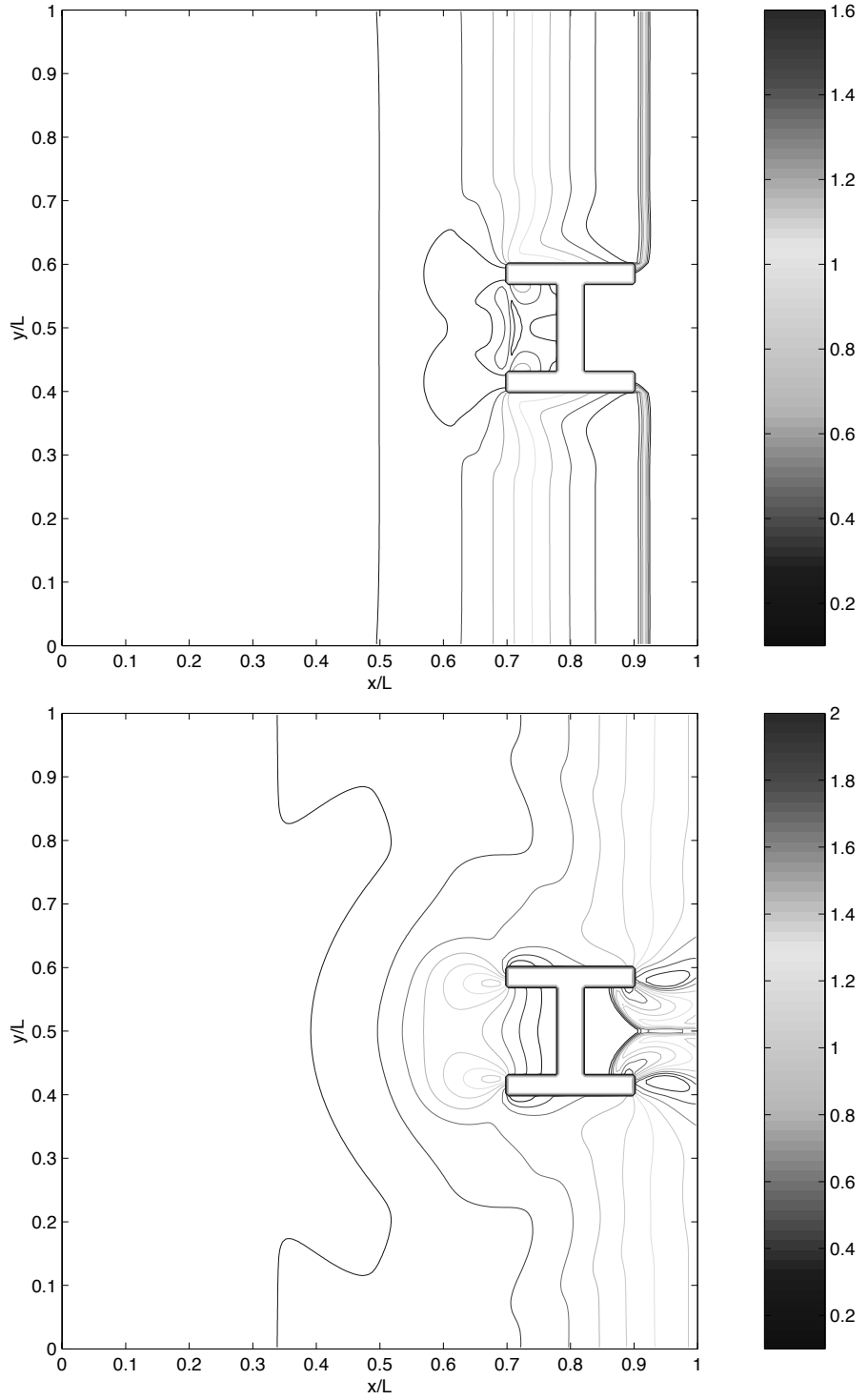

Figure 10.39: Mach number contours from the DTDEFM solution for a blast wave interaction with a the I-beam shown in Figure 10.30 at times $\sqrt{RT}t/L = 0.3$ and 0.4 where t is the flow time. The computational grid employs 200x200 cells. The gas in assumed inviscid with $\gamma = 1.4$. The approaching blast was generated by a region of gas with a temperature 1000 times higher than that of the surrounding fluid. All surface reflections off the I-beam are treated as specular. All boundaries encapsulating the computational grid are extrapolated from internal cells.

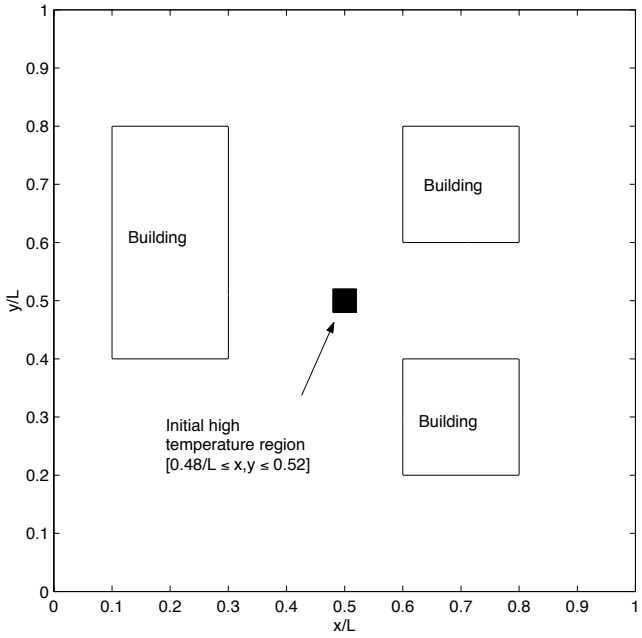

Figure 10.40: City A (3 building configuration) used to simulate blast waves in a city environment.

Figure 10.38 and Figure 10.39 show the unsteady development of the local Mach number in the flow around the I-beam resulting from the simulated explosion. Rather than forming a stationary bow shock as found when simulating a shock wave, the impacting blast wave reflects off the front of the I-beam and pushes forward back into the exploded gas. The resulting flow pattern is more complicated than the previous test cases because of the rapidly changing conditions in the gas behind the shock.

City A - 3 building configuration

Results for two dimensional flow around various rectangular bodies are presented for EFM, Riemann, TDEFM, VTDEFM, DTDEFM and CFD-FASTRAN. The flow is solved using the core engine behind the program FASTWAVE, described in 9. In the following sections, the buildings are treated as indestructible objects which force the flow around and between them. This is for the sake of simplicity as opposed to realism, since genuine blast waves will tend to break past the outer coverings (i.e. windows, glass panels) and cause flow to propagate into the interior of the building. This will be discussed in more detail in following sections. In this instance, two layout configurations are demonstrated (i) a blast wave propagating between and around three buildings, and (ii) the same blast wave propagating around and between nine buildings.

The layout of the buildings and the initial high temperature region are shown in Figure 10.40. The flow is initially at rest, is assumed inviscid and is a monatomic gas (i.e. $\gamma = 5/3$). The surfaces of the buildings are treated as specular and are indestructible. The outer boundaries are treated using the extrapolated outflow condition discussed in Section 9.8. The interior

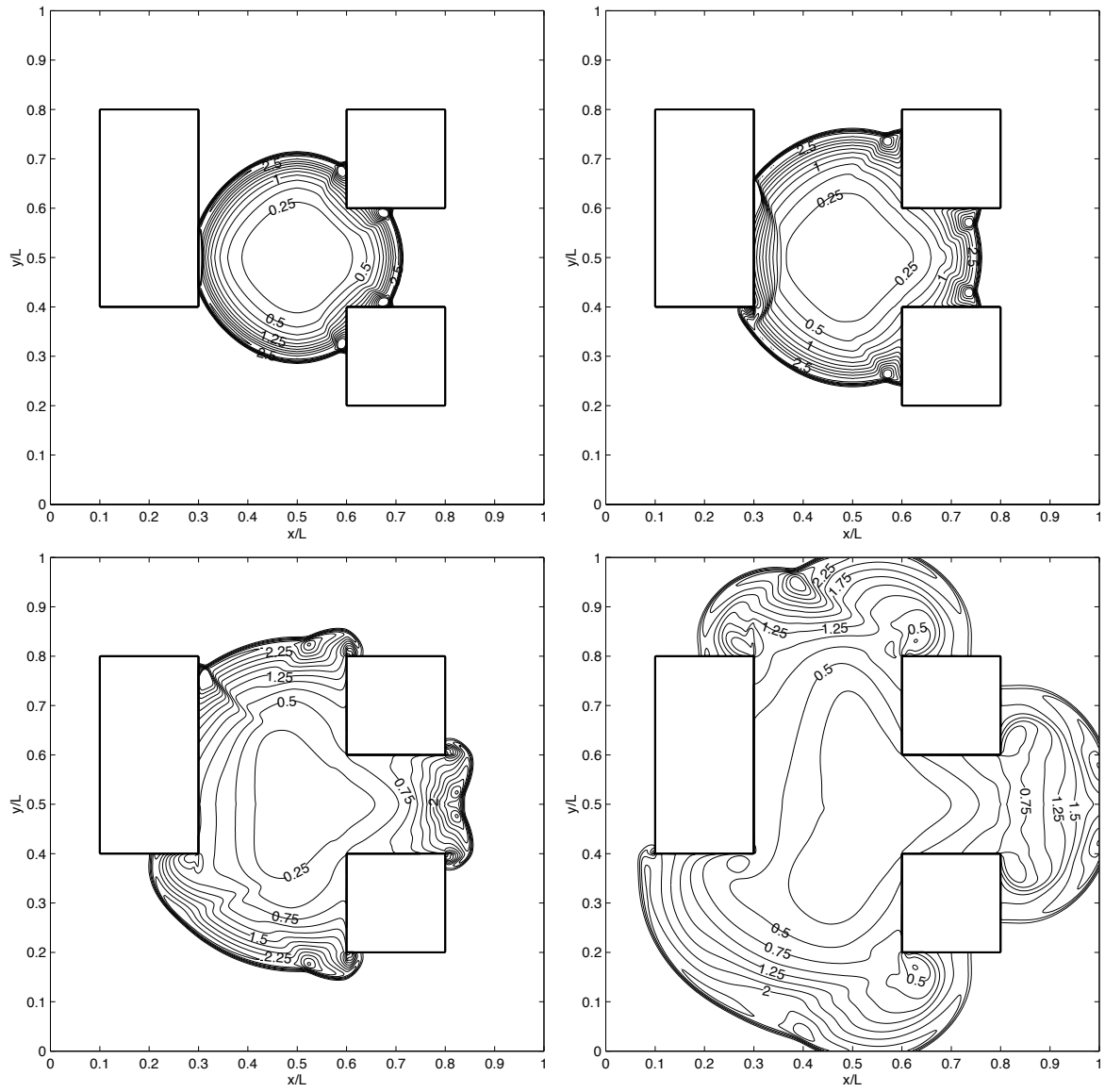

Figure 10.41: Density contours taken from the Roe solver results at times $t\sqrt{RT}/L = 0.02, 0.03, 0.05$ and 0.1. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extrapolated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.40. The computational grid is a cartesian mesh of 400x400 cells.

spaces of the buildings are treated as a solid and no flux calculation is performed. Simulations are performed for various solvers on varying mesh densities.

Figure 10.41 shows density contours as obtained using a Roe solver [44] on a mesh of 400x400 cells after $t\sqrt{RT}/L = 0.01$. The same asymmetric flow features discussed in the analysis of the blast wave problem are demonstrated here, with artificial high pressure regions located in regions of unaligned flow. These high pressure regions deflect the flow from the correct radial direction. This effect is present in all of the Roe solver, EFM and FASTRAN results.

Figure 10.42 and Figure 10.43 shows density contours obtained using EFM and TDEFM on a coarse mesh after $t\sqrt{RT}/L = 0.01$. The same radially asymmetric result obtained using

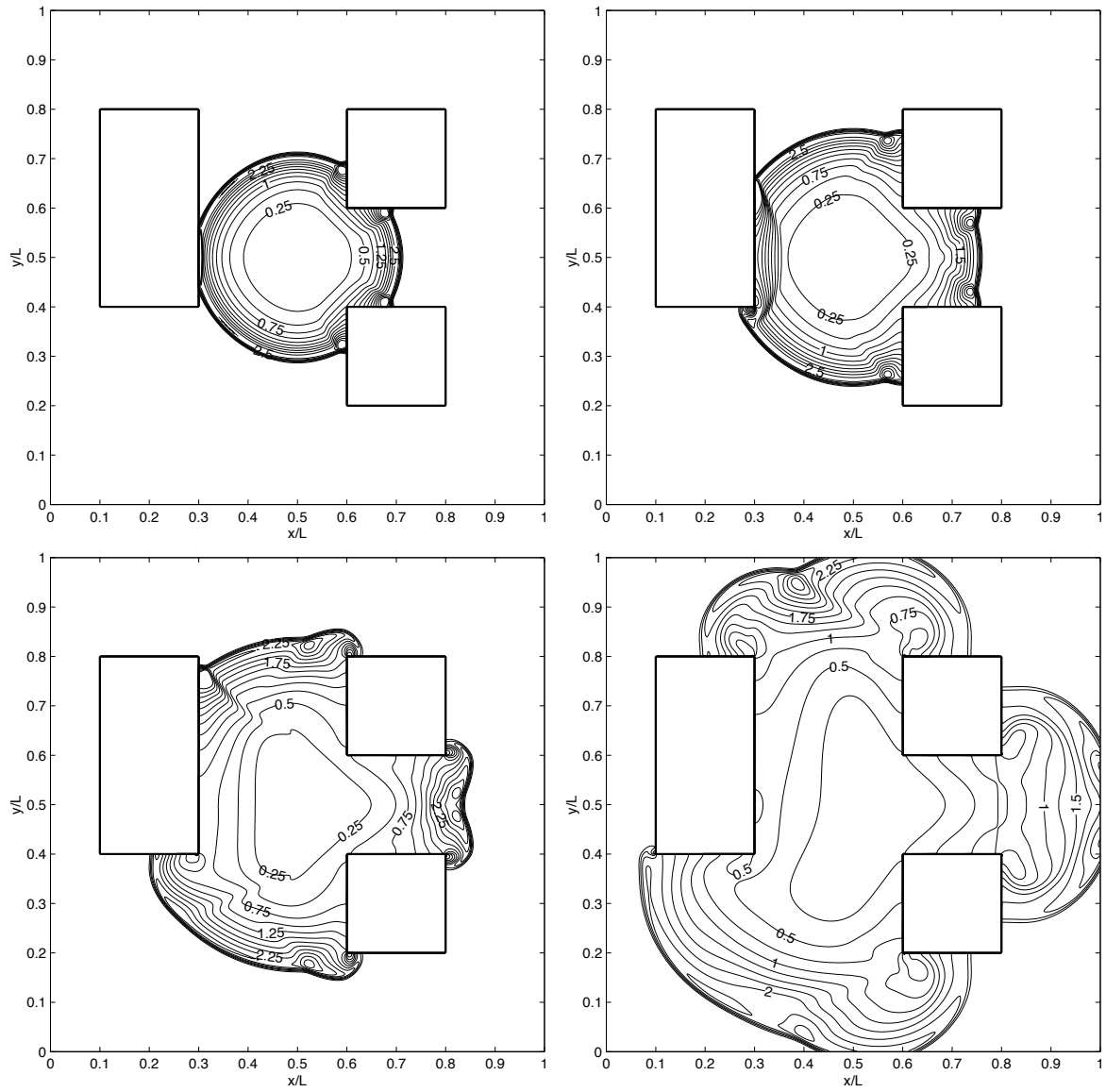

Figure 10.42: Density contours taken from EFM results at times $t\sqrt{RT}/L = 0.02, 0.03, 0.05$ and 0.1. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extrapolated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.40. The computational grid is a cartesian mesh of 400x400 cells.

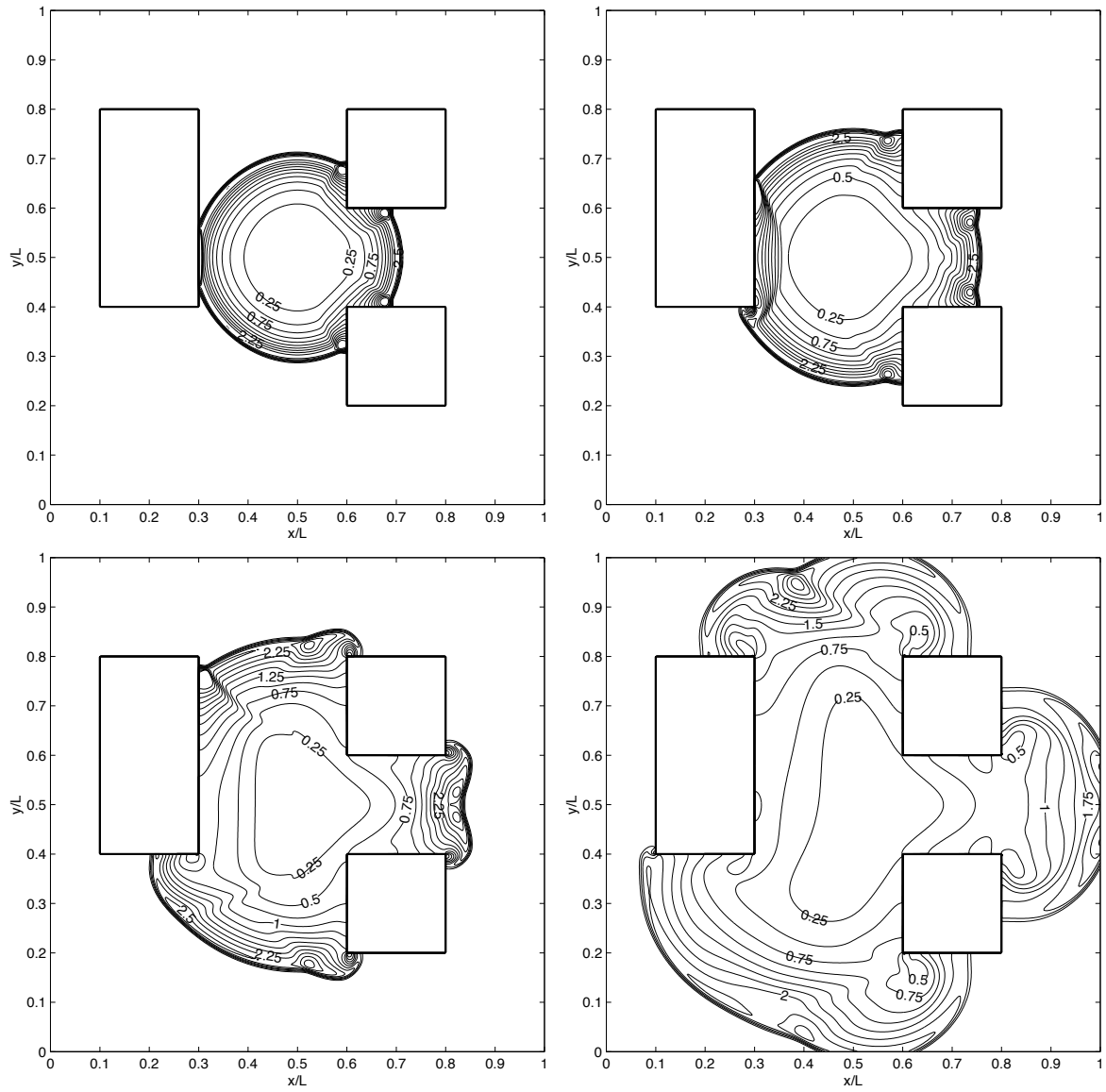

Figure 10.43: Density contours taken from TDEFM results at times $t\sqrt{RT}/L = 0.02, 0.03, 0.05$ and 0.1. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extrapolated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.40. The computational grid is a cartesian mesh of 400x400 cells.

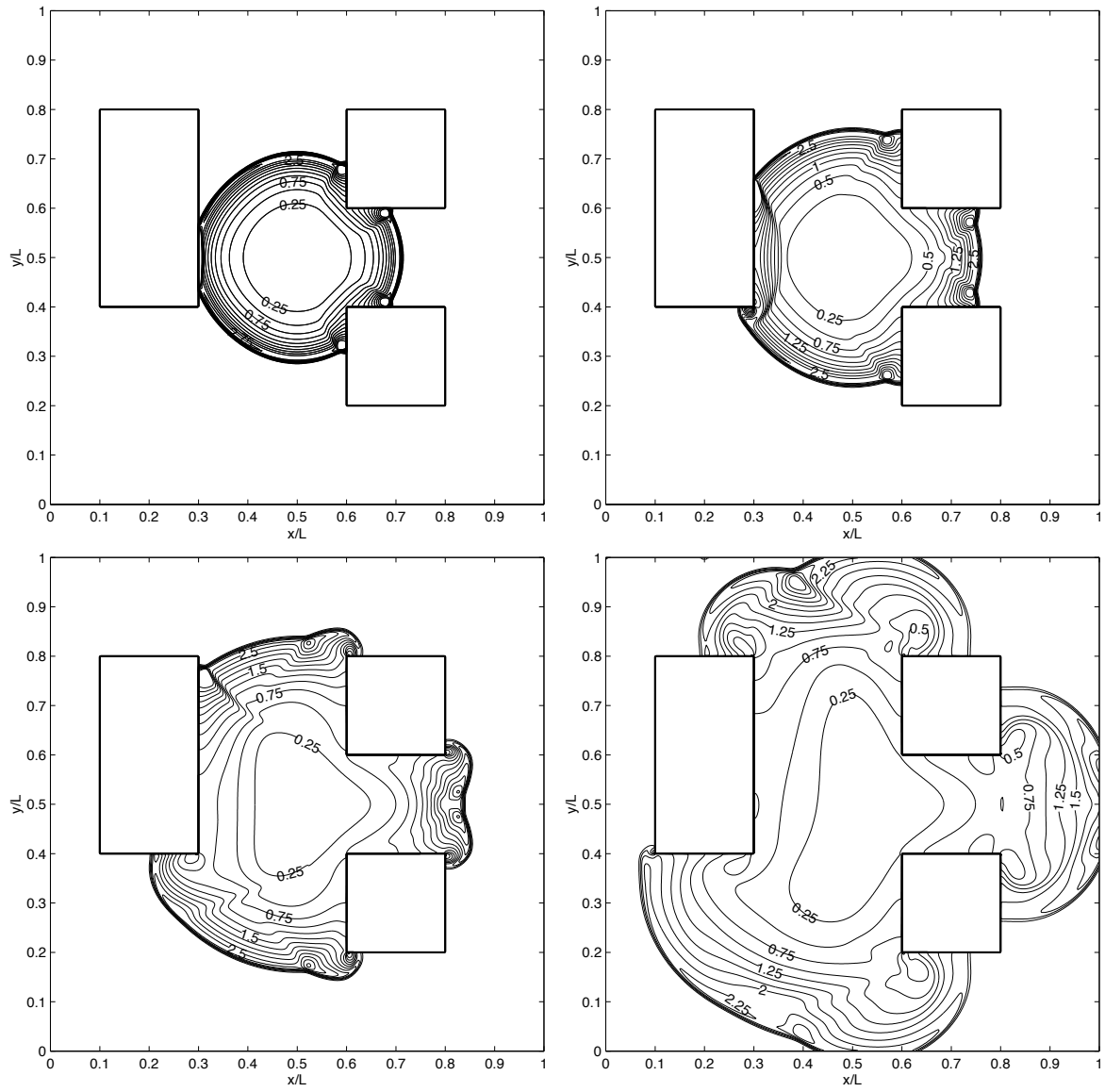

Figure 10.44: Density contours taken from DTDEFM results at times $t\sqrt{RT}/L = 0.02, 0.03, 0.05$ and 0.1. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extrapolated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.40. The computational grid is a cartesian mesh of 400x400 cells.

FASTRAN and a Roe solver is present in the EFM results. The direction coupled nature of TDEFM attempts to spread these high pressure regions in a radially symmetric fashion. This is not a consequence of the use of TDEFM - the method is not more inclined to solve radial flows more accurately - rather, the method attempts to capture the true motion of the fluid regardless of mesh alignment.

The results from DTDEFM and VTDEFM are shown in Figure 10.44 and Figure 10.45. The effect of the inclusion of flow gradients into the flux expressions is to improve the accuracy of the solution despite the nature of the coarse mesh. The solution obtained by DTDEFM demonstrates this, as no artificially high pressure regions exist in its solution. While the

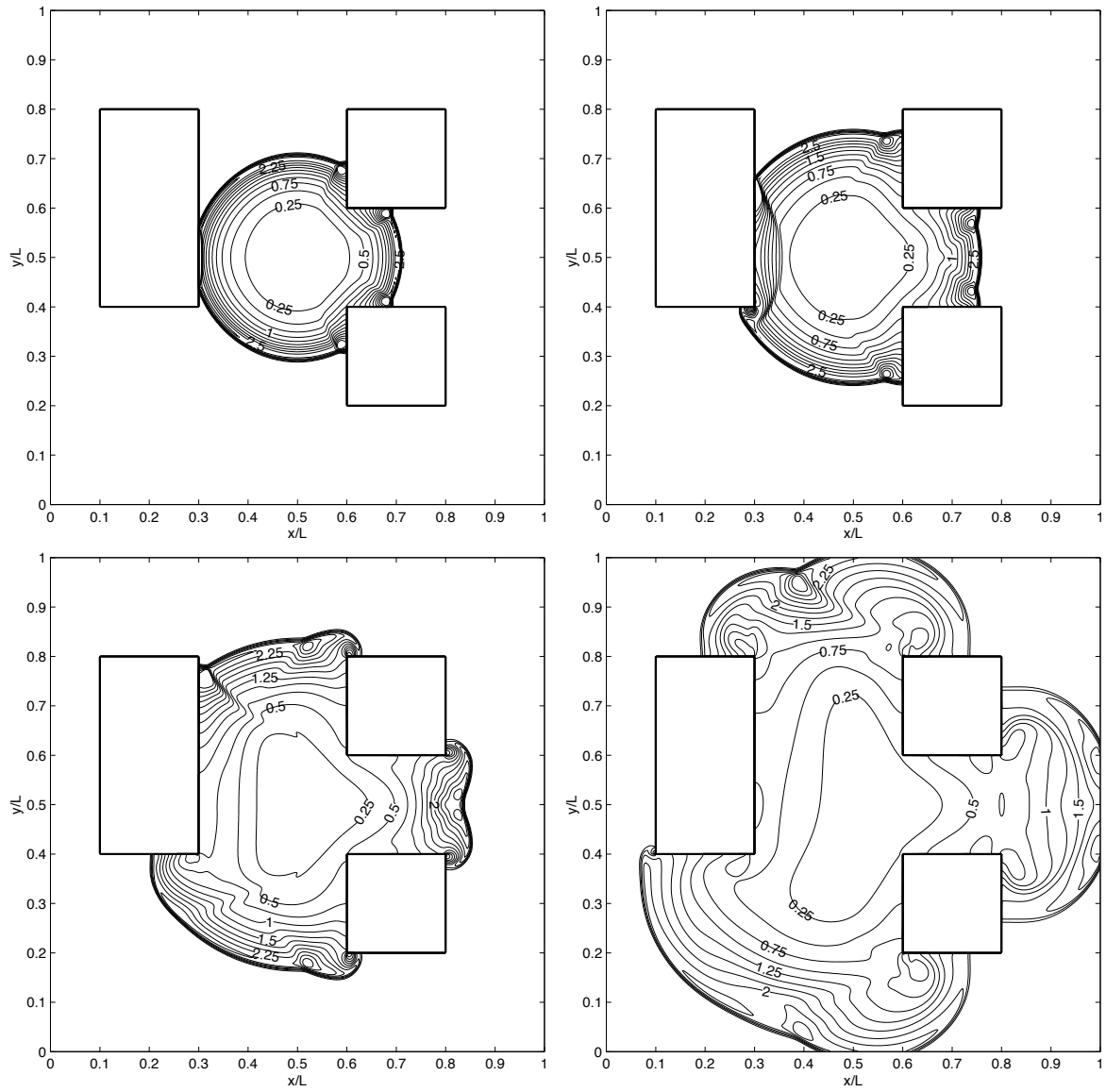

Figure 10.45: Density contours taken from VTDEFM results at times $t\sqrt{RT}/L = 0.02, 0.03, 0.05$ and 0.1. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extrapolated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.40. The computational grid is a cartesian mesh of 400x400 cells.

inclusion of a velocity gradient into the flux expressions improves the result, the improvement is not as significant as with DTDEFM.

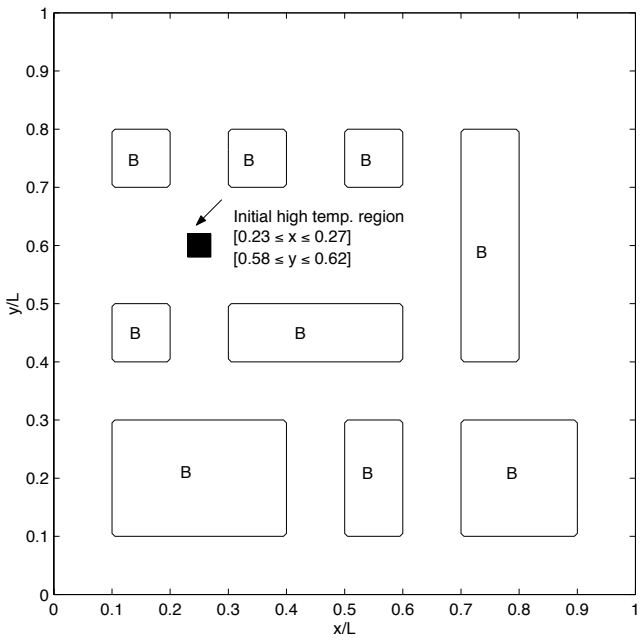

Figure 10.46: City B (9 building configuration) used to simulate blast waves in a city environment.

City B - 9 building configuration

The layout of the buildings and the initial high temperature region are shown in Figure 10.46. As in the previous test case, the flow is initially at rest, is assumed inviscid and is a monatomic gas (i.e. $\gamma = 5/3$). The surfaces of the buildings are again treated as specular and are indestructible. Simulations are performed for various solvers on varying mesh densities - presented here are results from simulations using 160000 cells.

Figure 10.47 displays the unsteady development of density resulting from a blast wave taken from the Roe solver results. The results generally agree well with the unsteady development of the EFM, TDEFM, DTDEFM and VTDEFM results displayed in Figure 10.48, Figure 10.49, Figure 10.50 and Figure 10.51. The initial blast propagates outwards, resulting in a complicated flow pattern as the exploded gas is channeled between buildings. All of the presented results were calculated using FASTWAVE and required no more than 10 minutes on PC with a single 3.0Ghz processor and 1Gb of ram.

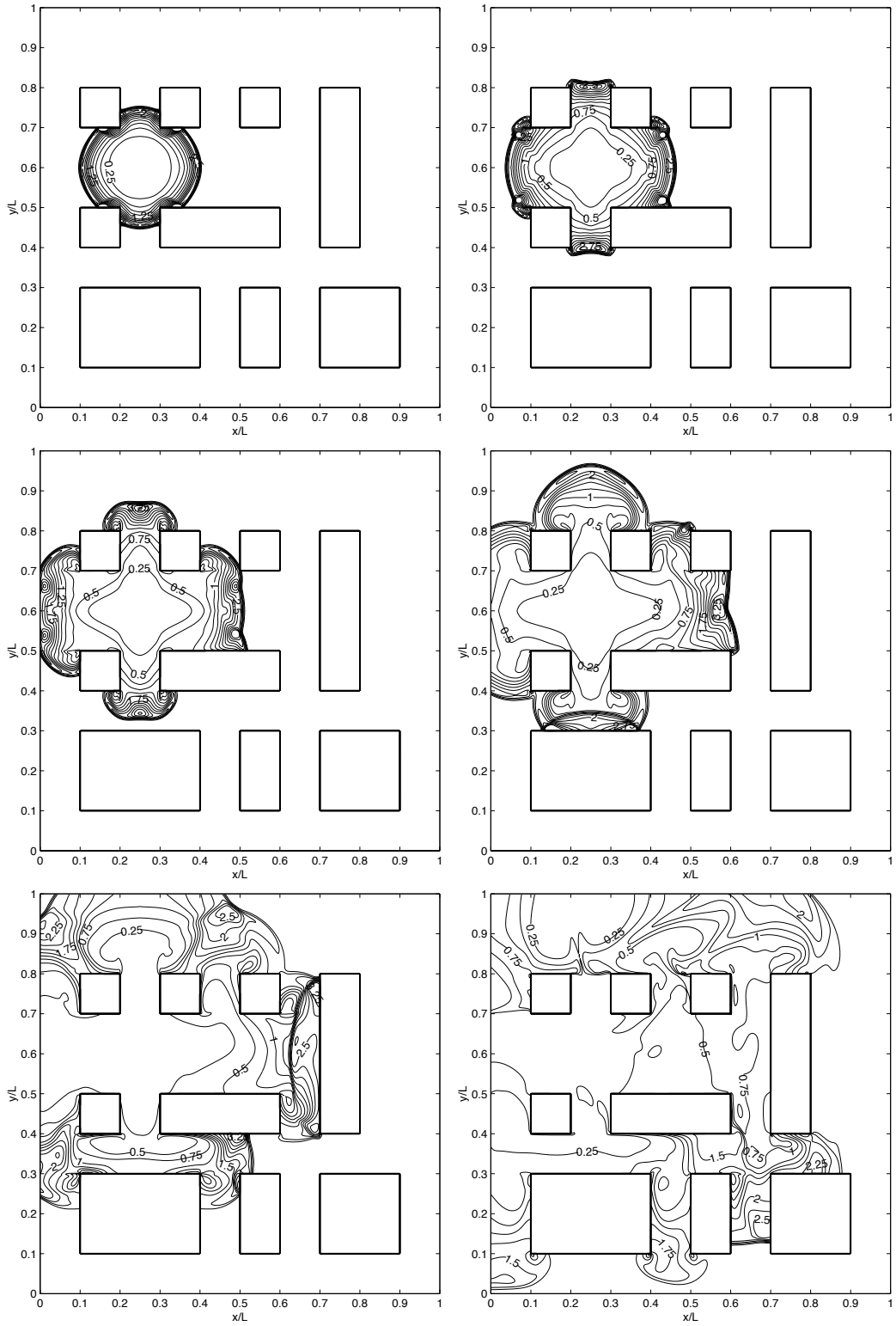

Figure 10.47: Density contours taken from the Roe solver results at times $t\sqrt{RT}/L = 0.01, 0.02, 0.03, 0.05, 0.1$ and 0.2. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extrapolated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.46. The computational grid is a cartesian mesh of 400x400 cells.

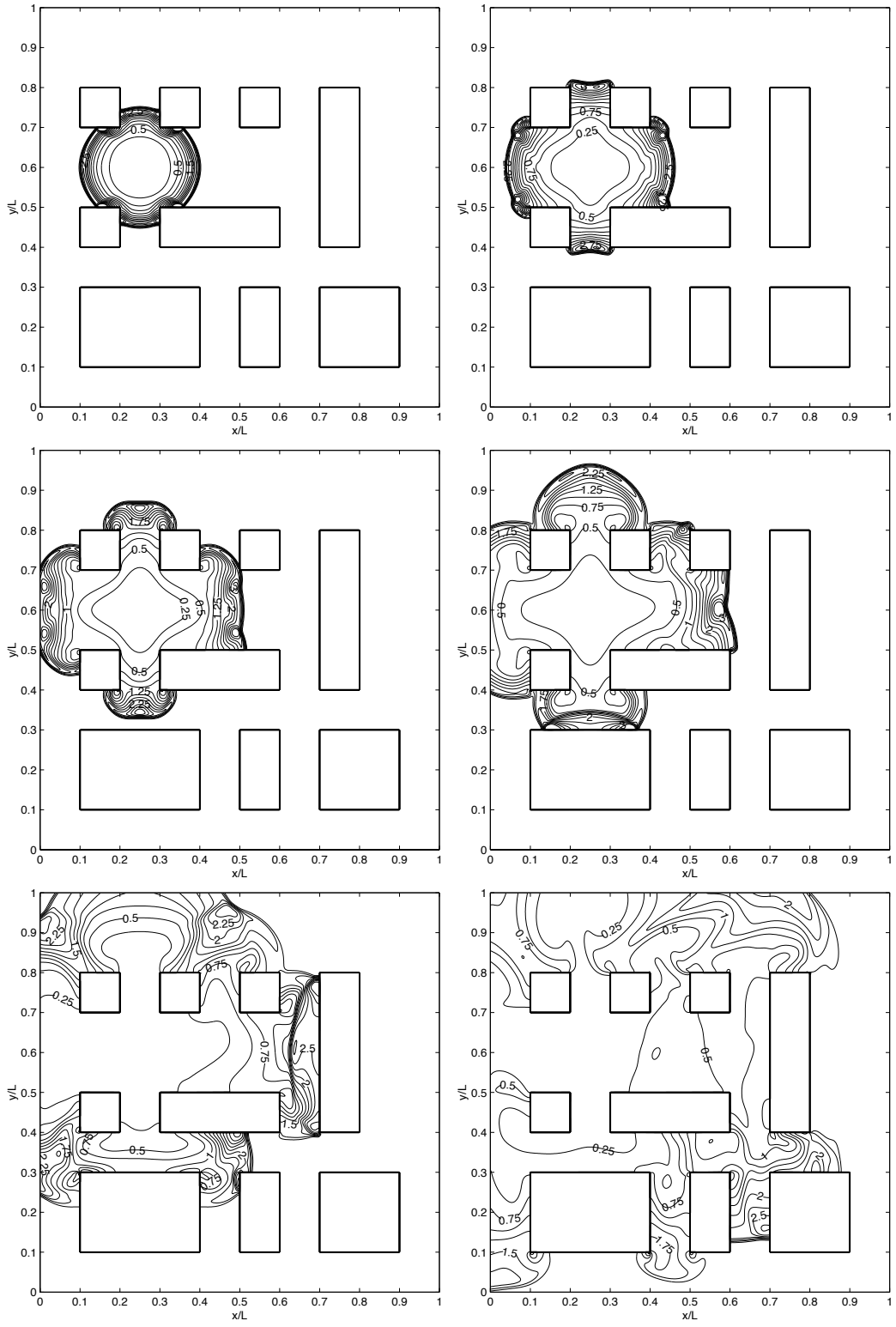

Figure 10.48: Density contours taken from EFM results at times $t\sqrt{RT}/L = 0.01, 0.02, 0.03, 0.05, 0.1$ and 0.2. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extrapolated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.46. The computational grid is a cartesian mesh of 400x400 cells.

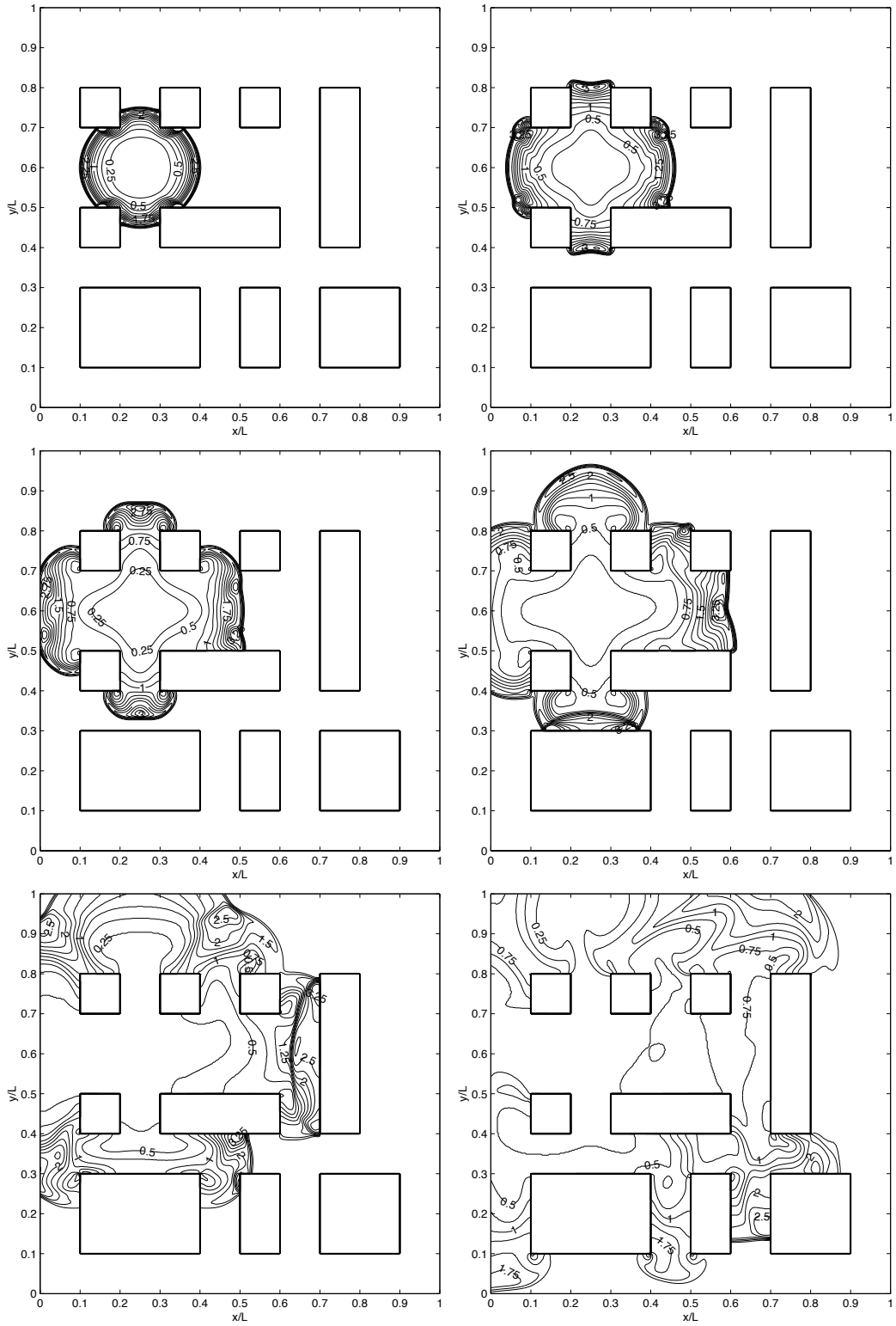

Figure 10.49: Density contours taken from TDEFM results at times $t\sqrt{RT}/L = 0.01, 0.02, 0.03, 0.05, 0.1$ and 0.2. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extrapolated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.46. The computational grid is a cartesian mesh of 400x400 cells.

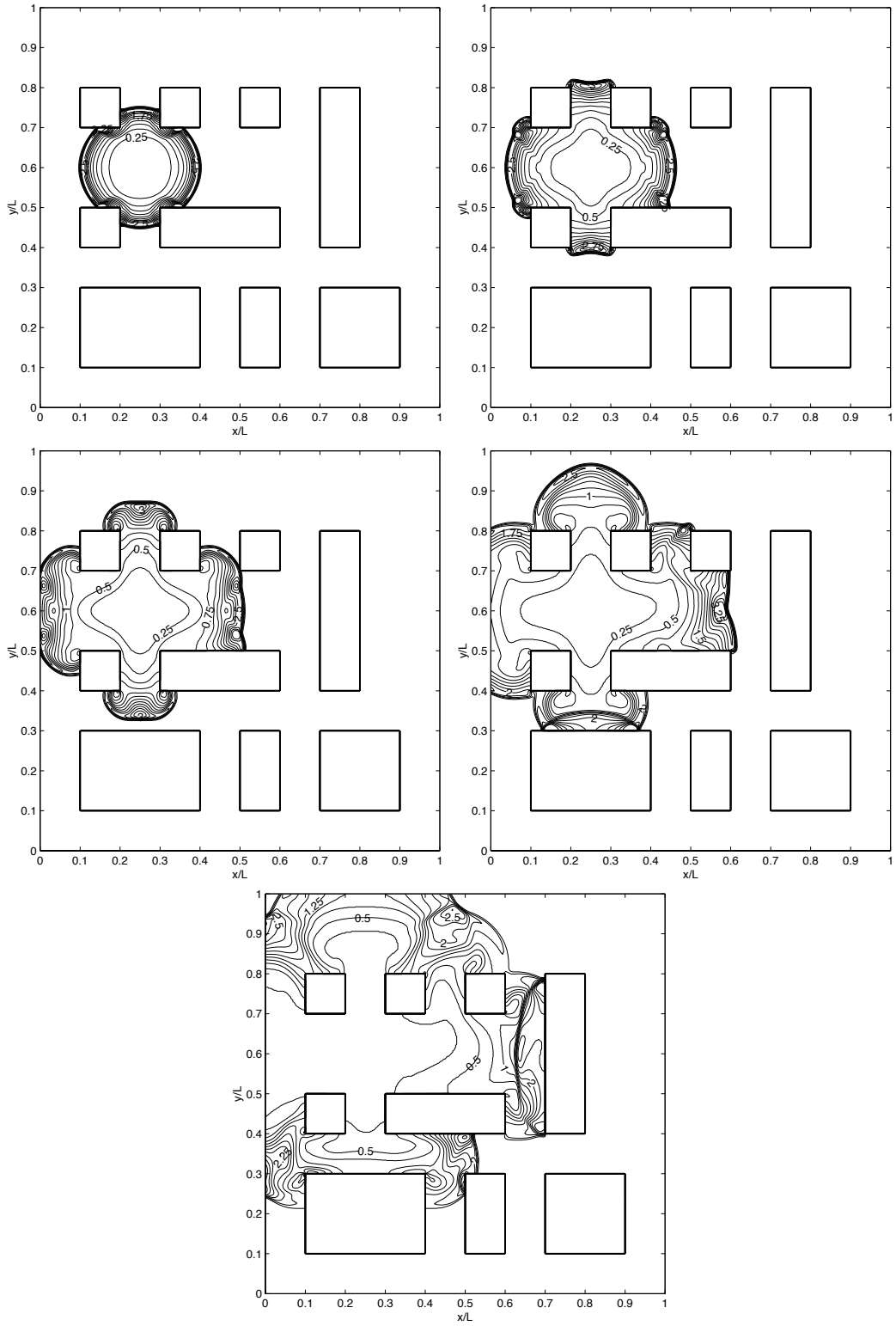

Figure 10.50: Density contours taken from DTDEFM results at times $t\sqrt{RT}/L = 0.01, 0.02, 0.03, 0.05$ and 0.1. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extrapolated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.46. The computational grid is a cartesian mesh of 400x400 cells.

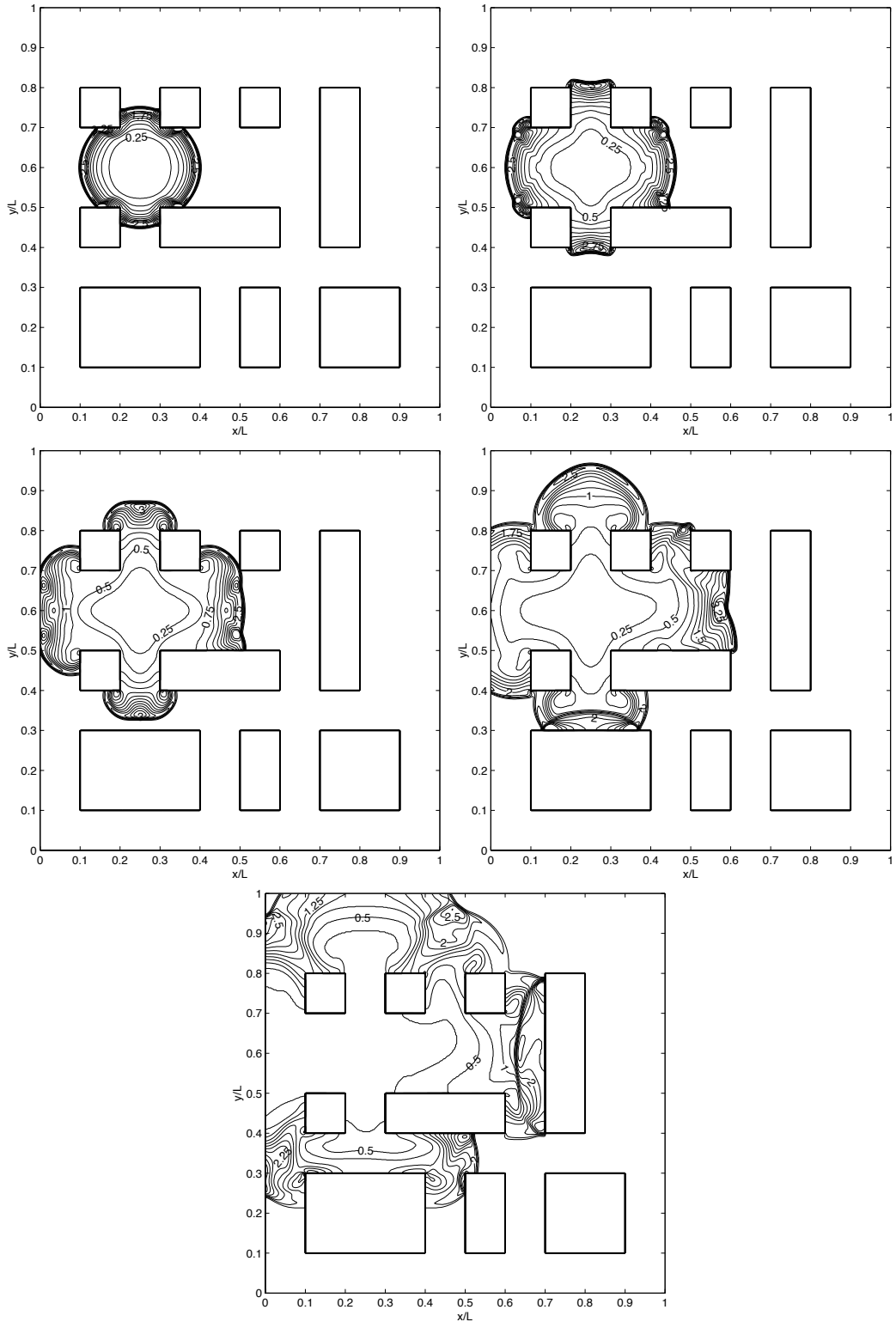

Figure 10.51: Density contours taken from VTDEFM results at times $t\sqrt{RT}/L = 0.01, 0.02, 0.03, 0.05$ and 0.1. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extrapolated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.46. The computational grid is a cartesian mesh of 400x400 cells.

Chapter 11

Analysis and Discussion

11.1 TDEFM flux transportation

Conventional finite volume solvers calculate states at the interfaces between cells and use these to estimate fluxes of mass, momentum and energy across cell surfaces. In kinetic theory based solvers such as EFM, the molecular velocity distribution is examined at the cell interfaces only and any particles with positive relative overall velocity are moved across the interface into the adjascent cell. While these approaches are correct, they contain no information regarding where the gas molecules originated from or which region specifically the molecules will fall

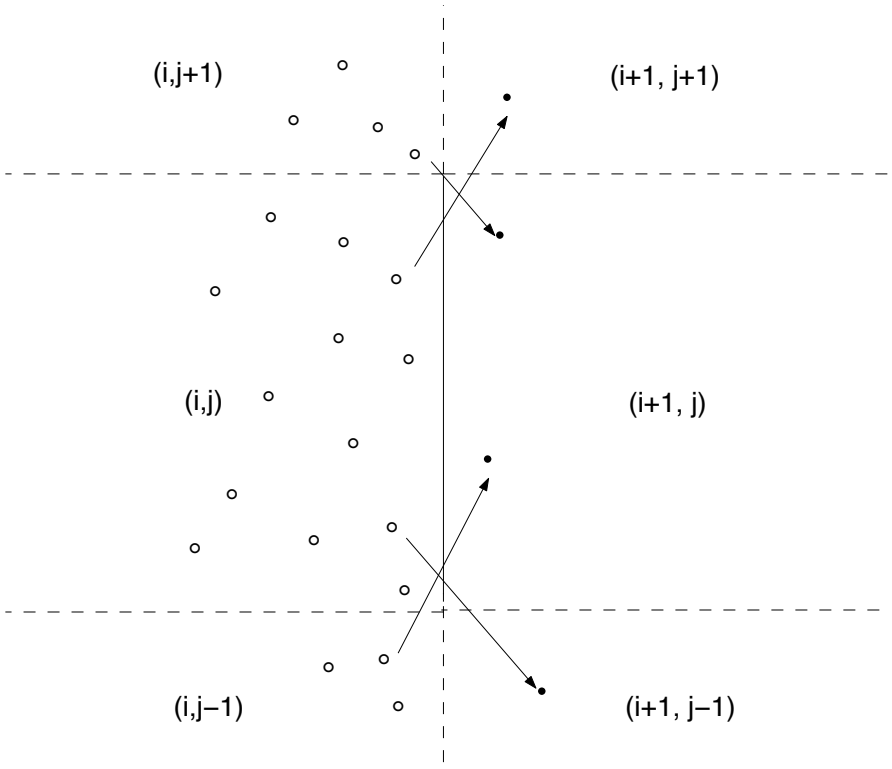

Figure 11.1: Finite volume discretisation using a cartesian grid. Molecules are shown fluxing across the cell surface shared by cells (i, j) and $(i+1, j)$. Traditional CFD finite volume solvers would calculate this flux, subtract the fluxes of conserved quantities from (i, j) and add them to cell $(i + 1, j)$ despite the fact that a fraction of the mass did not originate from cell (i, j) and, likewise, a fraction of the fluxed mass is not moving into cell $(i + 1, j)$.

into. An example of the fluxes theoretically calculated by EFM are present is diagramatically represented in Figure 11.1.

Another consideration is the number of particles in the source region. Conventional CFD finite volume solvers calculate fluxes which are independent of the cell size and the finite mass located in each cell volume. The assumption that these fluxes are built upon are justified since the fluxes of mass, momentum and energy typically represent the instantaneous rate of fluxes across a cell interface. This is usually a very good approximation, especially for the Riemann class solvers which calculate the state between propagating waves which should remain constant regardless of time. The timestep in such solvers is manually limited to ensure the propagating waves do not travel further than a cell width.

Therefore, the flux calculated over a surface by a kinetic theory based solver (like EFM) is the correct flux over a surface for a given small timestep but does not accurately represent the flux from a source finite volume to a destination finite volume. This is because there are components of the EFM flux that do not originate in the source volume, and similarly there are components of the flux which do not move to the destination volume. The fluxes determined by TDEFM represent analytical solution to the free flight phase of a direct simulation in the limit of an infinite number of simulation particles, thermal equilibrium and uniform conditions. Referring to Figure 11.1, the flux of mass from cell (i,j) to cell (i+1,j) calculated by TDEFM will be less than that calculated by EFM. The difference of between fluxes that cross the surface seperating cell (i,j) and (i+1,j) is recovered by the TDEFM fluxes travelling from cells (i,j-1) and (i, j+1) to the destination cell (i+1,j). The net forward flux through the surface seperating the cells (i,j) and (i+1,j) calculated by TDEFM will be identical to that calculated by EFM only if the time step is very small and the conditions in cells (i,j-1) and (i,j+1) are the same as the conditions in cell (i,j).

11.2 Effects of Direction Decoupling

In order to quantify the effect of direction decoupling, we use an "angle of deviation", designated as θ, to measure the radial symmetry present in the solution. This concept is applied to the two dimensional blast wave and implosion problem because of the approximately radially symmetric nature of the flow. The angle of deviation is defined as the angle between the radial position vector $\vec{r} = (x\boldsymbol{i} + y\boldsymbol{j})$ and the velocity vector $\vec{v} = (V_x\boldsymbol{i} + V_y\boldsymbol{j})$, and is given by:

$$\theta = \cos^{-1}\left(\frac{\vec{v}\cdot\vec{r}}{|\vec{r}||\vec{v}|}\right). \tag{11.1}$$

If the initial conditions were perfectly circular in shape, this angle should be zero because of the radially symmetric nature of the flow. The magnitude of θ at any position is a measure of radial asymmetry in the flow and therefore a measure of error. Although the initial conditions used in the blast wave and implosion problem are applied on rectangular cartesian grid and are

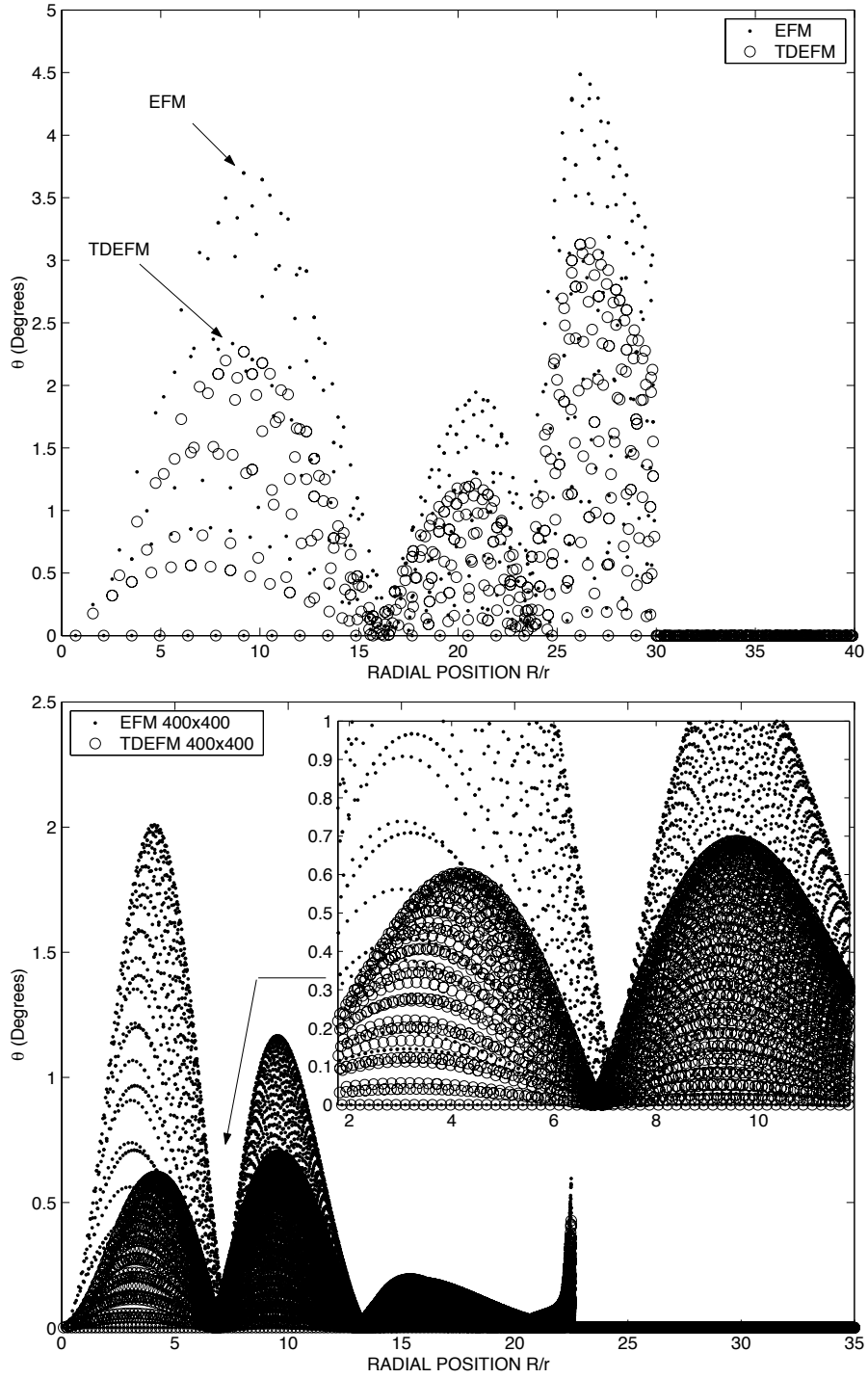

Figure 11.2: Angle of deviation for 2D-TDEFM and 2D-EFM for each cell versus radial position in the blast wave problem. (Top) 50x50 cells, (Bottom) 400x400 cells. Simulations are run up to $t\sqrt{RT}/r = 0.00196$. Each point represents the angle of deviation (*i.e.* the angular difference between the radial position vector and the velocity vector) for a given cell.

not perfectly circular, the quality of the flow approaches radial symmetry as the distance from the origin increases or the mesh resolution increases.

Figure 11.2 shows that deviation angle θ taken from the 2D-EFM and 2D-TDEFM results with meshes of 50x50 and 400x400 cells for the 2D blast wave problem. It is clear that the angle of deviation is consistently less for TDEFM than for EFM, indicating a higher level of fidelity. This fact remains true regardless of mesh density - simulations using much finer meshes ($\gg$ 2 million cells) have revealed that the magnitude of the angle of deviation is always lower in TDEFM results than in EFM results. Therefore, there is always an effect due to direction coupling, regardless of mesh density, although this effect diminishes as mesh density increases. This effect is also demonstrated in the results from the implosion problem shown in Figure 11.3.

The effect of direction decoupling is further demonstrated in the simulation of hypersonic flow over a rectangular body. As a result of the additional time step required to transverse to diagonally adjacent cells, the bow shock is pushed further in front and away from the body. This is demonstrated in Figure 11.4. Despite the increase in the resolution of the computational grid in the direction decoupled techniques to compensate for the additional computational expense, the direction coupled solution still provides a better result. This is confirmed by the location of the bow shock as calculated using Pullin's EPSM, which should be equivalent to collision limited DSMC. This is demonstrated in Figure 10.19 and Figure 10.20. These direct simulations, which are not susceptible to the failings of direction decoupling, provide predicted bow shock locations closer to the body in agreement with the results obtained from TDEFM and its derivatives DTDEFM and VTDEFM.

The differences between direction decoupled EFM and the direction coupled TDEFM results for simulation of a blast wave in a city environment are presented. The percentage difference in density at various flow times is presented in Figure 11.5 and Figure 11.6. The greatest differences in the results occur where the flow is both relatively fast and unaligned with the computational grid. This supports our previous hypothesis regarding the description of the errors resulting from direction decoupling. These errors are significant, resulting in a difference of at least 25 percent in density at multiple locations. The use of direction coupled fluxes is therefore justified despite the slight increase in computational expense.

It is important to note that the shape of the blast wave in the early stages of the blast is not perfectly circular. This can be explained through several arguments:

- The initial shape of the high temperature region is rectangular and several cells in size. In the results shown in Figure 11.5 and Figure 11.6 the initial high temperature region is 16x16 cells in size. The size of this rectangular region represents a significant fraction of the size of buildings in the vicinity.

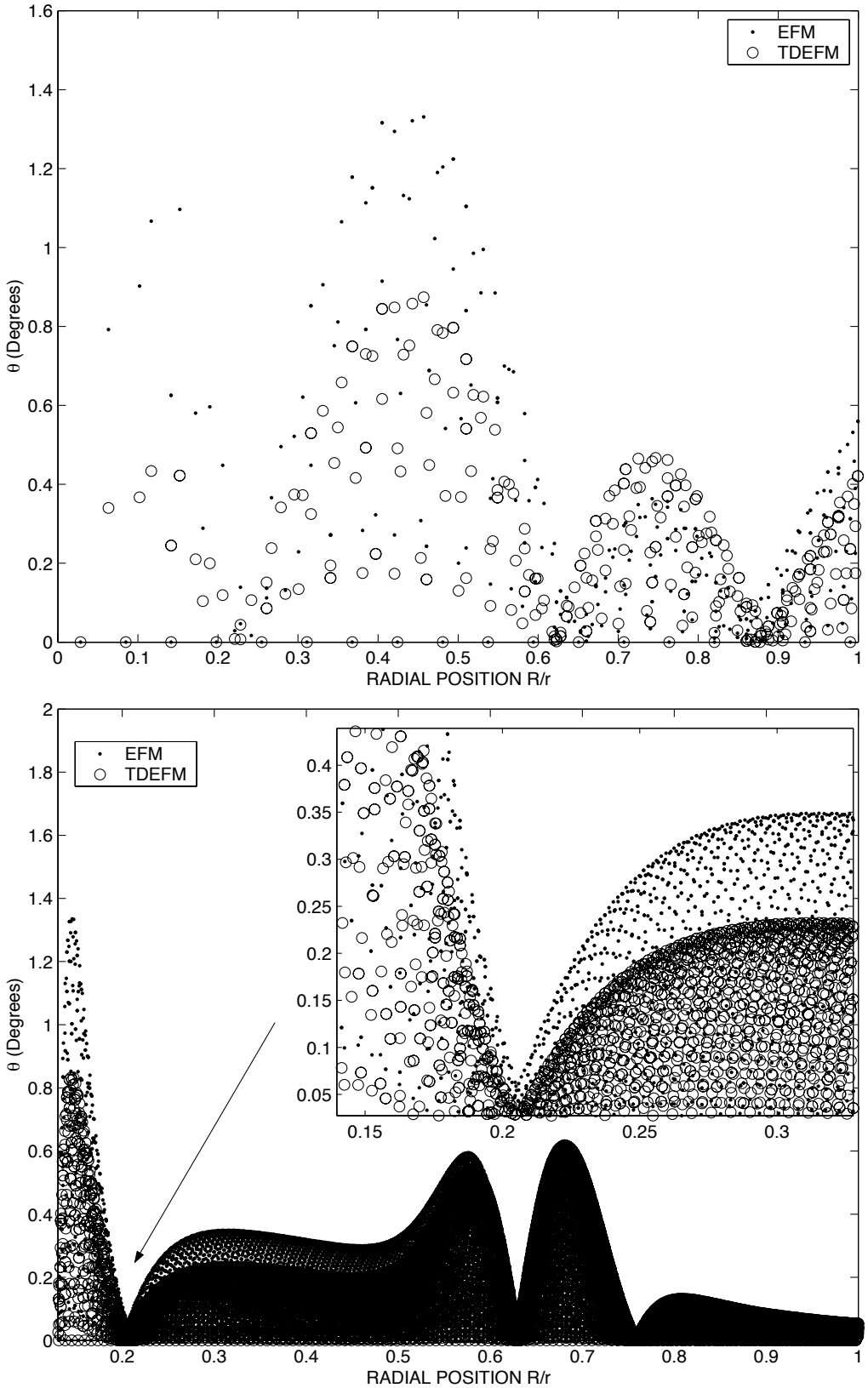

Figure 11.3: Angle of deviation for 2D-TDEFM and 2D-EFM in the implosion problem. (Top) 50x50 cells, (Bottom) 400x400 cells. Simulations are run up to $t\sqrt{RT}/r = 0.098$. Each point represents the angle of deviation (*i.e.* the angular difference between the radial position vector and the velocity vector) for a given cell.

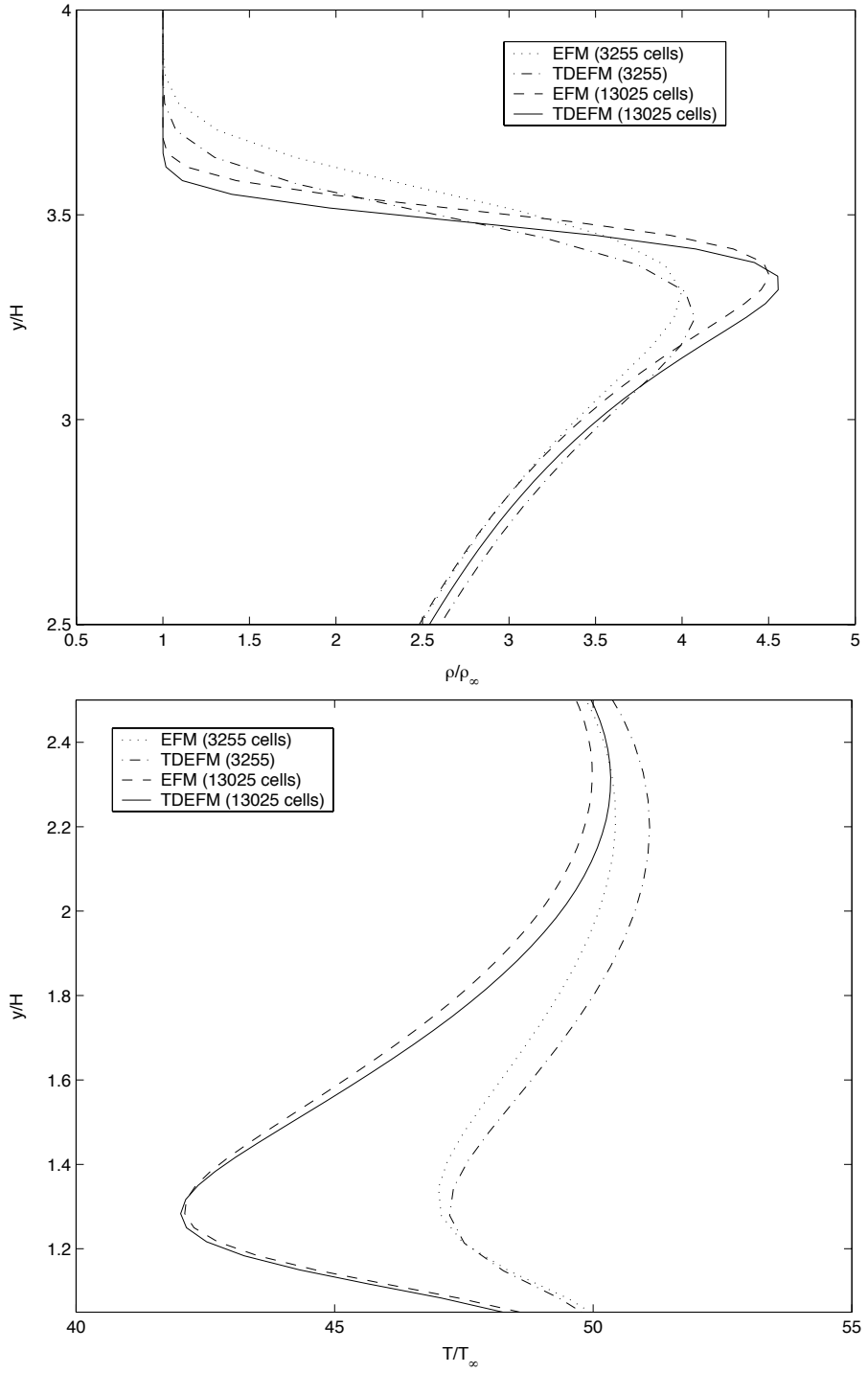

Figure 11.4: Density profiles (top) and temperature profiles (bottom) from EFM and TDEFM solutions using various mesh densities. The results shown are alone line A-A' as shown in Figure 10.16. The temperature profile is in the region ($1 \geq y/H \geq 2.5$). The density profile is in the region ($2.5 \geq y/H \geq 4$). Flow is at Mach 20 with $\gamma = 1.4$. Initial conditions are $\rho = \rho_\infty, M = M_\infty$ and $T = T_\infty$. The simulations are progressed in time to $t\sqrt{RT_\infty}/H = 3$.

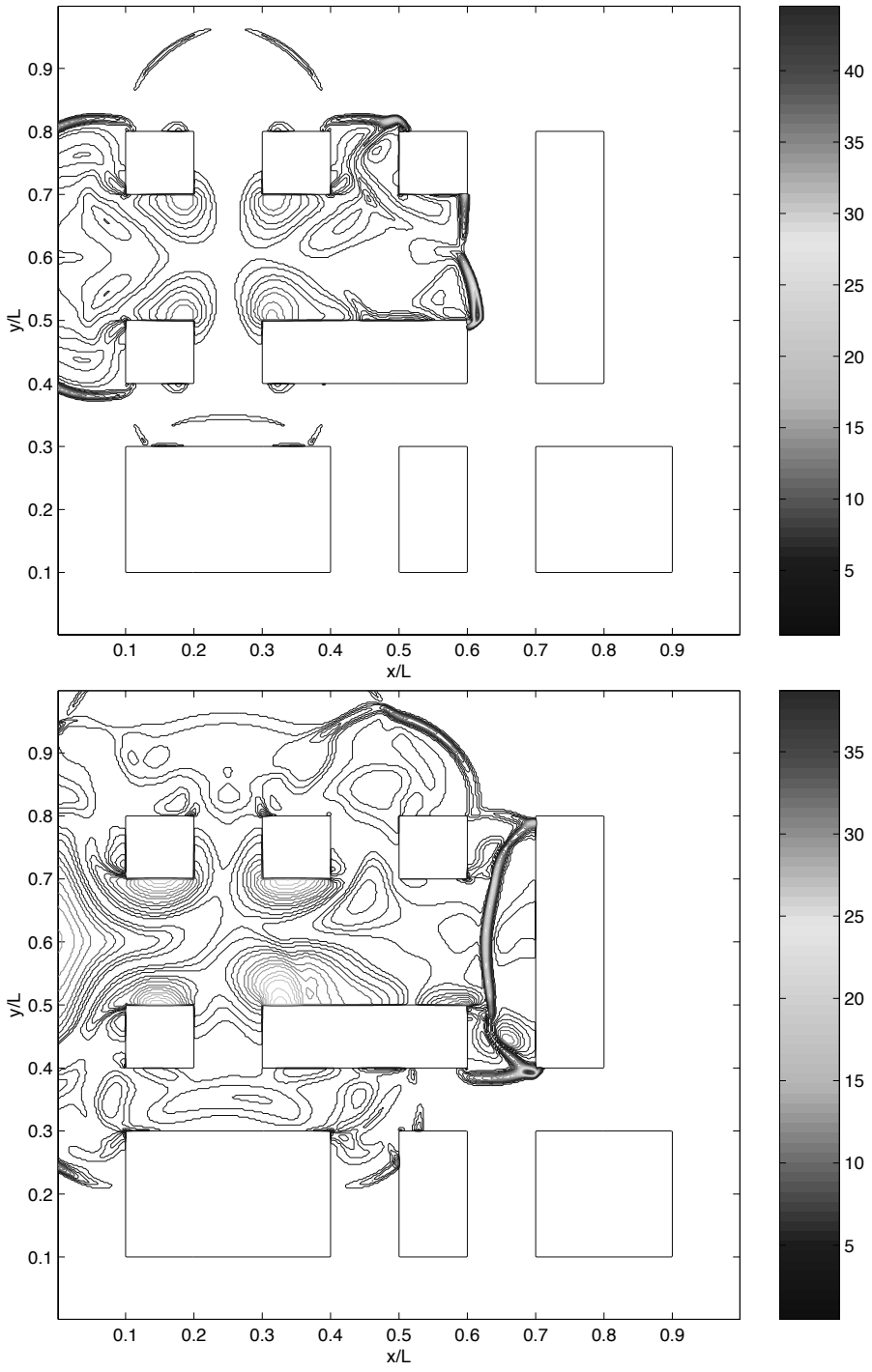

Figure 11.5: Percentage difference in density contours between TDEFM and EFM results at times $t\sqrt{RT}/L = 0.05$ and 0.1 for flow through city buildings. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extrapolated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.46. The computational grid is a cartesian mesh of 400x400 cells.

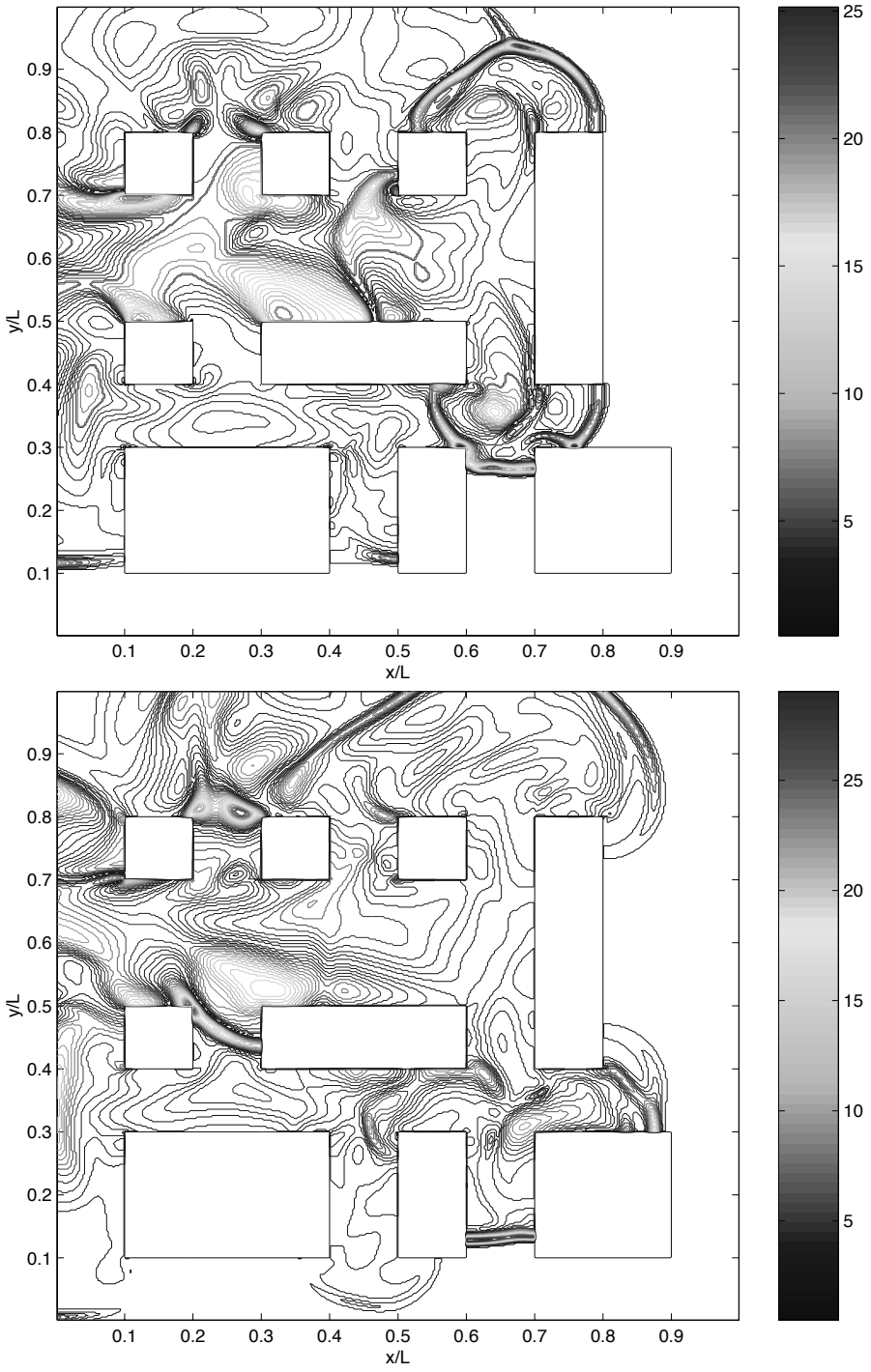

Figure 11.6: Percentage difference in density contours between TDEFM and EFM results at times $t\sqrt{RT}/L = 0.15$ and 0.2 for flow through city buildings. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extrapolated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.46. The computational grid is a cartesian mesh of 400x400 cells.

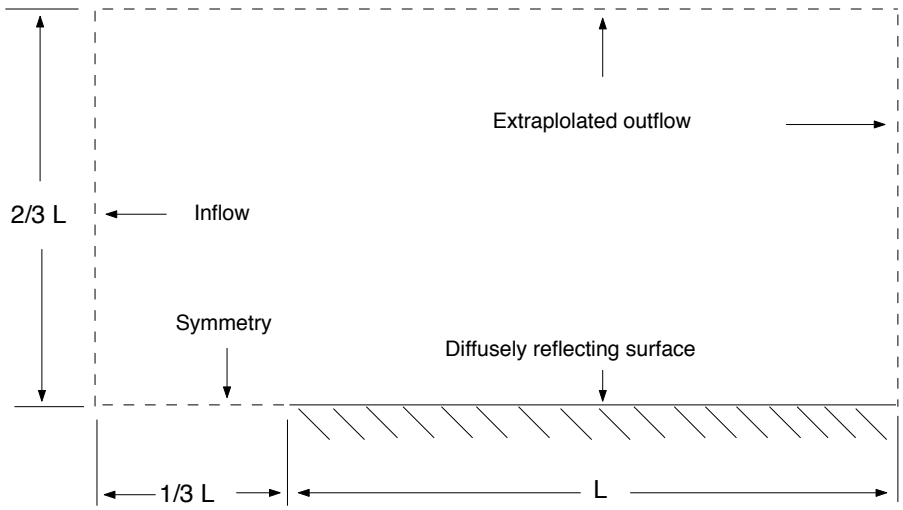

Figure 11.7: Computational domain used for simulation of hypersonic flow over a flat plate. The physical geometry is fixed in both TDEFM and DSMC computations. The knudsen number is varied through manipulation of the gas viscosity alone.

- While TDEFM incorporates true direction fluxes, the order of accuracy is still first order. Therefore, in each time step the fluxes masses are evenly distributed around their destination cells. This forces the solution to take on (to a limited extent) the shape of the computational grid. This flaw is common in every finite volume CFD code.

11.3 Adaptive mesh refinement based on mean free path length

Kinetic theory based continuum solvers such as EPSM and EFM are known to have an effective mean free path length equivilent to the cell size [1, 2, 97]. This is also true in the TDEFM fluxes - fluxes are transported from a source cell to a nearby destination cell, after which they effectively undergo an infinite number of collisions. Therefore, any kinetic theory based continuum solver (including direct simulations in the high collision rate limit) will demonstrate flow features that are reliant upon the cell size. This is often regarded as a negative feature of kinetic theory based solvers with significant effort by various authors [117] to eliminate the dependence of flow features on the cell width.

The use of Adaptive Mesh Refinement (AMR), together with the diffusely reflective flux expressions derived in Section 8.2, allows us the possibility of using this feature to attempt the simulation of a psuedo-viscous gas. Previous finite volume solvers which employed adaptive mesh refinement focused on the resolution of flow features such as shock waves through the addition of large numbers of cells in such regions. However, to more accurately represent the physical flow of gas, the mesh adaptation used by TDEFM focused on setting the local cell size to a fraction of the local mean free path.

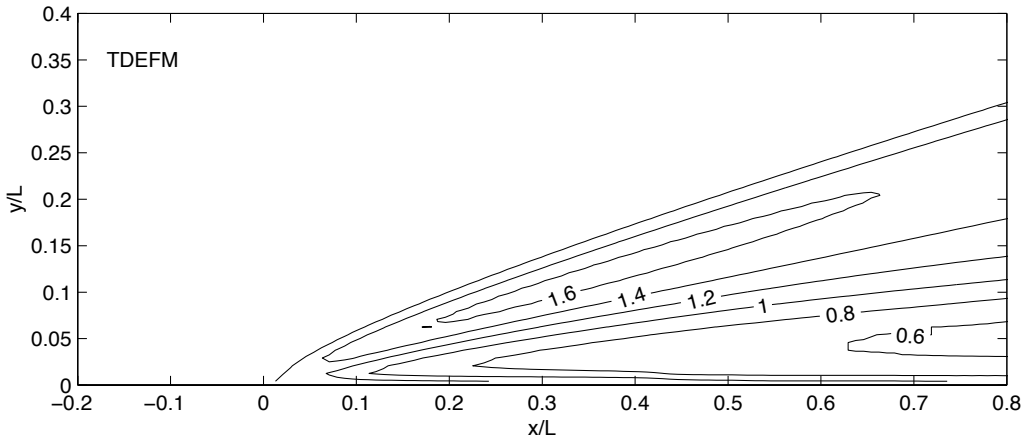

Figure 11.8: Density contours from adaptive TDEFM for hypersonic flow over a flat plate. The freestream Mach numer is $M_\infty = 5$. The Knudsen number is $Kn = 0.002666$. The gas is monatomic with a two coefficient sutherland viscosity law. Adaptive Mesh Refinement (AMR) is employed to ensure the cell size is approximately equal to the local mean free path.

This concept is tested by simulating hypersonic flow over a flat plate while employing adaptive mesh refinement. The initial conditions and geometry are shown in Figure 11.7. The simulated gas is an ideal, monatomic hypothetical gas with a viscosity governed by the two parameter sutherland viscosity law, defined as:

$$\mu(T) = \frac{C_1 T^{3/2}}{T + C_2} \tag{11.2}$$

where C_1 and C_2 are coefficients which differ between gases and T is the temperature. This viscosity law was selected due to its well known properties and common use by popular commercially available CFD packages. The mach number of the freestream gas is $M_\infty = 5$. The temperature of the plate is fixed at the freestream temperature. The top and right hand side boundaries are extrapolated outflow. The left hand side boundary is inflow while the lower surface located infront of the diffusely reflective surface is specularly reflective. The Knudsen number of the flow is varied to test the general capability of the adaptive grid TDEFM technique. This is done through manipulation of the gas viscosity - the physical geometry is fixed in its dimensions.

The results from TDEFM are compared to results taken from a DSMC solution. The number of simulation particles employed varied with the simulated Knuden number. Due to its fast speed and ease of implementation, Macrossan's ν-DSMC was used. Each DSMC simulation was run until the flow was steady, followed by another 10,000 steps for ensemble averaging to assist in the removal of statistical scatter. The inflow and outflow boundaries were managed through the use of 'reservoir' ghost cells. A calculated number of proposed simulation particles were uniformly distributed over each cell and moved through a time step - particles which resulted inside the computational domain were added to the simulation, while particles resulting outside the domain were disregarded.

Figure 11.8 shows density contours as calculated by TDEFM employing Adaptive Mesh Refinement. The initial grid consists of a regular grid of 20x10 cells (not including ghost cells) which was then refined every 500 time steps. The mesh adaptation is initially restricted to cell division. After the flow is steady and no further cell division occurs, complete mesh adaptation is permitted with cells able to combine with neighbours in the process outlined in Section 8.3. During the cell division phase, a cell is split if the ratio of the local mean free path to cell size $\lambda/\Delta x < 0.75$ and the local gradient length Knudsen number (given in Equation 3.2) is larger than 0.005. This ensures cells in the undisturbed freestream are not remeshed. During the cell combination phase, cells are combined if the ratio of local mean free path to cell size $\lambda/\Delta x > 0.75$. The selection of 0.75 ensures that the cells, on average, are on the order of a mean free path.

To compare the results obtained by DSMC and TDEFM, x-velocity and density profiles at regular locations along the plate are examined. Figure 11.9 shows the x-velocity as a function of distance from the plate surface at locations $x/L = 0.06, 0.33, 0.66$ and 0.86 for varying Knudsen numbers. The gradient of velocity at the plate surface calculated by TDEFM closely matchese that obtained by the DSMC results. There is generally very good agreement between the TDEFM and DSMC results over the entire flow field.

Figure 11.10 shows the density as a function of distance from the plate surface at locations $x/L = 0.06, 0.33, 0.66$ and 0.86 for varying Knudsen numbers. The differences between the TDEFM and DSMC results are more obvious - the thickness of the shock is larger in the TDEFM results than in the DSMC results. This is likely because of DSMC's ability to maintain information regarding mass distribution across cells. TDEFM forces uniform mass distribution across each cell at each time step. This feature of finite volume methods has been shown to artificially increase the width of flow features [89] and is usually resolved using higher order methods. The employment of DTDEFM as a flux calculator would see a decrease in the difference between the results. However, further investigation of this concept falls outside the scope of this thesis and this is not investigated here.

Figure 11.11 show the computational grids employed by the TDEFM solutions for the varying Knudsen numbers. The number of cells employed for low Knuden numbers is much larger than for higher numbers - the number of cells employed by each computation grid for each Knudsen number respectively is $N = 836, 4733$ and 14450 cells, not including ghost cells. Despite the large numbers of cells, the computational expense of using TDEFM with adaptive mesh refinement is still significantly less than using ν-DSMC, requiring less than 9 percent of the computation time required by ν-DSMC. It is important to note that the code utilised is a prototype code not developed for computational speed - a statement reinforced by the fact that the flux expressions are the complete (and computationally expensive) TDEFM flux expressions. Also, one of the advantages to using ν-DSMC is the absence of testing for collision pairs - all collisions are assumed equally likely and no testing is required. Macrossan showed that *nu*-DSMC was at least twice as fast as DSMC using variable hard spheres, depending

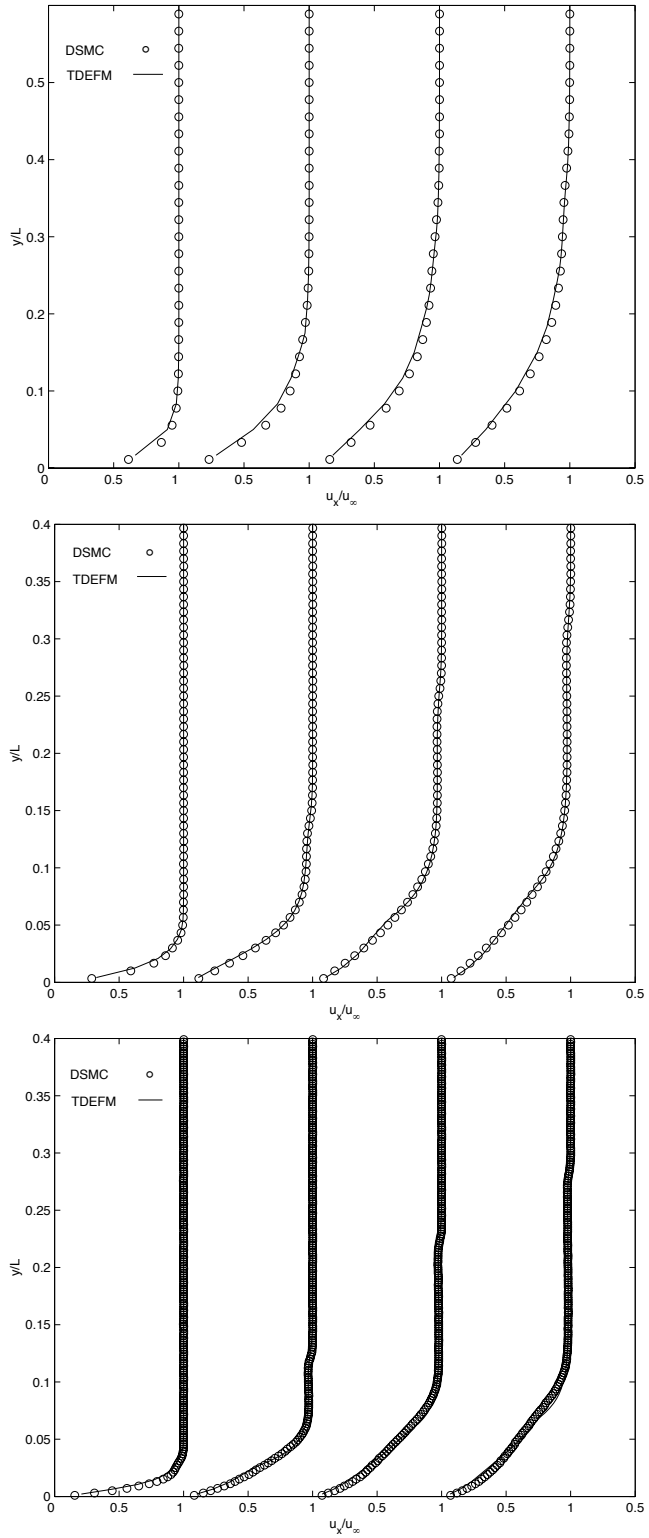

Figure 11.9: X-velocity profiles from TDEFM for simulation of hypersonic flow over a flat plate. Each line represents the variation in x-velocity at $x/L = 0.06, 0.33, 0.66$ and 0.86. The Knudsen numbers for each case are $Kn = 0.0133$ (Top), $Kn = 0.0026$ (Middle) and $Kn = 0.0013$ (Bottom). The gas is monatomic with a two coefficient sutherland viscosity law. Adaptive Mesh Refinement (AMR) is employed to ensure the cell size is approximately equal to the local mean free path.

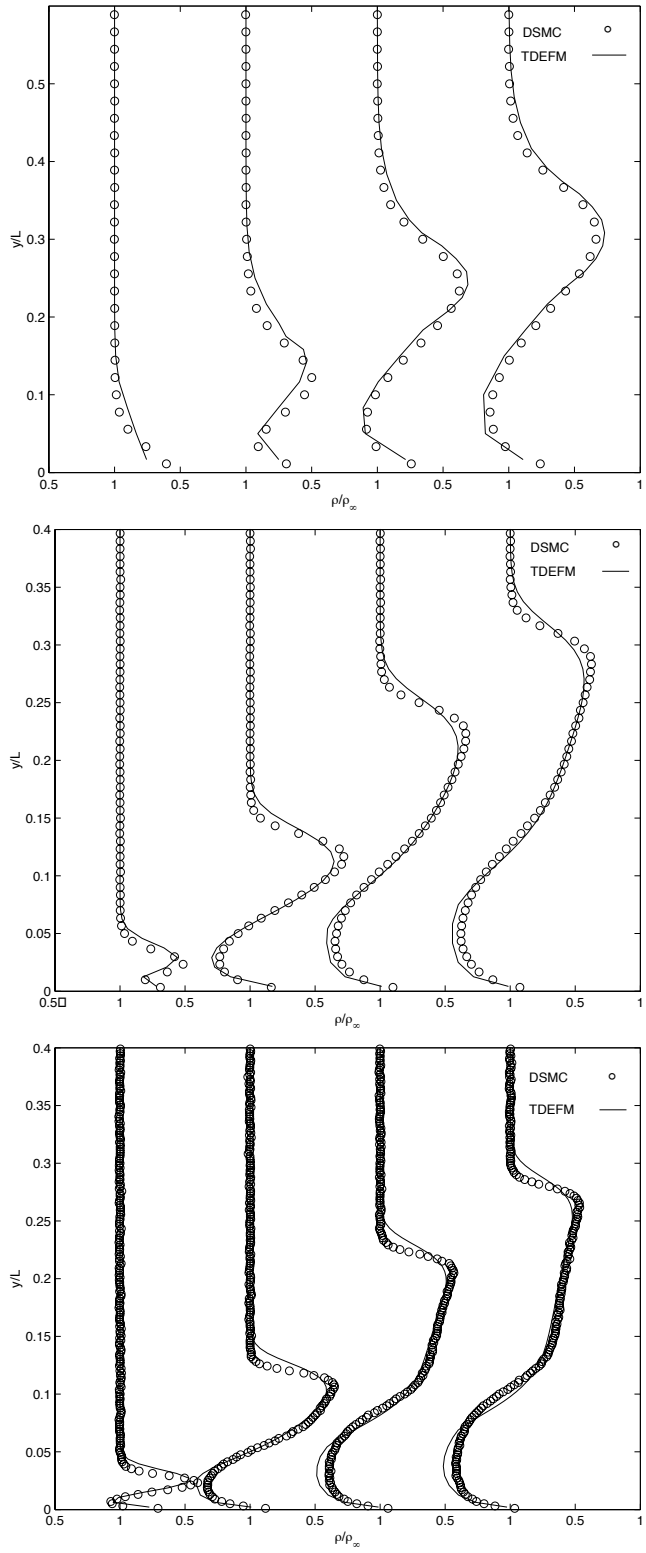

Figure 11.10: Density profiles from TDEFM for simulation of hypersonic flow over a flat plate. Each line represents the variation in density at $x/L = 0.06, 0.33, 0.66$ and 0.86. The Knudsen numbers for each case are $Kn = 0.0133$ (Top), $Kn = 0.0026$ (Middle) and $Kn = 0.0013$ (Bottom). The gas is monatomic with a two coefficient sutherland viscosity law. Adaptive Mesh Refinement (AMR) is employed to ensure the cell size is approximately equal to the local mean free path.

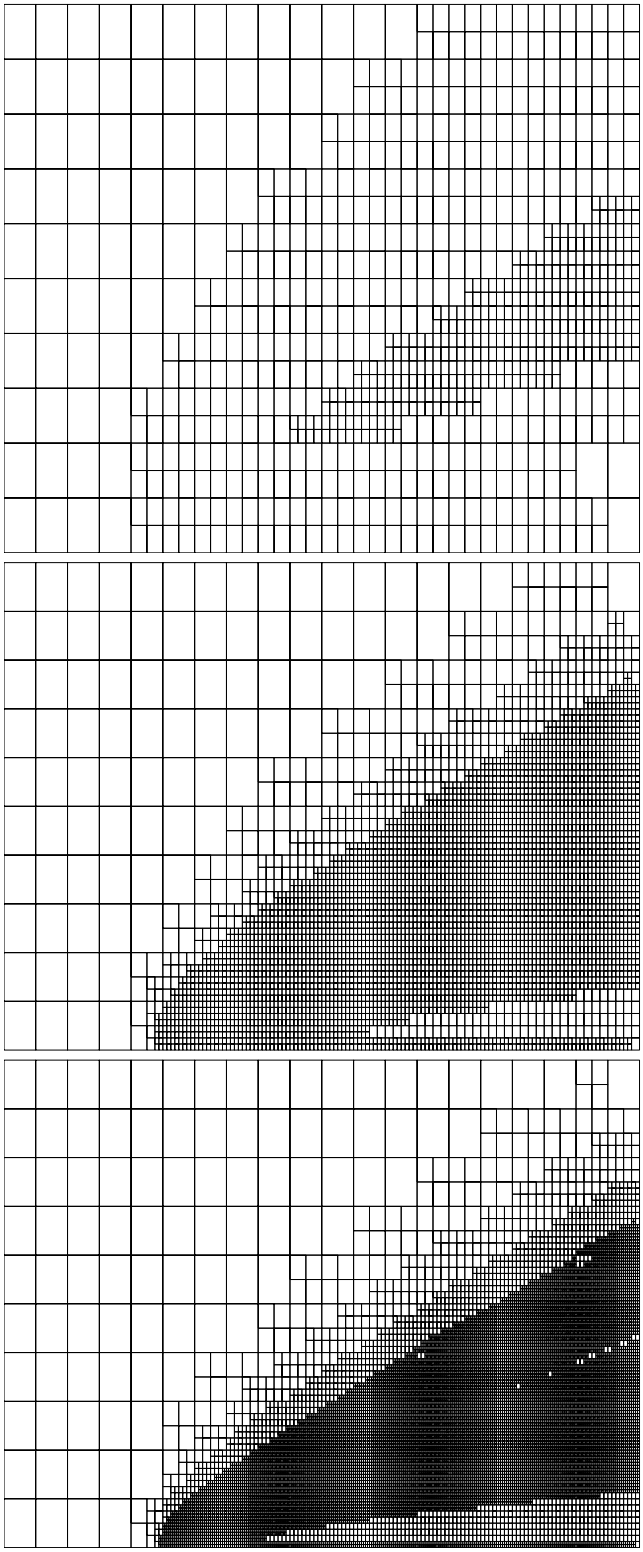

Figure 11.11: Adaptive grids employed by TDEFM for simulation of hypersonic flow over a flat plate. The Knudsen numbers for each case are $Kn = 0.0133$ (Top), $Kn = 0.0026$ (Middle) and $Kn = 0.0013$ (Bottom). The gas is monatomic with a two coefficient sutherland viscosity law. Adaptive Mesh Refinement (AMR) is employed to ensure the cell size is approximately equal to the local mean free path.

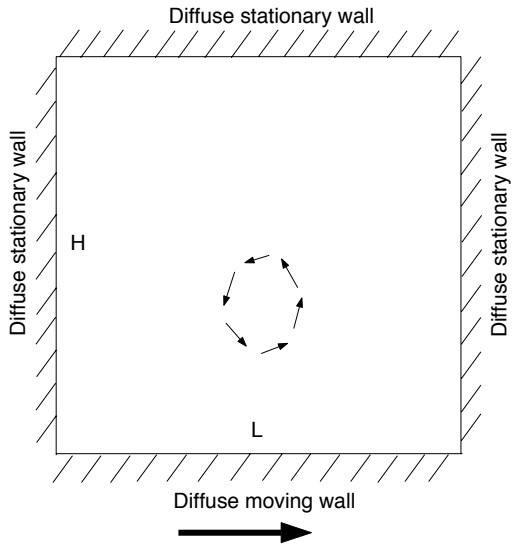

Figure 11.12: Diagram of a high speed viscous flow inside a lid driven cavity ($M_{wall} = 8.73, T_{wall}/T_{init} = 1, Kn = 0.04$ and $\mu = \mu_o(T/T_o)^{0.75}$).

on the ratio of $\Delta t/\tau$ [66]. The presented TDEFM results required approximately 30 minutes (using a Toshiba Satellite M50 with a single 3 GHz processor and 2GB of RAM) while typical modern DSMC codes can require days. It is fair to say the further investigation of this concept is certainly warranted though falls outside the scope of this work.

This concept is further tested though comparison with DSMC results obtained by Wu [113] for a high speed lid driven cavity problem. The results taken from adaptive mesh refinement are also compared to those found through arbitrarily increasing the number of computational cells employed. While previously results were compared by examining the x-velocity and density profiles over the plate, this comparison simply examines the location of the center of circulation. Figure 11.12 shows the computational domain used by both TDEFM and DSMC solutions.

The results from these simulations found that simply increasing the number of cells without consideration of the local mean free path length resulted in an increasingly incorrect solution. Due to the small cell sizes, the gas was effectively being collided very frequently and posessed a decreased effective viscosity. Thus the center of circulation seen in the simulation of a lid driven cavity flow was pulled closer to the wall and futher to the lower right of the simulation region. By setting the cell size splitting criteria such that the cell sizes are, on average, the size of the local mean free path length, a more accurate estimation of the location of the center of circulation can be found. The time step is also restricted to ensure particles do not travel further than a local mean free path, or in this case, the neighbouring cells. It should be noted that due to the simple isotropic splitting of the cells employed here the cell size is not always precisely the target size. This, combined with the fact that at the selected Knudsen number ($Kn = 0.04$) there are insufficient collisions to ensure thermal equilibrium, may be responsible for the differences seen between the location of the center of circulations shown in Table 11.1. The evolution of the computational grid is shown in Figure 11.13.

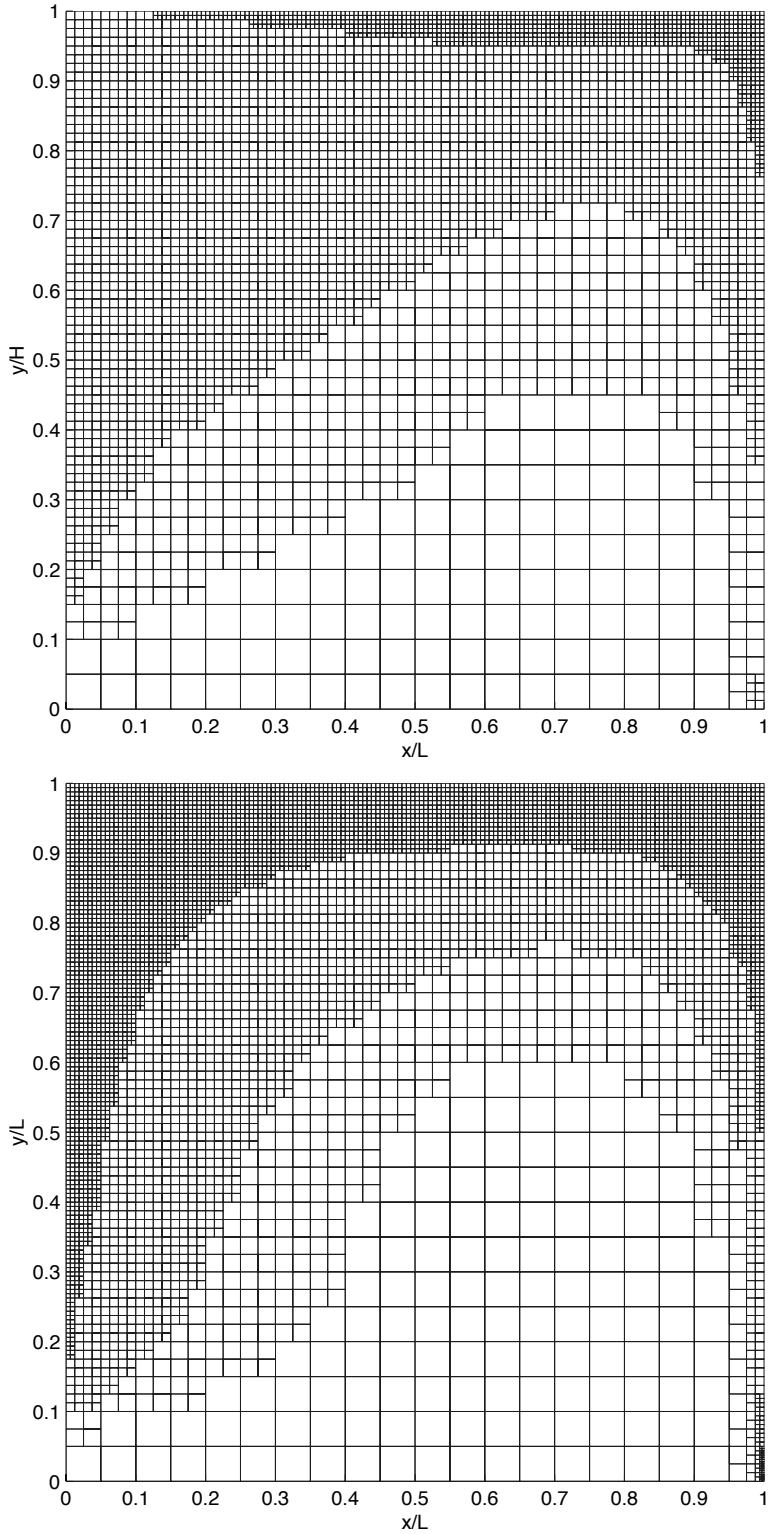

Figure 11.13: Development of the computational mesh for the high speed lid driven cavity flow when the local mean free path is used as a adaptation guide ($K_n = 0.05, M = 8.0, \mu = \mu_o(T/T_o)^{0.75}$).

Simulation	K_n	M	# Cells	Location
Wu [113]	0.04	8.73	2500	(0.67, 0.16)
Coarse Mesh	0.04	8.73	400	(0.6, 0.18)
Fine Mesh	0.04	8.73	6400	(0.77, 0.15)
λ	0.04	8.73	6010	(0.63, 0.16)

Table 11.1: Comparison of the location of the main circulation obtained Wu [113] and Smith (current) in the Lid Driven cavity problem.

Despite these differences there certainly seems to be the possibility of using true direction fluxes together with adaptive mesh refinement to simulate viscous flow. To further test the idea of using TDEFM as a viscous flow solver, many more tests should be run and compared to results obtained from DSMC. The inclusion of en route collisions in the TDEFM flux expressions seems like a sensible starting point and is briefly investigated in Section 11.4.

11.4 En route collisions in TDEFM flux expressions

The derivation of the TDEFM flux expressions is based upon the assumption that flow can be separated into a collision phase and a free movement phase. The TDEFM flux expressions represent the analytical form of Pullin's EPSM fluxes in the presence of an infinite number of simulation particles uniformly distributed across each cell at the beginning of each time step. It is also the analytical form of the PFM (Particle Flux Method) presented by Macrossan *et al.* [70]. The TDEFM flux expressions are also equivalent to Pullin's EFM fluxes in the small CFL number limit [93].

The TDEFM flux expressions are designed around the assumption of local thermal equilibrium, meaning a very large number of collisions are theoretically required for this assumption to be valid. It might therefore be unreasonable to decouple the movement phase and collision phase as the collision times are very small. Past attempts to derive kinetic theory based flux expressions are based on the integration of a distribution function created using the BGK equation and Taylor series expansions [117].

As a result, the effect of collisions and changes in the distribution function in space and time can be included. Knowing that the collision process retards the velocity (on average) by a known amount, we can predict what the effect of collisions will be during their supposed 'free flight phase'. The velocity persistence ratio, ϖ, is defined as the ratio of the initial and final velocity following a collision [20, 45]. Jeans [45] and Chapman [20] determined the mean persistence of velocity ratio varies from $\frac{1}{3}$ to $\frac{1}{2}$. For the range of velocities considered through limitation of the local kinetic CFL number, and following the results from Section 2.3 we assume that ϖ is equal to $\frac{1}{2}$. Since the decay of velocity follows an geometric progression of the persistence ratio, we assume that a particle's velocity (as a function of time) is:

$$v_x(t) = m_x + (v_{x0} - m_x)\varpi^{\left(\frac{t}{2\tau}\right)} \tag{11.3}$$

where τ is the mean collision time and m_x is the bulk velocity of the gas the particle is moving through.The motion of this particle can be predicted through integration:

$$x(t) = x_0 + m_x t - \frac{2\tau\left(\varpi^{\left(\frac{t}{2\tau}\right)} - 1\right)}{\ln(\varpi)}(m_x - v_{x0}) \tag{11.4}$$

with x_0 being the starting position. Therefore, the velocity required to reach a location x^* from x_0 in time step Δt is:

$$v(x^*, x, \Delta t, m_x, \tau_e) = m_x + \left(\frac{(x^* - x_0 - m_x\Delta t)\ln(\varpi)}{2\tau\left[\varpi^{\left(\frac{\Delta t}{2\tau}\right)} - 1\right]}\right) \tag{11.5}$$

Thus, the probability that particles from any location x will land within the region bounded by x_r and x_l in time step Δt is:

$$\begin{aligned} P(x) &= \int_{v(x_l,x,\Delta t,m_x,\tau_e)}^{v(x_r,x,\Delta t,m_x,\tau_e)} g(v_x)dv_x \\ &= \frac{1}{2}\left[\text{erf}\left(\frac{(x_l - x - m_x\Delta t)\ln(\varpi)}{2\sqrt{2}s\tau\left(\varpi^{\left(-\frac{\Delta t}{2\tau}\right)} - 1\right)}\right)\right] \\ &- \frac{1}{2}\left[\text{erf}\left(\frac{(x_r - x - m_x\Delta t)\ln(\varpi)}{2\sqrt{2}s\tau\left(\varpi^{\left(-\frac{\Delta t}{2\tau}\right)} - 1\right)}\right)\right] \end{aligned} \tag{11.6}$$

where $g(v_x)$ is the Maxwell-Boltzmann equilibrium velocity probability distribution function (PDF), $v(x_r, x, \Delta t, m_x, \tau)$ is the velocity required to reach location x_r and $v(x_l, x, \Delta t, m_x, \tau)$ the velocity required to reach location x_l:

$$g(v_x) = \frac{1}{\sqrt{2\pi}s}\exp\left[-\frac{(v_x - m_x)^2}{2s^2}\right] \tag{11.7}$$

$$v(x_r, x, \Delta t, m_x, \tau_e) = m_x + \left(\frac{(x_r - x - m_x\Delta t)\ln(\varpi)}{2\tau\left[\varpi^{\left(\frac{\Delta t}{2\tau}\right)} - 1\right]}\right) \tag{11.8}$$

$$v(x_l, x, \Delta t, m_x, \tau_e) = m_x + \left(\frac{(x_l - x - m_x\Delta t)\ln(\varpi)}{2\tau\left[\varpi^{\left(\frac{\Delta t}{2\tau}\right)} - 1\right]}\right) \tag{11.9}$$

The mean value of $P(x)$ over a region between x_L and x_R represents the fraction of mass f_M to move to between regions x_l and x_r and is:

$$
\begin{aligned}
f_M &= \frac{1}{(x_R - x_L)} \int_{x_L}^{x_R} P(x)dx \\
&= \mathbf{f}_{\mathrm{M}}(\mathrm{m}, \mathrm{s}, \Delta\mathrm{t}, \mathrm{Z}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \\
&= M_c \exp\left(Z^2\left[\frac{(-m\Delta t + x_R - x_l)^2}{2s^2\Delta t^2}\right]\right) + M_1 \mathrm{erf}\left(Z\left[\frac{m\Delta t + x_R - x_l}{\sqrt{2}s\Delta t}\right]\right) \\
- &\ M_c \exp\left(Z^2\left[\frac{(-m\Delta t + x_R - x_r)^2}{2s^2\Delta t^2}\right]\right) - M_2 \mathrm{erf}\left(Z\left[\frac{m\Delta t + x_R - x_r}{\sqrt{2}s\Delta t}\right]\right) \\
- &\ M_c \exp\left(Z^2\left[\frac{(-m\Delta t + x_L - x_l)^2}{2s^2\Delta t^2}\right]\right) - M_3 \mathrm{erf}\left(Z\left[\frac{m\Delta t + x_L - x_l}{\sqrt{2}s\Delta t}\right]\right) \\
+ &\ M_c \exp\left(Z^2\left[\frac{(-m\Delta t + x_L - x_r)^2}{2s^2\Delta t^2}\right]\right) + M_4 \mathrm{erf}\left(Z\left[\frac{m\Delta t + x_L - x_r}{\sqrt{2}s\Delta t}\right]\right) \quad (11.10)
\end{aligned}
$$

where the values of $M_c, M_1 - M_4$ are:

$$
\begin{aligned}
M_c &= \frac{s\Delta t}{Z\sqrt{2\pi}(x_R - x_L)} \\
M_1 &= \frac{1}{2(x_R - x_L)}(m\Delta t - x_l + x_R) \\
M_2 &= \frac{1}{2(x_R - x_L)}(m\Delta t - x_r + x_R) \\
M_3 &= \frac{1}{2(x_R - x_L)}(m\Delta t - x_l + x_L) \\
M_4 &= \frac{1}{2(x_R - x_L)}(m\Delta t - x_r + x_L)
\end{aligned}
$$

and Z is a dimensionless relaxation parameter:

$$
Z = \left(\frac{\Delta t}{\tau}\right) \frac{\ln(\varpi)}{2\left(\varpi^{\left(\frac{\Delta t}{2\tau}\right)} - 1\right)} \quad (11.11)
$$

It is clear that the flux expressions are identical to the previously defined expressions with the exception being the dimensionless parameter Z. This value of Z can also be phrased in terms of an effective collision time τ_e:

$$
Z = \left(\frac{\Delta t}{\tau_e}\right) \frac{\exp\left(\frac{\Delta t}{\tau_e}\right)}{\left(\exp\left(\frac{\Delta t}{\tau_e}\right) - 1\right)} \quad (11.12)
$$

where $\tau_e = -2/\ln(\varpi)\tau$. When the mean collision time τ is very large in comparison to the local time step Δt, the value of Z rapidly approaches 1, and the above expressions approach the original TDEFM expressions. When the local collision time τ is very small compared to the local time step Δt, the value of Z rapidly approaches infinity. The corresponding momentum fluxes are:

$$
\begin{aligned}
f_P &= \mathbf{f}_{\mathrm{P}}(\mathrm{m},\mathrm{s},\Delta\mathrm{t},\mathrm{Z},\mathrm{x_R},\mathrm{x_L},\mathrm{x_l},\mathrm{x_r}) \\
&= P_c \exp\left(Z^2\left[\frac{(-m\Delta t + x_R - x_l)^2}{2s^2\Delta t^2}\right]\right) + P_1 \mathrm{erf}\left(Z\left[\frac{m\Delta t + x_R - x_l}{\sqrt{2}s\Delta t}\right]\right) \\
&- P_c \exp\left(Z^2\left[\frac{(-m\Delta t + x_R - x_r)^2}{2s^2\Delta t^2}\right]\right) - P_2 \mathrm{erf}\left(Z\left[\frac{m\Delta t + x_R - x_r}{\sqrt{2}s\Delta t}\right]\right) \\
&- P_c \exp\left(Z^2\left[\frac{(-m\Delta t + x_L - x_l)^2}{2s^2\Delta t^2}\right]\right) - P_3 \mathrm{erf}\left(Z\left[\frac{m\Delta t + x_L - x_l}{\sqrt{2}s\Delta t}\right]\right) \\
&+ P_c \exp\left(Z^2\left[\frac{(-m\Delta t + x_L - x_r)^2}{2s^2\Delta t^2}\right]\right) + P_4 \mathrm{erf}\left(Z\left[\frac{m\Delta t + x_L - x_r}{\sqrt{2}s\Delta t}\right]\right) \qquad (11.13)
\end{aligned}
$$

where the values of $P_c, P_1 - P_4$ are:

$$
\begin{aligned}
P_c &= \frac{ms\Delta t}{Z\sqrt{2\pi}(x_R - x_L)} \\
P_1 &= \frac{1}{2(x_R - x_L)}\left(m(m\Delta t - x_l + x_R) + \frac{s^2\Delta t}{Z}\right) \\
P_2 &= \frac{1}{2(x_R - x_L)}\left(m(m\Delta t - x_r + x_R) + \frac{s^2\Delta t}{Z}\right) \\
P_3 &= \frac{1}{2(x_R - x_L)}\left(m(m\Delta t - x_l + x_L) + \frac{s^2\Delta t}{Z}\right) \\
P_4 &= \frac{1}{2(x_R - x_L)}\left(m(m\Delta t - x_r + x_L) + \frac{s^2\Delta t}{Z}\right)
\end{aligned}
$$

The energy flux (per unit source mass) is:

$$
\begin{aligned}
f_E &= \mathbf{f}_{\mathrm{E}}(\mathrm{m},\mathrm{s},\Delta\mathrm{t},\mathrm{Z},\mathrm{x_R},\mathrm{x_L},\mathrm{x_l},\mathrm{x_r}) \\
&= E_c \exp\left(Z^2\left[\frac{(-m\Delta t + x_R - x_l)^2}{2s^2\Delta t^2}\right]\right) + E_1 \mathrm{erf}\left(Z\left[\frac{m\Delta t + x_R - x_l}{\sqrt{2}s\Delta t}\right]\right) \\
&- E_c \exp\left(Z^2\left[\frac{(-m\Delta t + x_R - x_r)^2}{2s^2\Delta t^2}\right]\right) - E_2 \mathrm{erf}\left(Z\left[\frac{m\Delta t + x_R - x_r}{\sqrt{2}s\Delta t}\right]\right) \\
&- E_c \exp\left(Z^2\left[\frac{(-m\Delta t + x_L - x_l)^2}{2s^2\Delta t^2}\right]\right) - E_3 \mathrm{erf}\left(Z\left[\frac{m\Delta t + x_L - x_l}{\sqrt{2}s\Delta t}\right]\right) \\
&+ E_c \exp\left(Z^2\left[\frac{(-m\Delta t + x_L - x_r)^2}{2s^2\Delta t^2}\right]\right) + E_4 \mathrm{erf}\left(Z\left[\frac{m\Delta t + x_L - x_r}{\sqrt{2}s\Delta t}\right]\right) \qquad (11.14)
\end{aligned}
$$

where the values of $E_c, E_1 - E_4$ are:

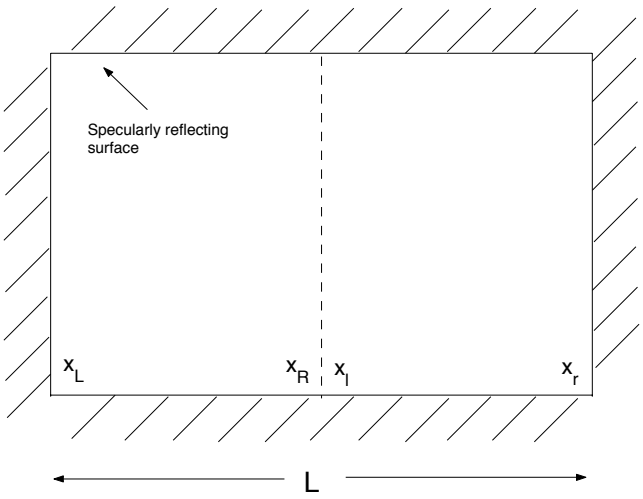

Figure 11.14: Computational domain for simple direct simulation test used to verify CTDEFM flux expressions. A simple box with specularly reflective boundaries and equilibrium conditions is split into two regions. Particles are either moved through a single time step of Δt (for free-flight tests) or moved through a larger number of small time steps Δt_s totalling Δt. Collision pairs are selected from anywhere within the cell since the Knudsen number of the gas is in the cell is larger than one. Due to the identical equilibrium conditions in the box, collisions do not affect the conditions in the cell.

$$
\begin{aligned}
E_c &= \frac{(2C + m^2 + 2s^2)s\Delta t}{2Z\sqrt{2\pi}(x_R - x_L)} \\
E_1 &= \frac{1}{4(x_R - x_L)}\left((m^2 + s^2 + 2C)(m\Delta t - x_l + x_R) + \frac{2ms^2\Delta t}{Z}\right) \\
E_2 &= \frac{1}{4(x_R - x_L)}\left((m^2 + s^2 + 2C)(m\Delta t - x_r + x_R) + \frac{2ms^2\Delta t}{Z}\right) \\
E_3 &= \frac{1}{4(x_R - x_L)}\left((m^2 + s^2 + 2C)(m\Delta t - x_l + x_L) + \frac{2ms^2\Delta t}{Z}\right) \\
E_4 &= \frac{1}{4(x_R - x_L)}\left((m^2 + s^2 + 2C)(m\Delta t - x_r + x_L) + \frac{2ms^2\Delta t}{Z}\right)
\end{aligned}
$$

To verify these fluxes, they were compared to results from a simple DSMC simulation. The computational domain employed by DSMC is shown in Figure 11.14. A box with equilibrium conditions and a Knudsen number of $\frac{2}{\pi}^{-1/2}$ is divided into a source region and a destination region. Particles are either moved through a single time step of Δt (for additional verification of the TDEFM expressions which assume free flight) or a larger number of small time steps of Δt_s. A collision phase follows each movement phase: performing a calculated number of collisions using simple hard spheres. Since the box is in thermal equilibrium, collisions do not affect the conditions (i.e. temperature, viscosity, collision rates). However, these collision do tend to hinder the movement of particles from the source region to the destination region. The flowchart showing the implementation of this test is shown in Figure 11.15. Two different implementations of DSMC were applied - Macrossan's ν-DSMC [66] and μ-DSMC [69]. Both

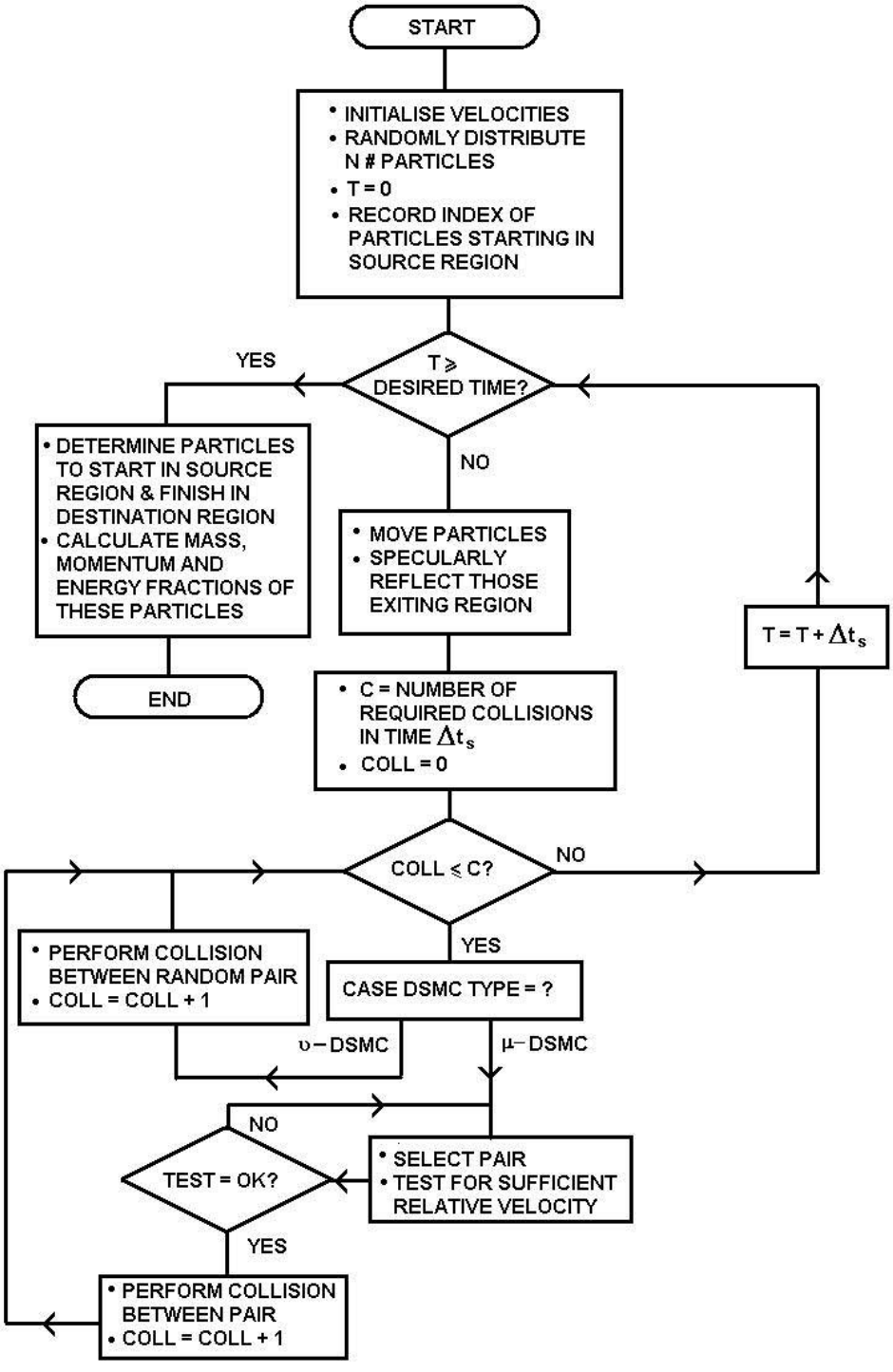

Figure 11.15: Flowchart for the implementation of a simple DSMC verification of the CTDEFM fluxes. Two types of DSMC were employed - Macrossan's ν-DSMC [66] and μ-DSMC [69]. Both solvers employed hard sphere collisions. The differences in the results obtained from each for this simple test were negligible.

solvers employed hard sphere collisions. The differences in the results obtained from each for this simple test were negligible. The results presented here are those obtained using ν-DSMC. The maximum time step Δt was selected such that the kinetic CFL number was unity.

The results obtained from the both free molecular and en route-collision tests are shown in Figure 11.16. As was expected, when particles were permitted to flux a time step Δt without collisions the DSMC and TDEFM results coincide. The mass flux is linear for small time steps, demonstrating that the use of existing kinetic theory methods (where fluxes are linear functions of the time step) is valid in the low time step limit. This mass increases as mass moves into the destination region and then decreases as the 'particles' move past the destination region.

Figure 11.16 also shows the results from DSMC simulations with en route collisions incorporated. It is clear that the results obtained using a single step of Δt in TDEFM, when including the effect of en route collisions, closely match the results obtained using DSMC with multiple time steps of Δt_s.

This method, named Collision TDEFM (CTDEFM), attempts to approximate the effects of collisions on the flight of molecules in an equilibrium gas. However, the presented derivation is flawed because it does not consider the transferal of momentum and energy to the appropriate destination during molecular flight. The velocity used in the calculation of momentum and energy fluxes is the initial pre-collision velocity. Therefore, of the expressions displayed for the momentum and energy flux a fraction will remain in the source cell and the remaining amount will sucessfully be transfered to the destination cell. Therefore, the fluxes of momentum and energy are rederived below to take this into consideration.

Let the fraction of each time step spent by a particle in the source region be represented by f. The velocity of the particle as it crosses the interface between the source and destination region is:

$$v_w(v_x) = m_x + (v_x - m_x)\varpi^{\left(\frac{ft}{2\tau}\right)} \tag{11.15}$$

where v_w is the velocity of the particle at the instant it crosses into the destination region. Assuming that all of the momentum and energy that this particle is carrying after this point in time will be deposited into the destination region, the revised momentum flux is:

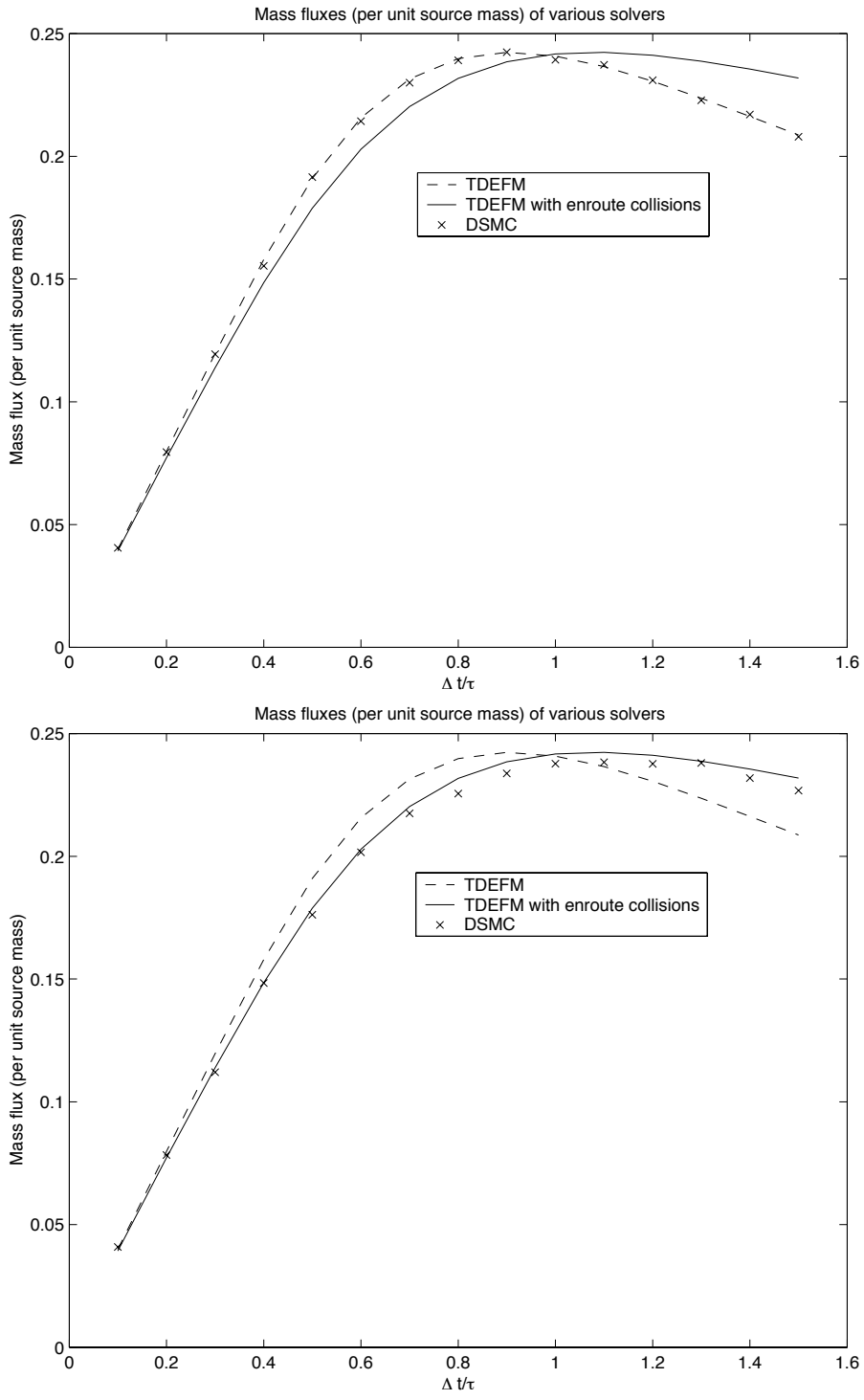

Figure 11.16: [Top] Mass flux (per unit source mass) from TDEFM and DSMC. The DSMC results presented do not incorporate collisions during the free flight phase: simulation particles are moved through a single time step of Δt. [Bottom] Mass flux (per unit source mass) from TDEFM and DSMC. The DSMC results presented incorporate collisions in the flight phase, moving through 10 time steps of Δt_s up to a total time of Δt. The conditions in the cells are not permitted to change as a result of the collisions. Each DSMC simulation used 100,000 simulation particles.

$$
\begin{aligned}
f_P &= \frac{1}{(x_R - x_L)} \int_{x_L}^{x_R} \int_{v(x_l,x,\Delta t,m_x,\tau_e)}^{v(x_r,x,\Delta t,m_x,\tau_e)} v_w(v_x) g(v_x) dv_x dx \\
&= \mathbf{f}_{\mathrm{P}}(\mathrm{m}, \mathrm{s}, \Delta\mathrm{t}, \mathrm{Z}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \\
&= P_c \exp\left(Z^2 \left[\frac{(-m\Delta t + x_R - x_l)^2}{2s^2\Delta t^2}\right]\right) + P_1 \mathrm{erf}\left(Z\left[\frac{m\Delta t + x_R - x_l}{\sqrt{2}s\Delta t}\right]\right) \\
&- P_c \exp\left(Z^2 \left[\frac{(-m\Delta t + x_R - x_r)^2}{2s^2\Delta t^2}\right]\right) - P_2 \mathrm{erf}\left(Z\left[\frac{m\Delta t + x_R - x_r}{\sqrt{2}s\Delta t}\right]\right) \\
&- P_c \exp\left(Z^2 \left[\frac{(-m\Delta t + x_L - x_l)^2}{2s^2\Delta t^2}\right]\right) - P_3 \mathrm{erf}\left(Z\left[\frac{m\Delta t + x_L - x_l}{\sqrt{2}s\Delta t}\right]\right) \\
&+ P_c \exp\left(Z^2 \left[\frac{(-m\Delta t + x_L - x_r)^2}{2s^2\Delta t^2}\right]\right) + P_4 \mathrm{erf}\left(Z\left[\frac{m\Delta t + x_L - x_r}{\sqrt{2}s\Delta t}\right]\right) \qquad (11.16)
\end{aligned}
$$

where the revised values of $P_c, P_1 - P_4$ are:

$$
\begin{aligned}
P_c &= \frac{ms\Delta t}{Z\sqrt{2\pi}(x_R - x_L)} \\
P_1 &= \frac{1}{2(x_R - x_L)} \left(m(m\Delta t - x_l + x_R) + \frac{\varpi^{\frac{f\Delta t}{2\tau}} s^2 \Delta t}{Z}\right) \\
P_2 &= \frac{1}{2(x_R - x_L)} \left(m(m\Delta t - x_r + x_R) + \frac{\varpi^{\frac{f\Delta t}{2\tau}} s^2 \Delta t}{Z}\right) \\
P_3 &= \frac{1}{2(x_R - x_L)} \left(m(m\Delta t - x_l + x_L) + \frac{\varpi^{\frac{f\Delta t}{2\tau}} s^2 \Delta t}{Z}\right) \\
P_4 &= \frac{1}{2(x_R - x_L)} \left(m(m\Delta t - x_r + x_L) + \frac{\varpi^{\frac{f\Delta t}{2\tau}} s^2 \Delta t}{Z}\right)
\end{aligned}
$$

The revised energy flux (per unit source mass) is:

$$
\begin{aligned}
f_E &= \frac{1}{(x_R - x_L)} \int_{x_L}^{x_R} \int_{v(x_l,x,\Delta t,m_x,\tau_e)}^{v(x_r,x,\Delta t,m_x,\tau_e)} (\frac{1}{2} v_w^2 + C) g(v_x) dv_x dx \\
&= \mathbf{f}_{\mathrm{E}}(\mathrm{m}, \mathrm{s}, \Delta\mathrm{t}, \mathrm{Z}, \mathrm{x_R}, \mathrm{x_L}, \mathrm{x_l}, \mathrm{x_r}) \\
&= E_c \exp\left(Z^2 \left[\frac{(-m\Delta t + x_R - x_l)^2}{2s^2\Delta t^2}\right]\right) + E_1 \mathrm{erf}\left(Z\left[\frac{m\Delta t + x_R - x_l}{\sqrt{2}s\Delta t}\right]\right) \\
&- E_c \exp\left(Z^2 \left[\frac{(-m\Delta t + x_R - x_r)^2}{2s^2\Delta t^2}\right]\right) - E_2 \mathrm{erf}\left(Z\left[\frac{m\Delta t + x_R - x_r}{\sqrt{2}s\Delta t}\right]\right) \\
&- E_c \exp\left(Z^2 \left[\frac{(-m\Delta t + x_L - x_l)^2}{2s^2\Delta t^2}\right]\right) - E_3 \mathrm{erf}\left(Z\left[\frac{m\Delta t + x_L - x_l}{\sqrt{2}s\Delta t}\right]\right) \\
&+ E_c \exp\left(Z^2 \left[\frac{(-m\Delta t + x_L - x_r)^2}{2s^2\Delta t^2}\right]\right) + E_4 \mathrm{erf}\left(Z\left[\frac{m\Delta t + x_L - x_r}{\sqrt{2}s\Delta t}\right]\right) \qquad (11.17)
\end{aligned}
$$

where the values of $E_c, E_1 - E_4$ are:

$$
\begin{aligned}
E_c &= \frac{(2C + m^2 + 2\varpi^{\frac{f\Delta t}{2\tau}} s^2) s\Delta t}{2Z\sqrt{2\pi}(x_R - x_L)} \\
E_1 &= \frac{1}{4(x_R - x_L)} \left((m^2 + \varpi^{\frac{f\Delta t}{2\tau}} s^2 + 2C)(m\Delta t - x_l + x_R) + \frac{2\varpi^{\frac{f\Delta t}{2\tau}} m s^2 \Delta t}{Z} \right) \\
E_2 &= \frac{1}{4(x_R - x_L)} \left((m^2 + \varpi^{\frac{f\Delta t}{2\tau}} s^2 + 2C)(m\Delta t - x_r + x_R) + \frac{2\varpi^{\frac{f\Delta t}{2\tau}} m s^2 \Delta t}{Z} \right) \\
E_3 &= \frac{1}{4(x_R - x_L)} \left((m^2 + \varpi^{\frac{f\Delta t}{2\tau}} s^2 + 2C)(m\Delta t - x_l + x_L) + \frac{2\varpi^{\frac{f\Delta t}{2\tau}} m s^2 \Delta t}{Z} \right) \\
E_4 &= \frac{1}{4(x_R - x_L)} \left((m^2 + \varpi^{\frac{f\Delta t}{2\tau}} s^2 + 2C)(m\Delta t - x_r + x_L) + \frac{2\varpi^{\frac{f\Delta t}{2\tau}} m s^2 \Delta t}{Z} \right)
\end{aligned}
$$

When the collision time τ is very large in comparison to the time step Δt, the term $\varpi^{\frac{f\Delta t}{2\tau}}$ and the value of Z both approach unity, therefore reducing the momentum and energy equations to the original TDEFM flux expressions. These flux expressions are compared to those obtained from the simple DSMC test used previously to check the mass flux expressions and are displayed in Figure 11.17. These fluxes are calculated for varying ratios of $\Delta t/\tau$ up to a value of $\Delta t/\tau = 0.5$, which approximately corresponds to a kinetic CFL of unity when the cell size is based on the local mean free path λ. Figure 11.17 show the revised expressions for momentum and energy succesfully reproduce the results obtained through the direct simulation. As expected, the oringinal CTDEFM results show a larger transfer of energy and momentum.

The results presented in Figure 11.17 use an estimated value for the fraction of time spent in the source region of $f = 0.5$. This value is verified using the results from the direct simulation. Figure 11.18 shows the mean fraction of time spent by particles in the source region which sucessfully travel from the source region to the destination region after 30 time steps of t_s. This test shows the value of f averaging just under $f = 0.5$ and is relatively constant over the examined ratios of $\Delta t/\tau$. Due to the complexity of the expressions, no analytical expression exists for the exact value of f. Further efforts to find alternative means of the calculation of f are outside the scope of this work.

11.5 Hybridisation of TDEFM-BGK solver

The assumption of thermal equilibrium is not valid in many modern engineering problems. Therefore, the fluxes obtained by integrating the Maxwell-Boltzmann equilibrium distribution function do not accurately represent the actual fluxes present in regions of thermal non-equilibrium. The traditional and most popular response to this problem is the use of a direct solver such as DSMC. However, despite recent advances, the direct simulation of a large number of simulation particles is computationally expensive when compared to traditional finite

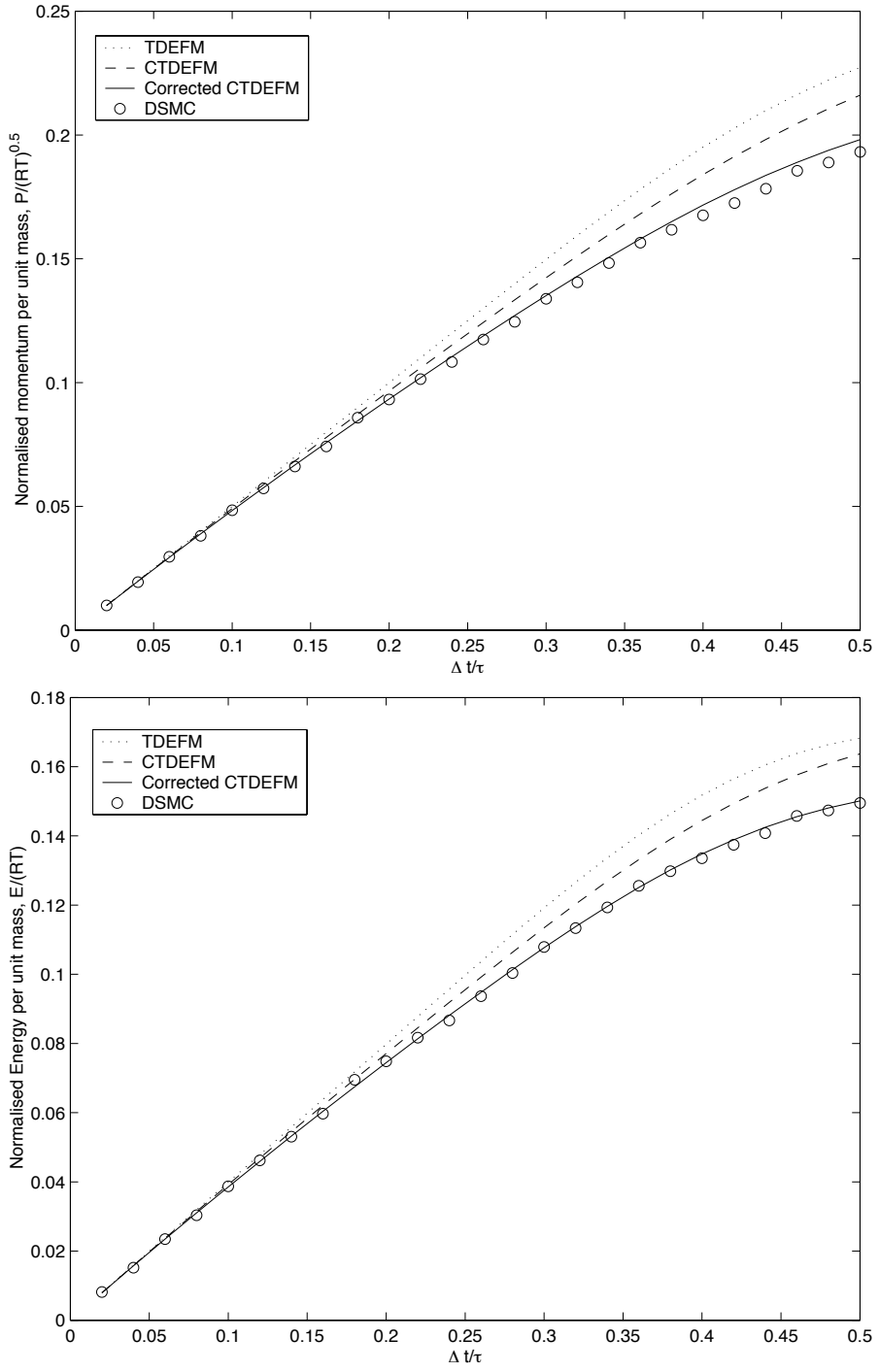

Figure 11.17: [Top] Momentum flux (per unit source mass) from the revised CTDEFM flux expressions and DSMC. [Bottom] Energy flux (per unit source mass) from the revised CTDEFM flux expressions and DSMC. The DSMC results presented incorporate collisions in the flight phase, moving through 30 time steps of Δt_s up to a total time of Δt. Each DSMC simulation used 500,000 simulation particles. The assumed fraction of time spent in the source region by each particle is $f = 0.5$.

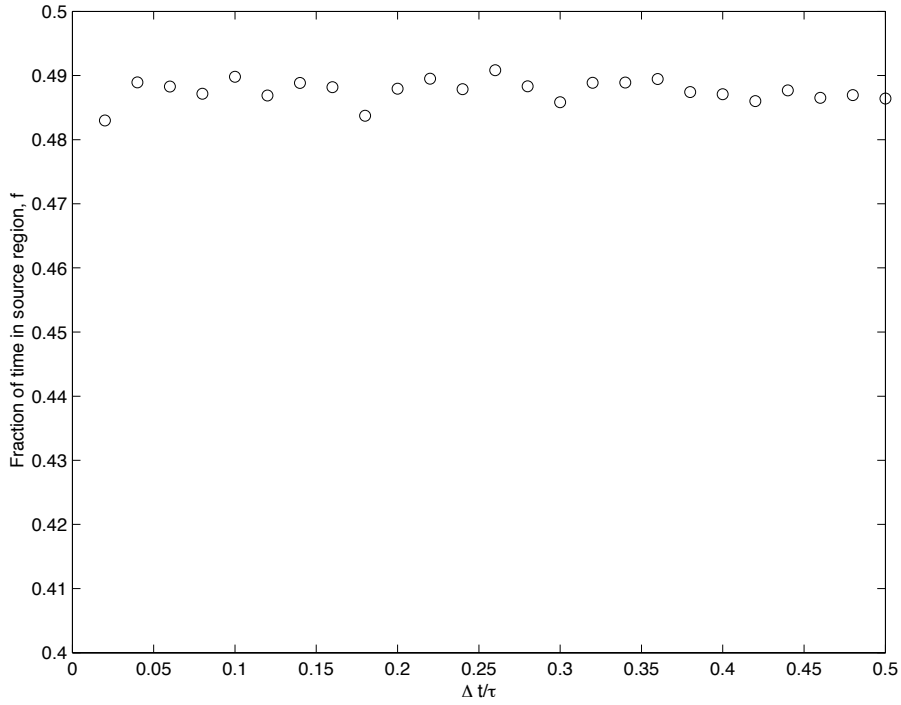

Figure 11.18: Mean fraction of a time step f spent in the source region as calculated by DSMC. Every simulation particle is tracked and the time spent in the source region recorded. Of these, each simulation particle starting in the source region and sucessfully reaching the destination region after 30 time steps of Δt_s is used to calculate the mean fraction f.

volume solvers. In addition, there is a large amount of statistical scatter associated with direct simulation techniques.

Presented are the preliminary results from a proposed hybrid TDEFM-BGK solver similar to that of Kolobov *et. al.* [49, 50]. True direction equilibrium fluxes are employed in regions of thermal equilibrium while a BGK solver is employed in regions of thermal non-equilibrium. To distinguish the regions, the local cell Knudsen number was used where the length scale was based on density gradient length scale as proposed in Equation 3.2. Following previous hybrid solvers [49, 50, 21, 22] where this local Knudsen number was greater than 0.01 the BGK solver was employed. In all other regions, the TDEFM flux solver was used. Both methods are true directional, meaning that fluxes can be transfered from any specified source region to any required destination region. The hybrid solver was implemented by splitting the flux exchange proceedure into the four possible situations:

1. Flux transfer from a TDEFM cell to a TDEFM cell
2. Flux transfer from a TDEFM cell to a BGK cell
3. Flux transfer from a BGK cell to a BGK cell
4. Flux transfer from a BGK cell to a TDEFM cell

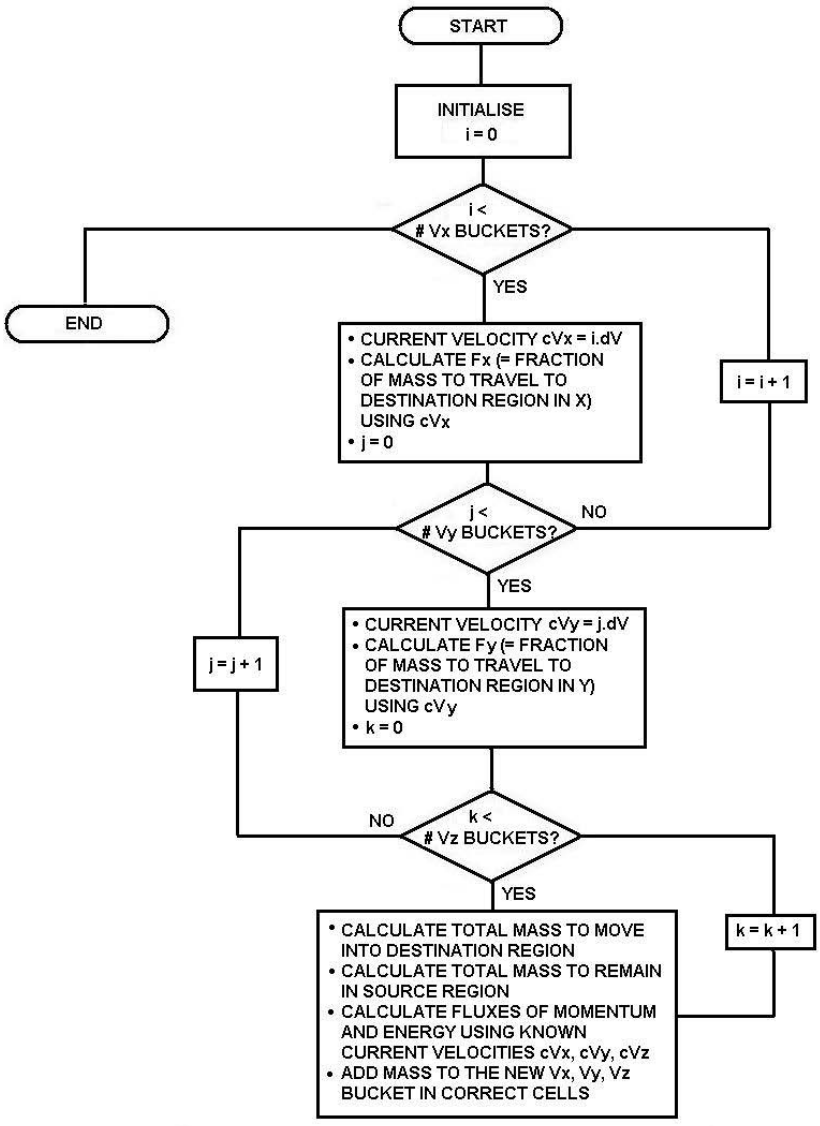

Figure 11.19: Flowchart describing the calculation of fluxes from a BGK source cell to a BGK destination cell.

Simulation	Local cell Kn cutoff	Relative Computational Time (Percent)
Pure BGK	-1	100
Hybrid BGK-TDEFM	0.01	9.85
Pure TDEFM	100	0.25

Table 11.2: Relative computational expense required by the hybrid BGK-TDEFM solver. These times are taken from the 2D implosion test case used in the results presented in Figure 11.20 and Figure 11.21. The BGK solver discretised velocity space in the x, y and z directions into 50 velocity buckets each with velocities ranging from $-6(RT)^{0.5}$ to $6(RT)^{0.5}$.

The simplest of these were the first and the third steps where flux transfer occured between like cells. The flowchart demonstrating how fluxes are calculated between BGK cells is shown in Figure 11.19. When a TDEFM cell was required to calculate the flux of mass, momentum and energy to a BGK cell, a temporary set of velocity buckets were created from the Maxwell-Boltzmann equilibrium distribution function as demonstrated in Figure 5.2 in Section 5.4. These were then used to calculate fluxes in the same way a BGK does. The fluxes from a TDEFM cell to a BGK cell are equivilent to the actual TDEFM fluxes in the presence of an infinitely fine discretisation of velocity space. The fluxes from a BGK cell to a TDEFM cell were calculated by cycling through the discretised velocity distributions (as demonstrated in Figure 11.19) and incrementally adding fluxes of mass, momentum and energy to the TDEFM cells. There is no guarantee these fluxes are the equilibrium fluxes as the BGK cell is capable of tranfering non-equilibrium fluxes into an equilibrium cell.

In addition to the one dimensional results presented in Section 10.1, presented here are two dimensional results for the implosion problem introduced in Section 10.3. The gas is ideal with $\gamma = 5/3$ using a power law viscosity model with a power coefficient of 0.75. The Knudsen number is 0.0125 (based on the length of the square region) and the initial velocity of the flow is zero. The initial conditions are the same as those used in Section 10.3. The BGK solver employed the discretisation of velocities in the three dimensions into 50 discrete velocity 'buckets' each. The time step was set to ensure that a particle moving with a velocity of $(|V| + 6(RT)^{0.5})$ would travel no further than an adjacent cell.

Presented in Figure 11.20 and Figure 11.21 are results from the two dimensional implosion problem. The same characteristic features are present as in the continuum solution, but the sharpness of the flow features is smeared due to the relatively rarefied nature of the flow. Also shown are the regions designated as thermal equilibrium solved using TDEFM and thermal non-equilibrium solved by the BGK solver. It can be seen that as the propagating shock travels inwards and away from the contact discontinuity, the region solved using BGK splits into two distinct parts. The region inbetween the shock and contact discontinuity, in addition to the expansion wave, are not classified as 'broken down' regions by the selected breakdown parameter and are thus solved using TDEFM. It is also seen that both the density colormaps and the region selections are radially symmetric - a feature not seen when using direction coupled fluxes as demonstrated in Section 10. The percentage of cells which employed the

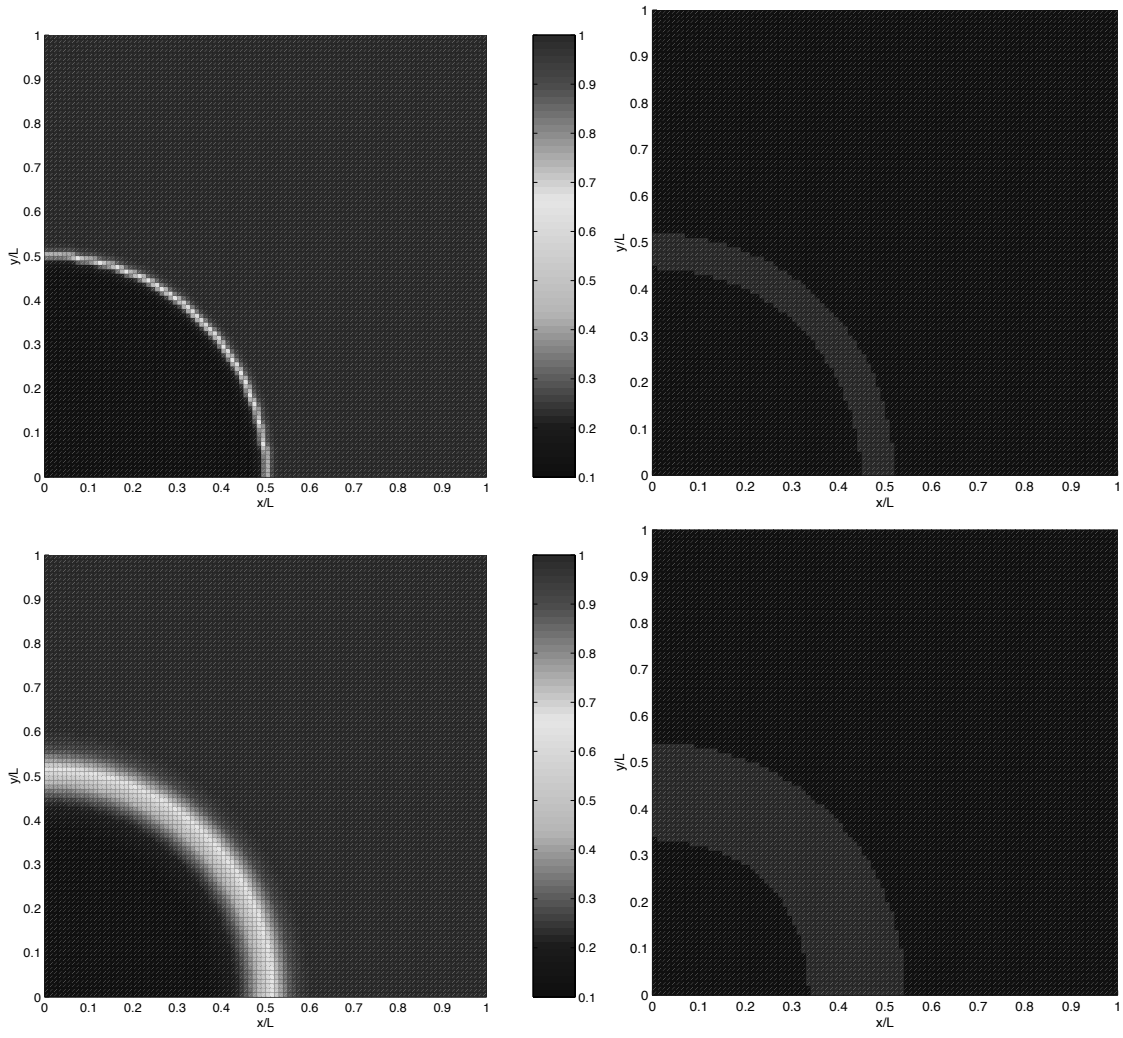

Figure 11.20: Colormaps of density at times $t\sqrt{RT}/L = 0.01$ (Top) and 0.05 (Bottom) for the simulation of a rarefied, viscous flow using a hybrid TDEFM-BGK solver. Also presented are the regions solved by TDEFM (blue) and the BGK solver (red). The Knudsen number of the initial flow (based on the low density region of gas) is 0.0125 (based on the length of the square region). (Continued in Figure 11.21)

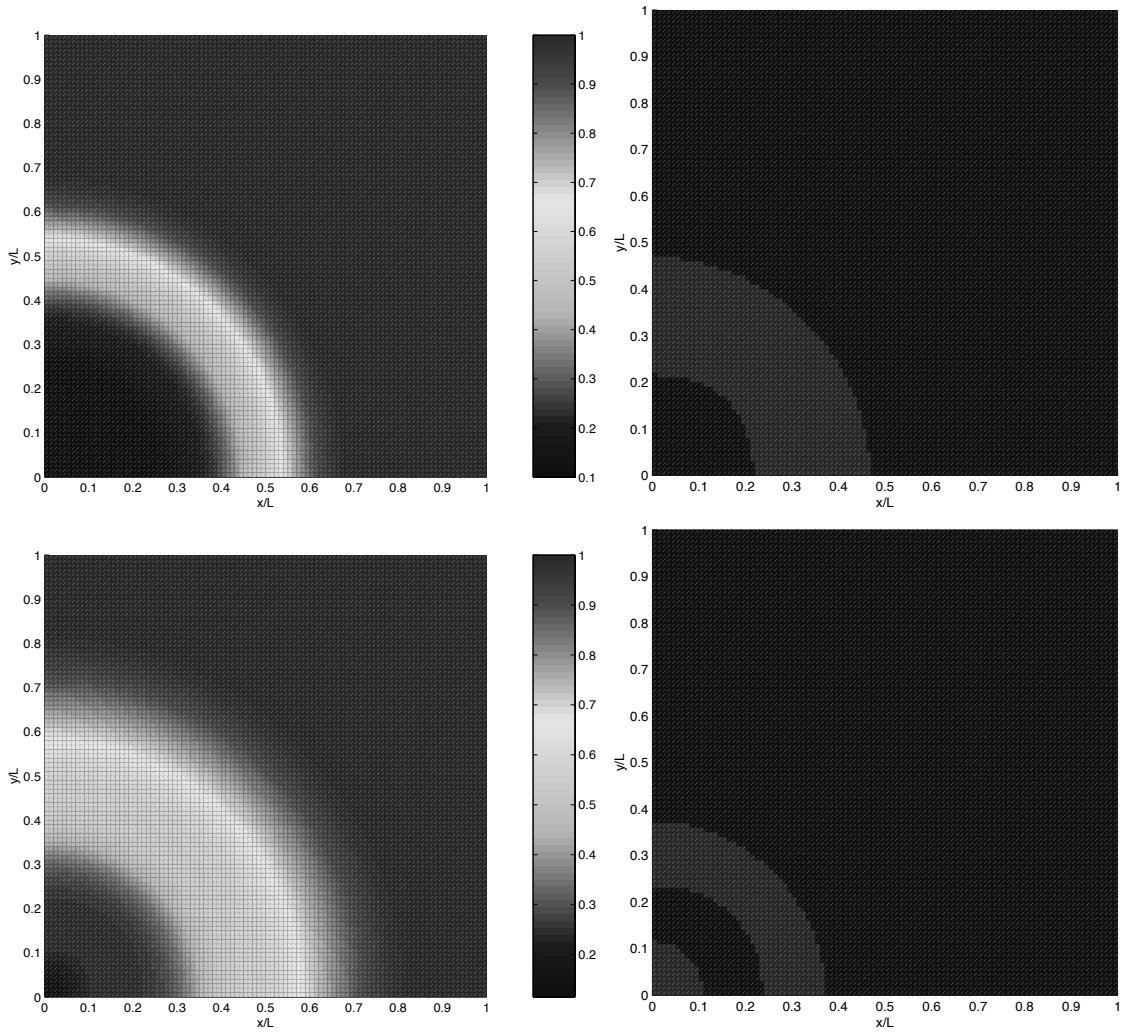

Figure 11.21: Colormaps of density at times $t\sqrt{RT}/L = 0.1$ (Top) and 0.2 (Bottom) for the simulation of a rarefied, viscous flow using a hybrid TDEFM-BGK solver. Also presented are the regions solved by TDEFM (blue) and the BGK solver (red). The Knudsen number of the initial flow (based on the low density region of gas) is 0.0125 (based on the length of the square region).

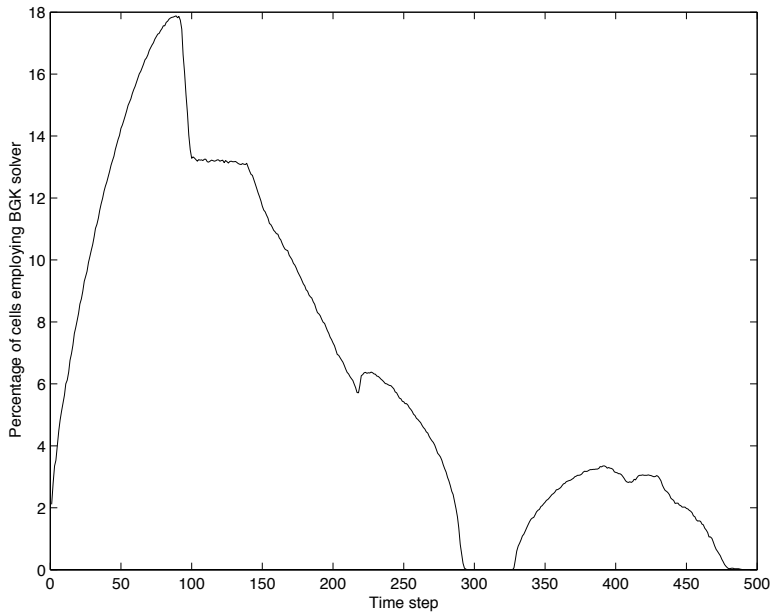

Figure 11.22: Percentage of cells employing the BGK solver in the hybrid BGK-TDEFM solver for the 2D implosion problem. The Knudsen number of the initial flow (based on the low density region of gas) is 0.0125 (based on the length of the square region).

BGK solver in the BGK-TDEFM hybrd as a function of time step is shown in Figure 11.22. The relative computational times required by the various configurations of the hybrid solver are shown in Table 11.2. The use of the current hybrid code as a pure BGK solver is very expensive when compared to the code being used with pure TDEFM or in its hybrid capacity.

Following the work of Kolobov *et. al*, the hybrid TDEFM-BGK is then extended to include an isotropic mesh refinement algorithm. The mesh is refined in regions where the local cell Knudsen number, based on the density gradient length scale, is larger than 0.005. When this is the case, the cell size is refined (or cells are combined) to ensure the cell size is approximately one mean free path length. Presented in Figure 11.23 is an example of the computational grid after several levels of refinement for the implosion problem demonstrated in Figure 11.20 and Figure 11.21.

The results obtained from the one dimensional and two dimensional simulations for the hybrid TDEFM-BGK solver with true directional fluxes seem promising. However, further development of the method past this point falls outside of the scope of this work.

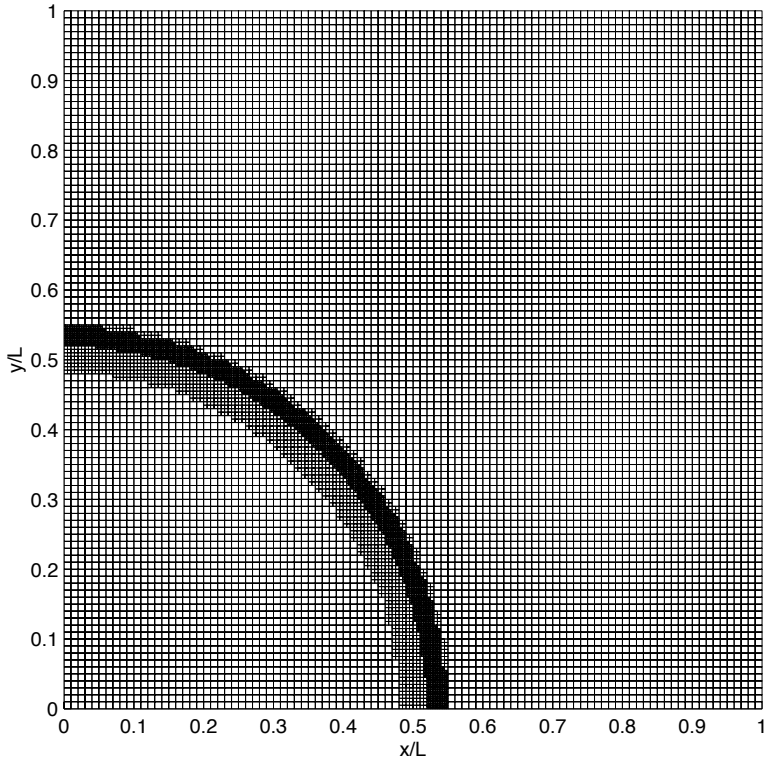

Figure 11.23: The computational mesh obtained using the hybrid BGK-TDEFM solver with isotropic mesh adaptation. The mesh is refined in regions with a local cell Knudsen number greater than 0.005. Regions with a local cell Knudsen number greater than 0.01 employ the BGK solver while other regions use TDEFM. The Knudsen number of the initial flow (based on the low density region of gas) is 0.0125 (based on the length of the square region).

Chapter 12

Conclusion

Since the inception of Computational Fluid Dynamics (CFD) a large number of numerical schemes have been developed. Despite the varied nature of these schemes and their implementations, a majority of these techniques are based around the approximate discretisation of the governing differential equations such as the Euler equations. These methods employ the discretisation of space into finite volumes called 'cells'. In such methods, the standard procedure is to calculate fluxes of mass, momentum and energy and exchange these between cells that share adjacent interfaces through one dimensional flux calculations. This procedure is referred to here as 'direction decoupling'.

An alternate approach is found in the Direct Simulation Monte Carlo (DSMC) method created by Bird. Rather than directly modeling a governing differential equation, the gas is modeled through the use of simulation particles which represent a large number of real particles. These particles are moved regardless of grid orientation or grid density and, for a given time step, can move from a source cell to any destination cell. Therefore, fluxes of mass, momentum and energy are carried by these particles into any cell and are not limited to those that share an adjacent interface.

Direct simulations such as DSMC split the flow into two phases - a collision phase and a movement phase. While the movement phase is handled deterministically, DSMC simulations employ stochastic modeling of the intermolecular collisions. Simulation particles are sorted into cells and perform a calculated number of collisions with other simulation particles. With increasing numbers of collisions the conditions in the cell rapidly approach those of thermal equilibrium. When the gas is in thermal equilibrium, the velocity distribution functions of the simulation particles becomes that of the Maxwell-Boltzmann equilibrium distribution function. Pullin proposed the Equilibrium Particle Simulation Method (EPSM) where, instead of performing a large number of collisions, the velocities of each simulation particle are reselected from the equilibrium distribution function. Since both EPSM and DSMC methods employ the use of a finite number of simulation particles, the results are subject to significant statistical scatter.

While EPSM is faster than performing the large number of collisions required by DSMC, it is still much slower than the existing continuum methods. To counter this, Pullin also proposed the Equilibrium Flux Method (EFM) where integrals of the equilibrium distribution

functions are evaluated over velocity space at cell interfaces. This method does not employ simulation particles like EPSM and DSMC. Instead, fluxes of mass, momentum and energy over cell interfaces are calculated by taking moments of the equilibrium distribution function. Therefore, there is no statistical scatter associated with EFM. However, since the equilibrium distribution functions are only integrated at the cell interfaces exchanges of mass, momentum and energy fluxes are limited to cells sharing an adjacent interface.

The True Direction Equilibrium Flux Method (TDEFM) is presented here with the aim of reproducing the results obtained by a direct simulation technique (such as EPSM). In TDEFM the integrals of the equilibrium distribution function are evaluated over both velocity space and the entire physical space of the cell, rather than just at the boundary. The fluxes of mass, momentum and energy are carried from any specified source region into any specified destination region. These fluxes are not limited to cells sharing adjacent interfaces and can, for a given time step, be exchanged between any source and destination cell. TDEFM is the analytical equivalent to EPSM when conditions in each cell are uniform and an infinite number of simulation particles are present.

The fluxes obtained by TDEFM are shown to be equivalent to that of EFM only when the kinetic CFL number is very small and fluxes are limited to one dimension. When maintaining a small kinetic CFL number and extending to higher dimensions, EFM and TDEFM differ only in that the fluxes calculated by TDEFM are exchanged between all surrounding cells, not just those sharing an adjacent interface. When the kinetic CFL number is very low and destination cells are adjacent, the TDEFM flux expressions can be simplified considerably. Thus, the computational expense associated with extending EFM to its 'direction coupled' equivalent is minimal.

At larger kinetic CFL numbers the results obtained by TDEFM diverge from EFM results and begin to converge on those obtained using EPSM. However, owing to EPSM's ability to maintain gradients of density within each cell through simulation particle location, results obtained by TDEFM do not match those of EPSM. Therefore, linear gradients of density are introduced into the flux expressions. This improvement, named DTDEFM (Density TDEFM), provides results that better match those of EPSM, are free of statistical scatter and require a fraction of the computational expense of that required by EPSM.

The proposed TDEFM method has been shown to provide superior results when compared against selected continuum solvers on structured cartesian computational grids when solving the Euler equations. By utilising an adaptive mesh where the desired cell size is based on a fraction of the local mean free path length and newly derived diffusely reflective flux expressions, the TDEFM fluxes are shown to approximately reproduce results obtained by DSMC for a viscous flow. These preliminary findings require further simulations and comparisons which lie outside the scope of this work. The consideration of viscous forces is then extended through the introduction of en route collisions. The resulting flux expressions are adjusted to take into account the ratio of time step to collision time - when the time step is very small compared to the

collision time, the revised flux expressions simplify to the complete TDEFM flux expressions.

The assumption of thermal equilibrium is invalid for a large number of engineering flows. In order to increase the applicability of TDEFM the method is hybridized with a model-Boltzmann solver. In regions of non-equilibrium the hybrid solver attempts to solve the BGK equation while in regions of thermal equilibrium the conventional TDEFM fluxes are employed. The BGK solver is based on the same true direction principles as TDEFM, allowing flow from a given source region to a given destination region. The BGK solver has been shown to provide identical results to TDEFM when the gas is forced to relax to equilibrium and when sufficiently discretised in velocity space. The proposed hybrid solver is truly a unified flow solver with true directional fluxes.

The simulation of blast waves in city environments by commercially available software is currently unable to be used in a predictive manner due to the complexity involved with the creation of the computational grid and definition of arbitrary geometries. A new program, FASTWAVE, has been presented as a tool for the approximate prediction of blast wave behaviour in city environments. A user is able to define the city geometry, create the computational grid and solve the user defined problem in a matter of minutes on an average desktop PC or laptop. In addition to conventional flux solvers, FASTWAVE also includes several of the true direction flux methods presented here. The current version of FASTWAVE is capable of solving two dimensional flows only. The extension of FASTWAVE to three dimensions lies outside the scope of this work.

Bibliography

[1] Alexander, F.J., Garcia, A.L. and Alder, B.J., 'Cell size dependence of transport coefficients in stochastic particle algorithms', *Physics of Fluids*, **10(6)** : 1540 - 1542, 1998.

[2] Alexander, F.J., Garcia, A.L. and Alder, B.J., 'Erratum: 'Cell size dependence of transport coefficients in stochastic particle algorithms', *Physics of Fluids* 10, 1540 (1998)', *Physics of Fluids*, **12(3)** : 731, 2000.

[3] Almeida, R.C. and Galeao, A.C.,'An adaptive Petrov-Galerkin formulation for the compressible Euler and Navier-Stokes Equations', *Computer methods in applied mechanics and engineering*, **129**:157-176, 1996.

[4] Anderson, J.B., Long, L.N., 'Direct Monte Carlo simulation of chemical reaction systems: prediction of ultrafast detonations', *Journal of Computational Physics*, **118(7)**, 2003.

[5] Anderson, J.D., *Hypersonic and High Temperature Gas Dynamics*, McGraw-Hill, New York, 1989. Reprinted by the AIAA, Reston, VA., 2000.

[6] Anderson, J.D., *Modern Compressible Flow: With historical perspective*, McGraw-Hill, New York, 3rd ed., 2003.

[7] Andries, P., Aoki, K. and Perthame, B., 'A Consistent BGK-Type Model for Gas Mixtures', *Journal of Statistical Physics*, **106(5/6)**: 993-1018, 2002.

[8] Andries, P., Bourgat, J.-F., le Tallec, P. and Perthame, B., 'Numerical comparison between the Boltzmann and ES-BGK models for rarefied gases', *Computer Methods in Applied Mechanics and Engineering*, **191**: 3369-3390, 2002.

[9] Aoki, K., Takata, S. and Nakanishi, T., 'Poiseuille-type flow of a rarefied gas between two parallel plates driven by a uniform external force', *Physical Review E*, **65**, 2002.

[10] Bartel, T.J., Johannes, J.E. and Furlani, T.R., 'Trace chemistry modeling with DSMC in chemically reacting plasmas', *AIAA Paper 98-2753*, AIAA, Washington, 1998.

[11] Bhatnagar, P.L., Gross, E.P. and Krook, M.,'A model for collision processes in gases. I. Small amplitude processes in charged and neutral one-component systems', *Physics Review*, **94(3)** : 511-525, 1954.

[12] Bhatnagar, P. L., 'On the BGK collision model for a two-component assembly', *Zeitschrift für Astrophysik*, **54(4)**: 234-243, 1962.

[13] Bird, G.A.,'Breakdown of Translational and Rotational Equilibrium in Gaseous Expansions', *AIAA Journal*, **8(11)**:1998-2003, 1970.

[14] Bird, G.A., 'Perception of numerical methods in rarefied gas dynamics', *Rarefied gas dynamics: Theoretical and computational techniques. Proceedings of the 16th International Symposium*, edited by E.P. Muntz, P. Weaver and D. Campbell, Progress in Astronautics and Aeronautics **118** : 211-226, AIAA, Washington, 1989.

[15] Bird, G.A., *Molecular Gas Dynamics and the direct simulation of gas flows*, Clarendon Press, Oxford, 1994.

[16] Cebeci, T., Shao, J.P., Kafyeke, F. and Laurendeau, E., *Computational Fluid Dynamics for Engineers*, Horizons Publishing Inc., California, 2005.

[17] Chae, D., Kim, C. and Rho, O.-H., 'Development of an Improved Gas-Kinetic BGK Scheme for Inviscid and Viscous Flows', *Journal of Computational Physics*, **158**: 1-27, 2000.

[18] Chang, C.L.,'Implementation Issues - A Parallel Code Framework Based on the CESE Method', In *Proceedings of the 1st Taiwan-USA workshop on the CESE Method*, **3**:1-56, 2007.

[19] Chapman, S. and Cowling, T.G., *The mathematical theory of non-uniform gases*, Cambridge University PRess, Cambridge, 3rd ed., 1970.

[20] Chapman, S. and Cowling, T.G., *The mathematical theory of non-uniform gases: An account of the kinetic theory of viscosity, thermal conduction and diffusion in gases*, Cambridge University Press, London, 1970.

[21] Chen, W.F., 'Study of Hybrid DSMC/EPSM method of supersonic flat plate', *Act. Aerodynamic Sinica*, **20(4)** : 441-445, 2002.

[22] Chen, W.F., 'Study of hybrid DSMC/EPSM method', *Chinese Journal of Computational Mechanics*, **20(3)** : 274-278, 2003.

[23] Connell, S.D., Holms, D.G., 'Three Dimensional unstructured adaptive multigrid scheme for the Euler Equations', *AIAA Journal* **32**:1626-1632, 1994.

[24] Cook, G., *High Accuracy Capture of Curved Shock Fronts Using the Method of Space-Time Conservation Element and Solution Element*, 37th AIAA (American Institute of Aeronautics and Astronautics) Aerospace Sciences Meeting and Exhibit, 1998.

[25] Dill, K.A. and Bromberg, S., *Molecular Driving Forces: Statistical thermodynamics in Chemistry and Biology*, Garland Science, New York, 2003.

[26] Dukowicz, J.K.,'A general, non-iterative Riemann solver for Godunov's method', *Journal of Computational Physics*, **61(1)**:119-137, 1985.

[27] Garcia, A.L. and Wagner, W.,'Time step truncation error in Direct Simulation Monte Carlo', *Physics of Fluids*, **12(10)** : 2621-2633, 2000.

[28] Godunov, S.K., 'A finite-difference method for the numerical computation and discontinous solutions fo the equations of fluid dynamics', *Matematischeskii Sbornik*, **47**:271-306, 1959.

[29] Godunov, S.K. (Ed.), *Numerical Solution of Multidimensional Problems in Gas Dynamics*, Nauka, Moscow, 1976.

[30] Gottlieb, J.J. and Groth, C.P.T., 'Assessment of Riemann solvers for unsteady one-dimensional inviscid flows of perfect gas', *Journal of Computational Physics*, **78(2)**:437-458. 1988.

[31] Gressier, M. and Moschetta, J.M.,'Robustness versus Accuracy in Shock Wave Computations', *International Journal for numerical methods in Fluids*, **33(3)** : 313-332, 2000.

[32] Gressier, M., Dumbser, M. and Moschetta, J.M.,'A Matrix Stability Analysis of the Carbuncle Phenomenon', *Journal of Computational Physics*, **197(2)** : 647-670, 2004.

[33] Gross, E. P. and Krook, M., 'Model for Collision Processes in Gases: Small-Amplitude Oscillations of Charged Two-Component Systems', *Physical Review*, **102(3)**: 593-604, 1956.

[34] Guoshun, Z.,'Causes and lessons of five explosion accidents', *Journal of Loss Prevention in the Process Industries*, **13**, 2000.

[35] Hadjicontantinou, N.G., 'Analysis of discretization in the direct simulation Monte Carlo', *Physics of Fluids*, **12(10)** : 2634-2638, 2000.

[36] Hadjicontantinou, N.G., 'Time step dependence of transport coefficients in the direct simulation Monte Carlo', *Rarefied Gas Dynamics: Proceedings of the 22nd International Symposium*, edited by T.J. Bartel and M.A. Galles, American Institute of Physics, New York : 381-387, 2001.

[37] Ham F., Lien F.S., Strong A.B., 'A Cartesian grid method with transient anisotropic adaptation', *Journal of Computational Physics*,**179**:469-494, 2002.

[38] Herzberg, G., *Atomic Spectra and Atomic Structure*, Dover, New York, 1944.

[39] Herzberg, G., *Atomic Spectra and Atomic Structure*, Dover, New York, 1963.

[40] Hirsch, C., *Numerical Computation of Internal and External Flows*, Vol. 1, John Wiley and Sons, New York, 1998.

[41] Hischfelder, J.O., Curtiss, C.F., and Bird, R.B., *Molecular theory of gases and liquids*, John Wiley & Sons, New York, 1954.

[42] Holway, L. H., 'Kinetic theory of shock structure using an ellipsoidal distribution function', *in Rarefied Gas Dynamics: Proceedings of the 4th International Symposium*, **1**: 193-215, New York, 1966.

[43] Jacobs, P.A., 'Single Block Navier Stokes Integrator', *NASA ICASE Report 187613*, July 1991.

[44] Jacobs, P.A., *MBCNS: A computer program for the simulation of transient compressible flows*, 1998 Update, Department of Mechanical Engineering Report 7/98, The University of Queensland, June 1998.

[45] Jeans, J., *An introduction to the kinetic theory of gases*, Cambridge University Press, London. First printed 1940, reprinted 1962.

[46] Josyula, E., Xu, K. and Wadsworth, D. C., 'Testing Continuum and Non-Continuum Descriptions in High Speed Flows', *in 24th International Symposium on Rarefied Gas Dynamics*, 10-16th July, Monopoli, Italy, 2004.

[47] Keats, W.A. and Lien, F.S., 'Two dimensional anisotropic cartesian mesh adaptation for the Euler Equations', *International Journal for Numerical Methods in Fluids*, **46**:1099-1125, 2004.

[48] Kim, C. and Jameson, A., 'A Robust and Accurate LED-BGK Solver on Unstructured Adaptive Meshes', *Journal of Computational Physics*, **143**: 598-627, 1998.

[49] Kolobov, V.I., Bayyuk, S.A., Arslanbekov, R.R., Aristov, V.V., Frolova, A.A. and Zabelok, S.A.,'Construction of a unified continuum/kinetic solver for aerodynamic problems', *AIAA Journal of Spacecraft and Rockets*, **42**:598-605, 2005.

[50] Kolobov, V.I., Arslanbekov, R.R., Aristov, V.V., Frolova, A.A. and Zabelok, S.A., 'Unified solver for rarefied and continuum flows with adaptive mesh and algorithm refinement', *Journal of Computational Physics*, **223**:589-608, 2007.

[51] Lebowitz, J. L.,Frisch, H. L. and Helfand, E., 'Nonequilibrium Distribution Functions in a Fluid', *Physics of Fluids*, **3(3)**: 325-338, 1960.

[52] Lengrand, J.C., Raffin, M. and Allegre, J., *Monte Carlo Simulation Method Applied to Jet Wall Interactions under Continuum Flow Conditions*, in Rarefied Gas Dynamics, edited by S.S Fisher, Prog. Astro. Aero. V.74, AIAA, New York, 994-1006, 1981.od Applied to Jet Wall Interactions under Continuum Flow Conditions, in Rarefied Gas Dynamics, editied by S.S Fisher, Prog. Astro. Aero. V.74, AIAA, New York, 994-1006, 1981.

[53] Li, Z.-H. and Zhang, H.-X., 'Numerical investigation from rarefied flow to continuum by solving the Boltzmann model equation', *International Journal for Numerical Methods in Fluids*, **42**: 361-382, 2003.

[54] Li, Z.H. and Zhang, H.X., 'Gas kinetic algorithm using Boltzmann model equation', *Computers and Fluids*, **33**:967-991, 2004.

[55] Li, Z.-H. and Zhang, H.-X., 'Study on gas kinetic unified algorithm for flows from rarefied transition to continuum', *Journal of Computational Physics*, **193**: 708-738, 2004.

[56] Lien F.S., 'A pressure-based unstructured-grid method for all-speed flows', *International Journal for Numerical Methods in Fluids*, **33**:355-374, 2000.

[57] Liever, P.A., Habchi, S.D., Burnell, S.I. and Lingard, J.S., 'Computational Fluid Dynamics Prediciton of the Beagle 2 Aerodynamic Database', *Journal of Spacecraft and Rockets* , **40(5)**:632-638, 2003.

[58] Lilley, C. and Macrossan, M.N., 'Methods for implementing the stream boundary condition in DSMC computations', *International Journal for Numerical Methods in Fluids*, **42**:1363-1371, 2003.

[59] Lilley, C. *A macroscopic chemistry method for the direct simulation of non-equilibrium gas flows*, PhD Thesis, The University of Queensland, Australia, 2005.

[60] Long, L.N. and Sharma, A., 'Numerical simulation of the blast impact problem using the Direct Simulation Monte Carlo (DSMC) method', *Journal of Computational Physics*, **200** : 211-237, 2004.

[61] Luccioni,B.M., Amrosini, R.D. and Danesi, R.F.,'Analysis of building collapse under blast loads', *Engineering Structures*, **26**:63-71, 2004.

[62] Macrossan, M.N., 'The equilibrium flux method for the calculation of flows with non-equilibrium chemical reactions', *Journal of Computational Physics* **80**: 204-231, 1989.

[63] Macrossan, M.N., 'Hypervelocity flow of dissociating nitrogen downstream of a blunt nose', *Journal of Fluid Mechanics* **207**: 167-202, 1990.

[64] Macrossan, M.N. and Pullin, D.I., 'A Computational Investigation of Inviscid Hypervelocity Flow of a Dissociating Gas Past a Cone at Incidence', *Journal of Fluid Mechanics* **266**: 69-92, 1994.

[65] Macrossan, M.N. and Hancock, M., 'A method of assessing the effects of artificial dissipation in Navier Stokes codes', *International Journal for Numerical Methods in Fluids* **22(10)**: 979-985, 1996.

[66] Macrossan, M.N.,'ν-DSMC: A Fast Simulation method for Rarefied Flow', *Journal of Computational Physics*, **173** : 600-619, 2000.

[67] Macrossan, M. N., 'A Particle Simulation Method for the BGK Equation', *in Proceedings of the 22nd International Symposium on Rarefied Gas Dynamics*, American Institute of Physics, Sydney, Australia, **585**: 426-433, 2000.

[68] Macrossan, M.N., 'A particle only hybrid method for near continuum flows', In *AIP Conference Proceedings: 22nd International Symposium on Rarefied Gas Dynamics*, Edited by Bartel and Gallis, **585**:426-433, 2001.

[69] Macrossan, M.N., 'μ-DSMC : A general viscosity method for rarefied flow', *Journal of Computational Physics*, **185**:612-627, 2003.

[70] Macrossan, M.N., Metchnik, M. and Pinto, P.A., 'Hypersonic flow over a wedge with a particle flux method', In Capetilli, Mario, Eds., *24th International Symposium on Rarefied Gas Dynamics*, **762**: 650-656, 10-16th July, 2004.

[71] Macrossan, M.N., Smith, M.R., Metchnik, M. and Pinto, P.A.,'True Direction Equilibrium Flux Method: Applications on Rectangular 2D Meshes', In *Proceedings of the 25th International Symposium on Rarefied Gas Dynamics*, Edited by M.S. Ivanov and A. K. Rebrov, Siberian Branch of the Russian Academey of Sciences : 239-244, 2007. : 239-244, 2007.

[72] Mallett, E.R. and Pullin. D.I. and Macrossan, M.N.,'Numerical study of hpersonic leeward flow over a blunt nosed delta wing',*AIAA Journal* **33**: 1626-1633, 1995.

[73] Merkle, C.L., Behrens, H.W. and Hughes, R.D., 'Application of the Monte-Carlo Simulation Procedure in the Near Continuum Regime', *in Rarefied Gas Dynamics*, edited by S. S. Fisher, Prog. Astro. Aero. v74, AIAA, New York, 1981, pp. 256V268.

[74] Montanero, J.M., Duffy J.W. and Lutsko, J.F., 'Stability of uniform shear flow', *Physics Review E*, **57**:547, 1998.

[75] Nocilla, S., 'Surface Interaction and Applications', In *Rarefied Gas Flows: Theory and Experiment*, Edited by Fiszdon, W., 1981, **3**, New York: Springer-Verlag, 1981.

[76] Ohwada, T. and Fukata, S., 'Simple derivation of high-resolution schemes for compressible flows by kinetic approach', *Journal of Computational Physics*, **211**:424, 2006.

[77] Osher, S. and Solomon, F.,'Upwind difference schemes for hyperbolic systems of conservation laws', *Mathematics of Computation*, **38(158)**:339-375, 1982.

[78] Peery, K.M. and Imlay, S.T. 'Blunt body flow simulations', in *AIAA Paper*, textbf88-2924, 1988.

[79] Peter, W., 'Quiet direct simulation Monte-Carlo with random timesteps', *Journal of Computational Physics*, **221** : 1-8, 2007.

[80] Pullin, D.I., 'Direct Simulation Methods for Compressible Ideal Gas Flow',*Journal of Computational Physics* **34**: 231-244, 1980.

[81] Pullin, D.I. and Moschetta, J.M., 'A Robust Low Diffusive Kinetic Scheme for the Navier-Stokes/Euler Equations', *Journal of Computational Physics*, **133(2)** : 193-204, 1997.

[82] Rausch, R.D., Batina, J.T. and Yang, H.T.Y., 'Spatial Adpaptation procedures on Unstructured Meshes for accurate Unsteady Aerodynamics flow computation', *AIAA Paper 91-1106*.

[83] Igra, O., Falcovitz, J., Reichenbach, H. and W. Heilig, 'Experimental and numerical study of the interaction between a planar shock wave and a square cavity', *Journal of Fluid Mechanics* **313**, 1996.

[84] Remennikov, A.M.,'A review of methods for predicting bomb blast effects on buildings', *Journal of battlefield technology*,**6(3)**:5-10, 2003.

[85] Remennikov, A.M. and Rose, T.A.,'Modelling blast loads on buildings in complex city geometries', *Computers and Structures*, **83**:2197-2205, 2005.

[86] Rigas, F. and Sklavounos, S., 'Experimentally validated 3-D simulation of shock waves generated by dense explosives in confined complex geometries', *Journal of Hazardous Materials*, **A121** : 23-30, 2005.

[87] Roe, P.L., 'Approximate Riemann solversm parameter vectors, and difference schemes', *Journal of Computational Physics*, **43**:357-372, 1981.

[88] Roe, P., 'Characteristic-Based Schemes for the Euler Equations', *Annual Review of Fluid Mechanics*, **1**: 337-365, 1986.

[89] Roe, P., *A Brief Introduction to High-Resolution Schemes*, in 'Upwind and High Resolution Schemes', edited by Hussaini, M.Y., van Leer, B. and van Leer, Rosendale, J.: 9-28, 1997.

[90] Rose, T.A., Smith, P.D. and May, J.H.,'The interaction of oblique blast waves with buildings', *Shock Waves*,**16**:35-44, 2006.

[91] Schwartzentruber, T.E. and Boyd, I.D.,'A hybrid particle-continuum method applied to shock waves', *Journal of Computational Physics*, **215(2)** : 402-416, 2006.

[92] Sirovich, L., 'Kinetic Modeling of Gas Mixtures', *Physics of Fluids*, **5(8)**: 908-918, 1962.

[93] Smith, M.R., Macrossan, M.N. and Abdel-Jawad, M.M, 'Effects of Direction Decoupling in flux calculation in Euler Solvers', *Journal of Computational Physics*, Accepted for publication December 2007, Submitted May 2007.

[94] Smith, M.R., Macrossan, M.N. and Abdel-Jawad, M.M., 'Two Dimensional Isotropic Mesh Adaptation for the Euler Equations using TDEFM', In *Proceedings of the 16th Australasian Fluid Mechanics Conference*, 2-7th December, 2007, Gold Coast Australia.

[95] Smith, M.R., Macrossan, M.N. and Abdel-jawad, M.M., 'Direction coupled equilibrium fluxes and the influence of source gradients on flux expressions', Department of Mechanical Engineering Report, The University of Queensland, December 2006.

[96] Smith, M.R., Macrossan, M.N, Metchnik, M. and Pinto, P.A., 'True Direction Equilibrium Flux Method: Applications on Rectangular 2D Meshes', Department of Mechanical Engineering Report, The University of Queensland, March 2006.

[97] Smith, M.R., *Hybrid methods in near-continuum flows*, M.Phil. Thesis, The University of Queensland, Australia, 2003.

[98] Smith, M.R., Macrossan, M.N., Abdel-jawad, M.M. and Ferguson, A.,'DSMC in the Euler Limit and its approximate Kinetic Theory Fluxes', In *Proceedings of the 14th National Taiwan CFD Conference*, 16-18th August, 2007, Nantou, Taiwan.

[99] Smith, P.D., Whalen, G.P., Feng, L.J. and Rose, T.A.,'Blast loading on buildings from explosions in city streets', *Proceedings of the institution of civil engineers structures and buildings journal*, **146(1)**:47-50, 2001.

[100] Smith, P.D. and Rose, T.A,'Blast wave propagation in city streets - an overview', *Progress in Structural Engineering and Materials*, **8**:16-28, 2006.

[101] Sun M., *Numerical and experimental studies of shock wave interaction with bodies*, Ph.D. Thesis, Tohoku University, 1998.

[102] Sun, Q.H., Boyd, I.D. and Candler, G.V., 'A hybrid continuum/particle approach for modeling subsonic, rarefied gas flows', *Journal of Computational Physics*, **194** : 256-277, 2003.

[103] van den Berg, A.C., Weerheijm, J., 'Blast phenomena in urban tunnel systems',*Journal of Loss Prevention in the Process Industries*, **19**, 2006.

[104] van Leer, B., 'Towards the ultimate conservative differencing scheme IV, A new numerical approach to numerical convection', *Journal of Computational Physics*, **23**:276, 1977.

[105] Van Leer, B., 'Flux-Vector Splitting for the Euler Equations', *Lecture Notes in Physics*, **170**:507-512, 1982.

[106] van Leer, B., *ICASE and the history of high resolution schemes*, in 'Upwind and High Resolution Schemes', edited by Hussaini, M.Y., van Leer, B. and van Leer, Rosendale, J.: 1-7, 1997.

[107] Vincenti, W.G. and Kruger, G.H., *Introduction to physical gas dynamics*, John Wiley & Sons, New York, 1965.

[108] Wang L., and Harvey, J.K., 'The application of adaptive unstructured grid technique to the computation of rarefied hypersonic flows using the DSMC method', in *Rarefied Gas Dynamics*, edited by Harvey, J. and Lord, G., 19th International Symposium: 843-849, 1994.

[109] Welander, P., 'On the temperature jump in a rarefied gas', *Arkiv för Fysik*, **7(44)**: 507-553, 1954.

[110] Wijesinghe, H.S., Hornung, R.D., Garcia, A.L. and Hadjiconstantinou, N.G., 'Three-dimensional Hybrid Continuum-Atomistic Simulations For Multiscale Hydrodynamics', *Journal of Fluids Engineering*, **126** : 768-777, 2004.

[111] Wijesinghe, H.S., Hadjisconstantinou, N.G., 'A discussion of hybrid atomistic-continuum moethods for multiscale hydrodynamics', *International Journal of Multiscale Computational Engineering*, **2**:189, 2004.

[112] Woodward, P. and Collela, P., 'The Numerical simulation of two dimensional fluid flow with strong shocks', *Journal of Computational Physics*, **54**:115-173, 1984.

[113] Wu, J.S., Tseng, K.C. and Kuo, C.H.,'The direct simulation Monte Carlo method using unstructured adaptive mesh and its application', *International Journal for Numerical Methods in Fluids*, **38**:351-375, 2002.

[114] Wu, J.S., Tseng, K.C. and Wu, F.Y., 'Parallel three-dimensional DSMC method using mesh refinement and variable time-step scheme', *Computer Physics Communications*, **162**:166-187, 2004.

[115] Wu, M.G., 'Study on Hybrid EPSM/EPSM method for chemical reacting flow', *Chinese Journal of Computational Mechanics*, **20(3)** : 564-567, 2003.

[116] Wu, Z.N. and Li, K. 'Anisotropic cartesian grid method for steady inviscid shocked flow computation', *International Journal for Numerical Methods in Fluids*, **41**:1053-1084, 2003.

[117] Xu, K., 'A gas-kinetic BGK scheme for the Navier-Stokes equations and its connection with artificial dissipation and Godunov method', *Journal of Computational Physics*, **171**:289-335, 2001.

[118] Xu, K. and Tang, L., 'Nonequilibrium Bhatnagar-Gross-Krook model for nitrogen shock structure', *Physics of Fluids*, **16(10)**, 2004.

[119] Xu, K. and Josyula, E., 'Multiple translational temperature model and its shock structure solution', *Physical Review E*, **71**, 2005.

[120] Xu, K. and Liu, H., 'Multiscale gas-kinetic simulation for continuum and near-continuum flows', *Physical Review E*, **75**, 2007.

[121] Xu, K., Liu, H. and Jiang J., 'Multiple-temperature kinetic model for continuum and near continuum flows', *Physics of Fluids*, **19**, 2007.

Chapter 13

Appendix

13.1 TDEFM coefficients with a uniform mass distribution

Mass coefficients

$$\begin{aligned} M_c &= \frac{s\Delta t}{(x_R - x_L)\sqrt{2\pi}} \\ M_1 &= \frac{1}{2(x_R - x_L)}(m\Delta t - x_l + x_R) \\ M_2 &= \frac{1}{2(x_R - x_L)}(m\Delta t - x_r + x_R) \\ M_3 &= \frac{1}{2(x_R - x_L)}(m\Delta t - x_l + x_L) \\ M_4 &= \frac{1}{2(x_R - x_L)}(m\Delta t - x_r + x_L) \end{aligned}$$

Momentum coefficients

$$\begin{aligned} P_c &= \frac{ms\Delta t}{(x_R - x_L)\sqrt{2\pi}} \\ P_1 &= \frac{1}{2(x_R - x_L)}(m(m\Delta t - x_l + x_R) + s^2\Delta t) \\ P_2 &= \frac{1}{2(x_R - x_L)}(m(m\Delta t - x_r + x_R) + s^2\Delta t) \\ P_3 &= \frac{1}{2(x_R - x_L)}(m(m\Delta t - x_l + x_L) + s^2\Delta t) \\ P_4 &= \frac{1}{2(x_R - x_L)}(m(m\Delta t - x_r + x_L) + s^2\Delta t) \end{aligned}$$

Energy coefficients

$$\begin{aligned} E_c &= \frac{(2C + m^2 + 2s^2)s\Delta t}{2(x_R - x_L)\sqrt{2\pi}} \\ E_1 &= \frac{1}{4(x_R - x_L)}\left((m^2 + s^2 + 2C)(m\Delta t - x_l + x_R) + 2ms^2\Delta t\right) \end{aligned}$$

$$
\begin{aligned}
E_2 &= \frac{1}{4(x_R - x_L)}\left((m^2 + s^2 + 2C)(m\Delta t - x_r + x_R) + 2ms^2\Delta t\right) \\
E_3 &= \frac{1}{4(x_R - x_L)}\left((m^2 + s^2 + 2C)(m\Delta t - x_l + x_L) + 2ms^2\Delta t\right) \\
E_4 &= \frac{1}{4(x_R - x_L)}\left((m^2 + s^2 + 2C)(m\Delta t - x_r + x_L) + 2ms^2\Delta t\right)
\end{aligned}
$$

13.2 TDEFM coefficients with a non-uniform mass distribution

Mass coefficients

$$
\begin{aligned}
M_o &= \int_{x_L}^{x_R}(a + bx)dx = \frac{1}{2}a(x_R^2 - x_L^2) + b(x_R - x_L) \\
M_c &= \frac{s\Delta t}{M_o(x_R - x_L)\sqrt{2\pi}} \\
M_1 &= M_c\left(b + \frac{1}{2}a(-m\Delta t + x_l + x_R)\right) \\
M_2 &= M_c\left(b + \frac{1}{2}a(-m\Delta t + x_r + x_R)\right) \\
M_3 &= M_c\left(b + \frac{1}{2}a(-m\Delta t + x_l + x_L)\right) \\
M_4 &= M_c\left(b + \frac{1}{2}a(-m\Delta t + x_r + x_L)\right) \\
M_5 &= \frac{1}{M_o(x_R - x_L)}\left(\frac{1}{2}b(m\Delta t - x_l + x_R) - \frac{1}{4}a((m^2 + s^2)\Delta t^2 - 2mxl\Delta t + x_l^2 - x_R^2)\right) \\
M_6 &= \frac{1}{M_o(x_R - x_L)}\left(\frac{1}{2}b(m\Delta t - x_r + x_R) - \frac{1}{4}a((m^2 + s^2)\Delta t^2 - 2mxl\Delta t + x_r^2 - x_R^2)\right) \\
M_7 &= \frac{1}{M_o(x_R - x_L)}\left(\frac{1}{2}b(m\Delta t - x_l + x_L) - \frac{1}{4}a((m^2 + s^2)\Delta t^2 - 2mxl\Delta t + x_l^2 - x_L^2)\right) \\
M_8 &= \frac{1}{M_o(x_R - x_L)}\left(\frac{1}{2}b(m\Delta t - x_r + x_L) - \frac{1}{4}a((m^2 + s^2)\Delta t^2 - 2mxl\Delta t + x_r^2 - x_L^2)\right)
\end{aligned}
$$

Momentum coefficients

$$
\begin{aligned}
P_c &= \frac{s\Delta t}{2M_o(x_R - x_L)\sqrt{2\pi}} \\
P_1 &= P_c\left[2bm + a(m(x_l + x_R) - (m^2 + 2s^2)\Delta t)\right] \\
P_2 &= P_c\left[2bm + a(m(x_r + x_R) - (m^2 + 2s^2)\Delta t)\right] \\
P_3 &= P_c\left[2bm + a(m(x_l + x_L) - (m^2 + 2s^2)\Delta t)\right] \\
P_4 &= P_c\left[2bm + a(m(x_r + x_L) - (m^2 + 2s^2)\Delta t)\right] \\
P_5 &= \frac{1}{2M_o(x_R - x_L)}b(m(m\Delta t - x_l + x_R) + s^2\Delta t) \\
&\quad + \frac{1}{4M_o(x_R - x_L)}a[m\Delta t^2(-m^2 - 3s^2) + x_l\Delta t(2m^2 + 2s^2) + m(-x_l^2 + x_R^2)]
\end{aligned}
$$

$$
\begin{aligned}
P_6 &= \frac{1}{2M_o(x_R-x_L)}b(m(m\Delta t - x_r + x_R) + s^2\Delta t) \\
&\quad + \frac{1}{4M_o(x_R-x_L)}a[m\Delta t^2(-m^2-3s^2) + x_r\Delta t(2m^2+2s^2) + m(-x_r^2+x_R^2)] \\
P_7 &= \frac{1}{2M_o(x_R-x_L)}b(m(m\Delta t - x_l + x_L) + s^2\Delta t) \\
&\quad + \frac{1}{4M_o(x_R-x_L)}a[m\Delta t^2(-m^2-3s^2) + x_l\Delta t(2m^2+2s^2) + m(-x_l^2+x_L^2)] \\
P_8 &= \frac{1}{2M_o(x_R-x_L)}b(m(m\Delta t - x_r + x_L) + s^2\Delta t) \\
&\quad + \frac{1}{4M_o(x_R-x_L)}a[m\Delta t^2(-m^2-3s^2) + x_r\Delta t(2m^2+2s^2) + m(-x_r^2+x_L^2)]
\end{aligned}
$$

Energy coefficients

$$
\begin{aligned}
E_c &= \frac{s\Delta t}{4M_o(x_R-x_L)\sqrt{2\pi}} \\
E_1 &= 2E_cb(2C+m^2+2s^2) \\
&\quad + E_ca\left[(2C+m^2+s^2)(-m\Delta t + x_l + x_R) + s^2(x_R - x_l - 3m\Delta t)\right] \\
E_2 &= 2E_cb(2C+m^2+2s^2) \\
&\quad + E_ca\left[(2C+m^2+s^2)(-m\Delta t + x_r + x_R) + s^2(x_R - x_r - 3m\Delta t)\right] \\
E_3 &= 2E_cb(2C+m^2+2s^2) \\
&\quad + E_ca\left[(2C+m^2+s^2)(-m\Delta t + x_l + x_L) + s^2(x_L - x_l - 3m\Delta t)\right] \\
E_4 &= 2E_cb(2C+m^2+2s^2) \\
&\quad + E_ca\left[(2C+m^2+s^2)(-m\Delta t + x_r + x_L) + s^2(x_L - x_r - 3m\Delta t)\right] \\
E_5 &= \frac{b}{4M_o(x_R-x_L)}((m^2+s^2+2C)(m\Delta t - x_l + x_R) + 2m\Delta t s^2) \\
&\quad + \frac{a}{8M_o(x_R-x_L)}[-\Delta t^2(m^2(m^2+6s^2) + 2C(m^2+s^2) + 3s^4) \\
&\quad + mx_l\Delta t(2m^2+6s^2+4C) - (x_l^2 - x_R^2)(s^2+m^2+2C)] \\
E_6 &= \frac{b}{4M_o(x_R-x_L)}((m^2+s^2+2C)(m\Delta t - x_r + xR) + 2m\Delta t s^2) \\
&\quad + \frac{a}{8M_o(x_R-x_L)}[-\Delta t^2(m^2(m^2+6s^2) + 2C(m^2+s^2) + 3s^4) \\
&\quad + mx_r\Delta t(2m^2+6s^2+4C) - (x_r^2 - x_R^2)(s^2+m^2+2C)] \\
E_7 &= \frac{b}{4M_o(x_R-x_L)}((m^2+s^2+2C)(m\Delta t - xl + xL) + 2m\Delta t s^2) \\
&\quad + \frac{a}{8M_o(x_R-x_L)}[-\Delta t^2(m^2(m^2+6s^2) + 2C(m^2+s^2) + 3s^4) \\
&\quad + mx_l\Delta t(2m^2+6s^2+4C) - (x_l^2 - x_L^2)(s^2+m^2+2C)] \\
E_8 &= \frac{b}{4M_o(x_R-x_L)}((m^2+s^2+2C)(m\Delta t - xr + xL) + 2m\Delta t s^2) \\
&\quad + \frac{a}{8M_o(x_R-x_L)}[-\Delta t^2(m^2(m^2+6s^2) + 2C(m^2+s^2) + 3s^4)
\end{aligned}
$$

$$+mx_r\Delta t(2m^2+6s^2+4C)-(x_r^2-x_L^2)(s^2+m^2+2C)]$$

13.3 TDEFM coefficients with a non-uniform velocity distribution function

Mass coefficients

$$\begin{aligned}
M_c &= \frac{s\Delta t}{\sqrt{2\pi}(a\Delta tt+1)(x_R-x_L)}\\
M_1 &= \frac{((a\Delta t+1)x_R+b\Delta t-x_l)}{2(a\Delta t+1)(x_R-x_L)}\\
M_2 &= \frac{((a\Delta t+1)x_R+b\Delta t-x_r)}{2(a\Delta t+1)(x_R-x_L)}\\
M_3 &= \frac{((a\Delta t+1)x_L+b\Delta t-x_l)}{2(a\Delta t+1)(x_R-x_L)}\\
M_4 &= \frac{((a\Delta t+1)x_L+b\Delta t-x_r)}{2(a\Delta t+1)(x_R-x_L)}
\end{aligned}$$

Momentum coefficients

$$\begin{aligned}
P_c &= \frac{s\Delta t}{2\sqrt{2\pi}(a\Delta t+1)^2(x_R-x_L)}\\
P_1 &= P_c\left[a\left(x_R(a\Delta t+1)+b\Delta t+x_l\right)+2b\right]\\
P_2 &= P_c\left[a\left(x_R(a\Delta t+1)+b\Delta t+x_r\right)+2b\right]\\
P_3 &= P_c\left[a\left(x_L(a\Delta t+1)+b\Delta t+x_l\right)+2b\right]\\
P_4 &= P_c\left[a\left(x_L(a\Delta t+1)+b\Delta t+x_r\right)+2b\right]\\
P_5 &= \frac{a((x_R(a\Delta t+1))^2-x_l^2)+(b^2+s^2)(2+a\Delta t)\Delta t+2b(x_R(a\Delta t+1)^2-x_l)}{4(a\Delta t+1)^2(x_R-x_L)}\\
P_6 &= \frac{a((x_R(a\Delta t+1))^2-x_r^2)+(b^2+s^2)(2+a\Delta t)\Delta t+2b(x_R(a\Delta t+1)^2-x_r)}{4(a\Delta t+1)^2(x_R-x_L)}\\
P_7 &= \frac{a((x_L(a\Delta t+1))^2-x_l^2)+(b^2+s^2)(2+a\Delta t)\Delta t+2b(x_L(a\Delta t+1)^2-x_l)}{4(a\Delta t+1)^2(x_R-x_L)}\\
P_8 &= \frac{a((x_L(a\Delta t+1))^2-x_r^2)+(b^2+s^2)(2+a\Delta t)\Delta t+2b(x_R(a\Delta t+1)^2-x_r)}{4(a\Delta t+1)^2(x_R-x_L)}
\end{aligned} \tag{13.1}$$

Energy coefficients

$$\begin{aligned}
E_c &= \frac{s\Delta t}{6\sqrt{2\pi}(a\Delta t+1)^3(x_R-x_L)}\\
E_1 &= E_c\left[(2s^2+b^2)(3+3a\Delta t+a^2\Delta t^2)\right.\\
&\quad +6C(a\Delta t+1)^2+a^2x_R(a\Delta t+1)(x_l+x_R(a\Delta t+1))
\end{aligned}$$

$$
\begin{aligned}
& +a^2x_l^2+ab((3+5a\Delta t+2a^2\Delta t^2)x_R+(3+a\Delta t)x_l)\big]\\
E_2 &= E_c\big[(2s^2+b^2)(3+3a\Delta t+a^2\Delta t^2)\\
& +6C(a\Delta t+1)^2+a^2x_R(a\Delta t+1)(x_r+x_R(a\Delta t+1))\\
& +a^2x_r^2+ab((3+5a\Delta t+2a^2\Delta t^2)x_R+(3+a\Delta t)x_r)\big]\\
E_3 &= E_c\big[(2s^2+b^2)(3+3a\Delta t+a^2\Delta t^2)\\
& +6C(a\Delta t+1)^2+a^2x_L(a\Delta t+1)(x_l+x_L(a\Delta t+1))\\
& +a^2x_l^2+ab((3+5a\Delta t+2a^2\Delta t^2)x_L+(3+a\Delta t)x_l)\big]\\
E_4 &= \big[(2s^2+b^2)(3+3a\Delta t+a^2\Delta t^2)\\
& +6C(a\Delta t+1)^2+a^2x_R(a\Delta t+1)(x_r+x_L(a\Delta t+1))\\
& +a^2x_r^2+ab((3+5a\Delta t+2a^2\Delta t^2)x_L+(3+a\Delta t)x_r)\big]\\
E_5 &= \frac{1}{12(a\Delta t+1)^3(x_R-x_L)}\big[b\Delta t(b^2+3s^2)(3+3a\Delta t+a^2\Delta t^2)\\
& +x_R(3s^2+a^2x_R^2+6C+3b^2+3abx_R)(a\Delta t+1)^3\\
& -x_l(6C(a\Delta t+1)^2+3b^2+3s^2+3bax_l+a^2x_l^2)+6bC\Delta t(a\Delta t+1)^2\big]\\
E_6 &= \frac{1}{12(a\Delta t+1)^3(x_R-x_L)}\big[b\Delta t(b^2+3s^2)(3+3a\Delta t+a^2\Delta t^2)\\
& +x_R(3s^2+a^2x_R^2+6C+3b^2+3abx_R)(a\Delta t+1)^3\\
& -x_r(6C(a\Delta t+1)^2+3b^2+3s^2+3bax_l+a^2x_r^2)+6bC\Delta t(a\Delta t+1)^2\big]\\
E_7 &= \frac{1}{12(a\Delta t+1)^3(x_R-x_L)}\big[b\Delta t(b^2+3s^2)(3+3a\Delta t+a^2\Delta t^2)\\
& +x_L(3s^2+a^2x_L^2+6C+3b^2+3abx_L)(a\Delta t+1)^3\\
& -x_l(6C(a\Delta t+1)^2+3b^2+3s^2+3bax_l+a^2x_l^2)+6bC\Delta t(a\Delta t+1)^2\big]\\
E_8 &= \frac{1}{12(a\Delta t+1)^3(x_R-x_L)}\big[b\Delta t(b^2+3s^2)(3+3a\Delta t+a^2\Delta t^2)\\
& +x_L(3s^2+a^2x_L^2+6C+3b^2+3abx_L)(a\Delta t+1)^3\\
& -x_r(6C(a\Delta t+1)^2+3b^2+3s^2+3bax_l+a^2x_r^2)+6bC\Delta t(a\Delta t+1)^2\big]
\end{aligned}
$$

13.4 Hypersonic flow over a rectangular body

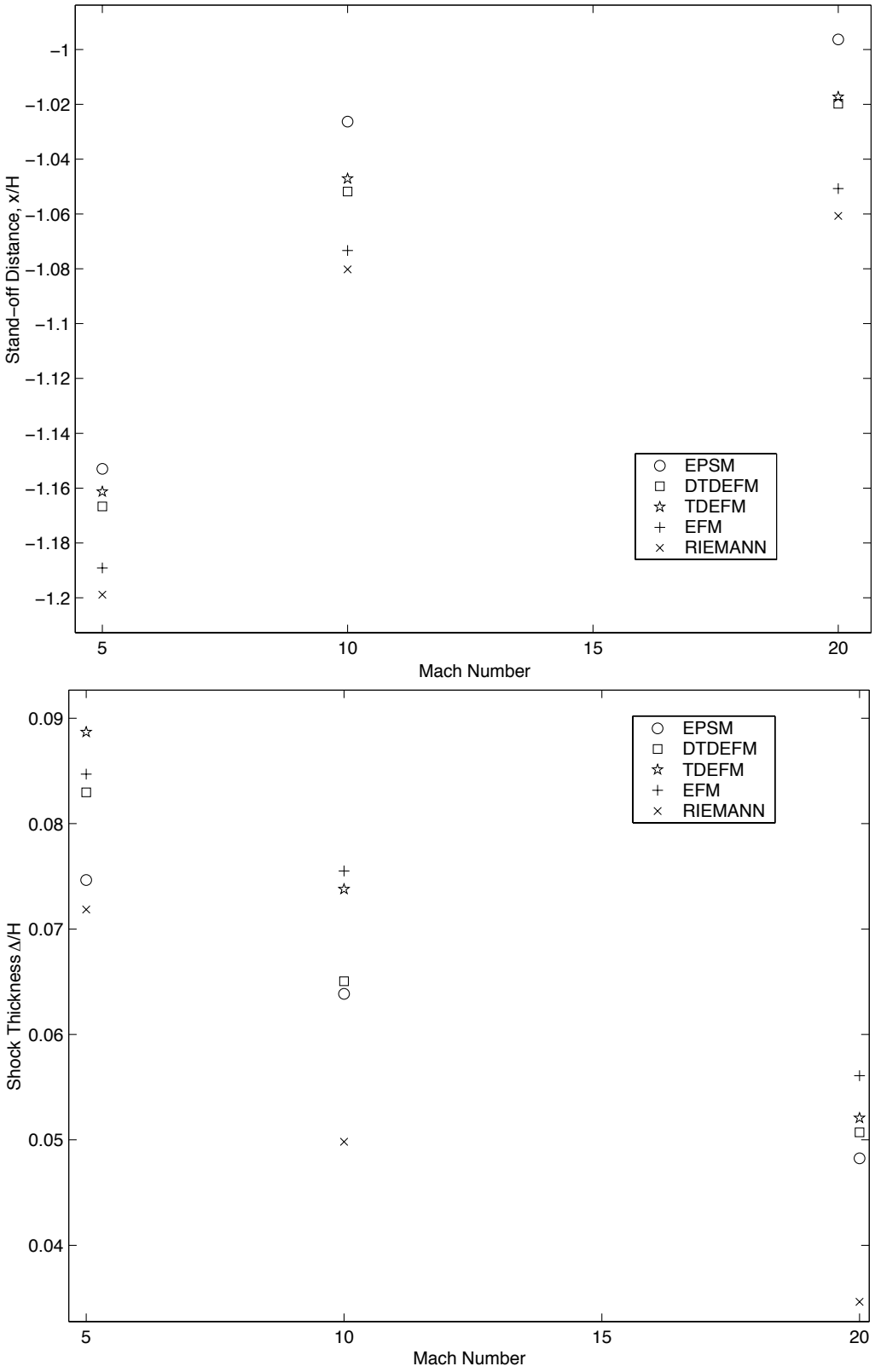

Figure 13.1: (Top) Shock standoff distance and (Bottom) Normalised shock thickness from EPSM, DTDEFM and TDEFM, EFM and a Godunov Solver. DTDEFM resulted obtained using MINMOD to calculate density gradients in the source cells. $\gamma = 7/5$. $\Delta x = \Delta y = H/30$

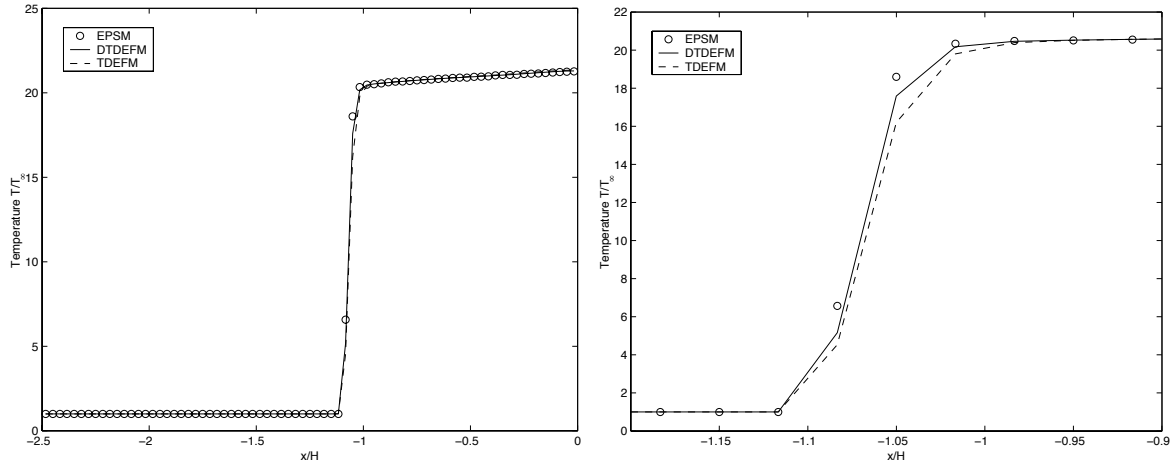

Figure 13.2: Temperature profiles along stagnation line showing results from EPSM, DTDEFM and TDEFM. $M_\infty = 10$, $\gamma = 7/5$. $\Delta x = \Delta y = H/30$. DTDEFM resulted obtained using MINMOD to calculate density gradients in the source cells.

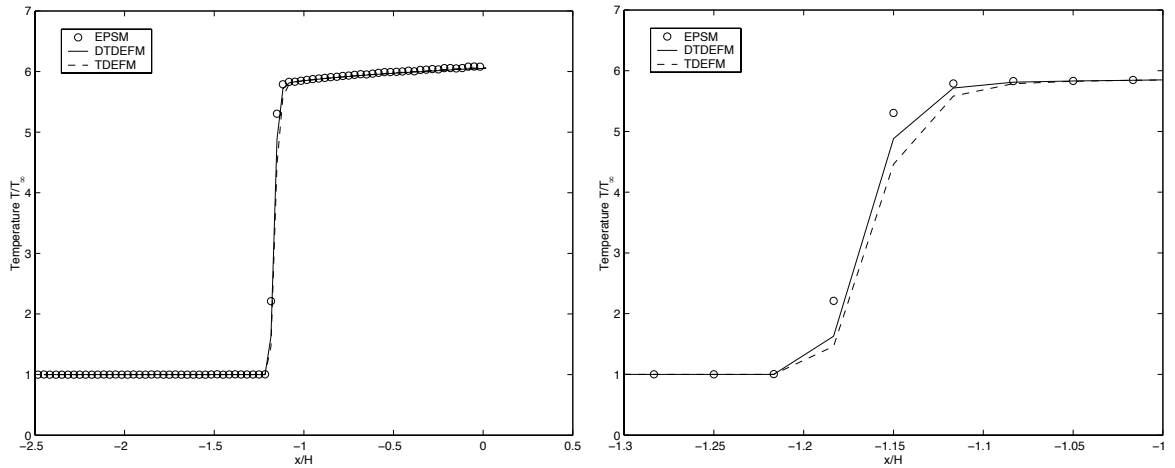

Figure 13.3: Temperature profiles along stagnation line showing results from EPSM, DTDEFM and TDEFM. $M_\infty = 5$, $\gamma = 7/5$. $\Delta x = \Delta y = H/30$. DTDEFM resulted obtained using MINMOD to calculate density gradients in the source cells.

13.5 2D Hypersonic flow over a forward facing step

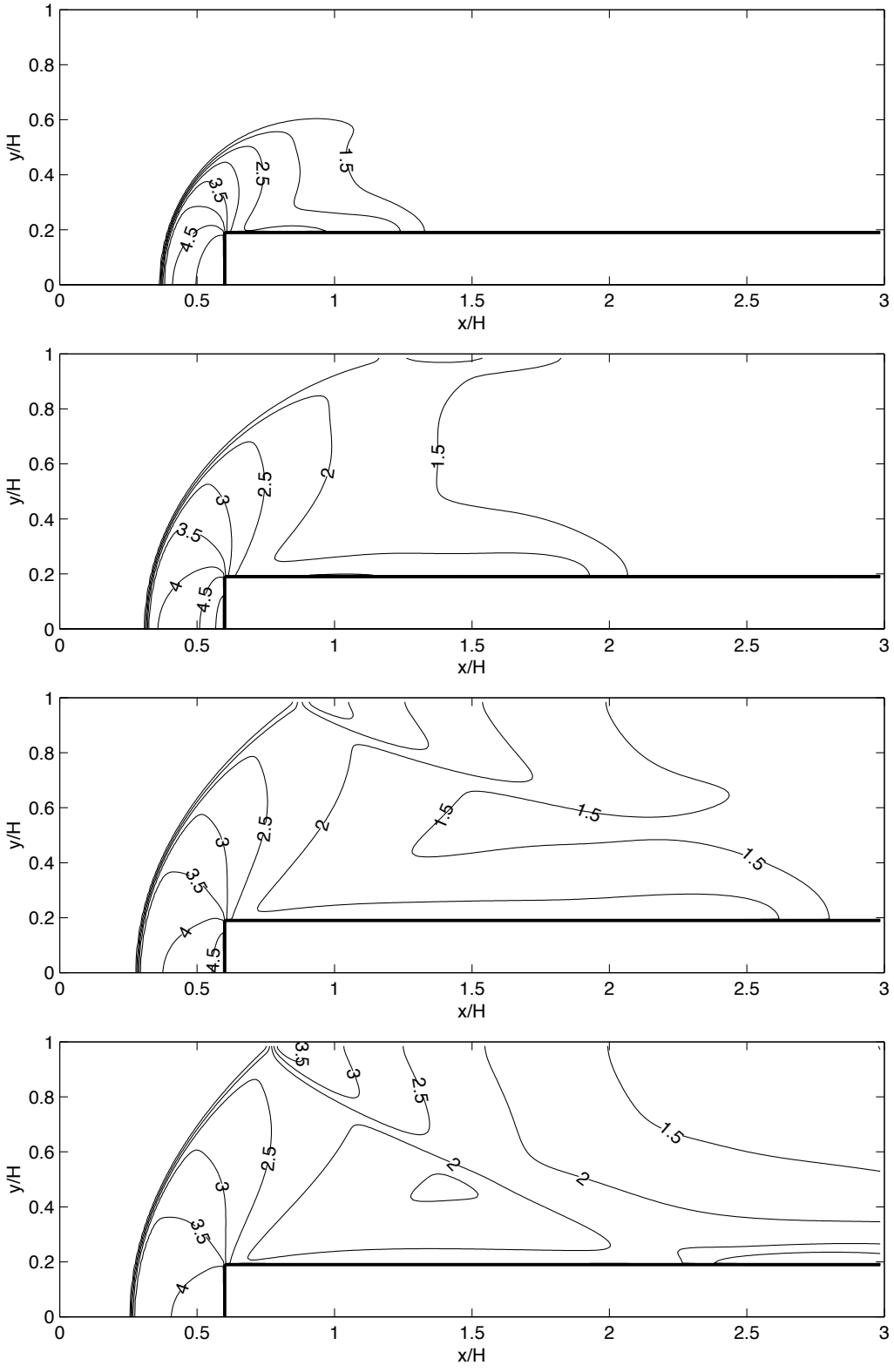

Figure 13.4: Temperature contours taken from the EFM solution after 400, 800, 1200 and 1600 timesteps of $\Delta t = 0.0005$. The gas is ideal with $\gamma = 5/3$. The grid resolution was 100x300 cells. The freestream flow mach number is $M = 3$. The gas is inviscid and all bounaries are specularly reflective with exception to the inflow and outflow boundaries.

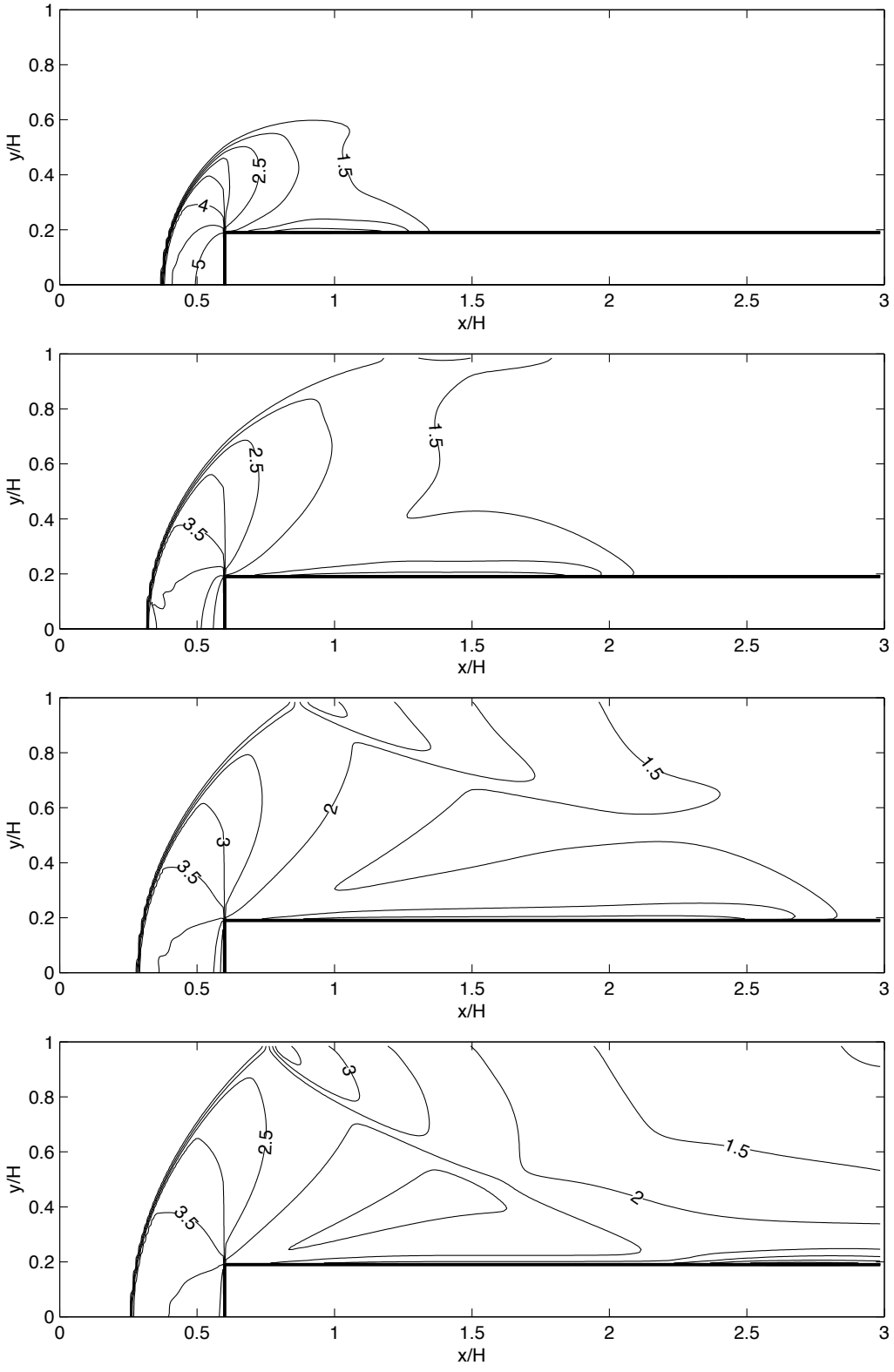

Figure 13.5: Temperature contours taken from the Riemann solution after 400, 800, 1200 and 1600 timesteps of $\Delta t = 0.0005$. The gas is ideal with $\gamma = 5/3$. The grid resolution was 100x300 cells. The freestream flow mach number is $M = 3$. The gas is inviscid and all bounaries are specularly reflective with exception to the inflow and outflow boundaries.

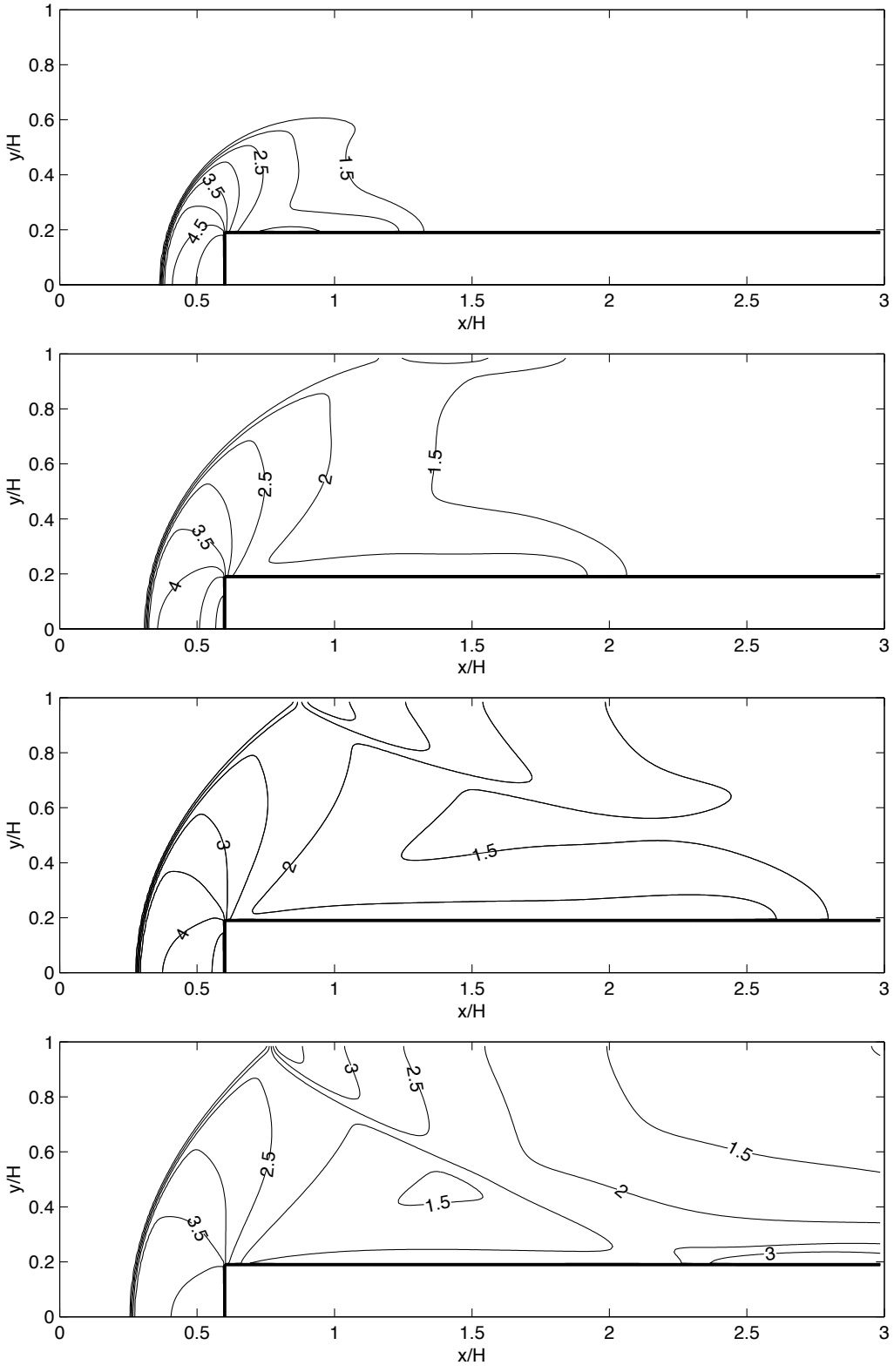

Figure 13.6: Temperature contours taken from the TDEFM solution after 400, 800, 1200 and 1600 timesteps of $\Delta t = 0.0005$. The gas is ideal with $\gamma = 5/3$. The grid resolution was 100x300 cells. The freestream flow mach number is $M = 3$. The gas is inviscid and all bounaries are specularly reflective with exception to the inflow and outflow boundaries.

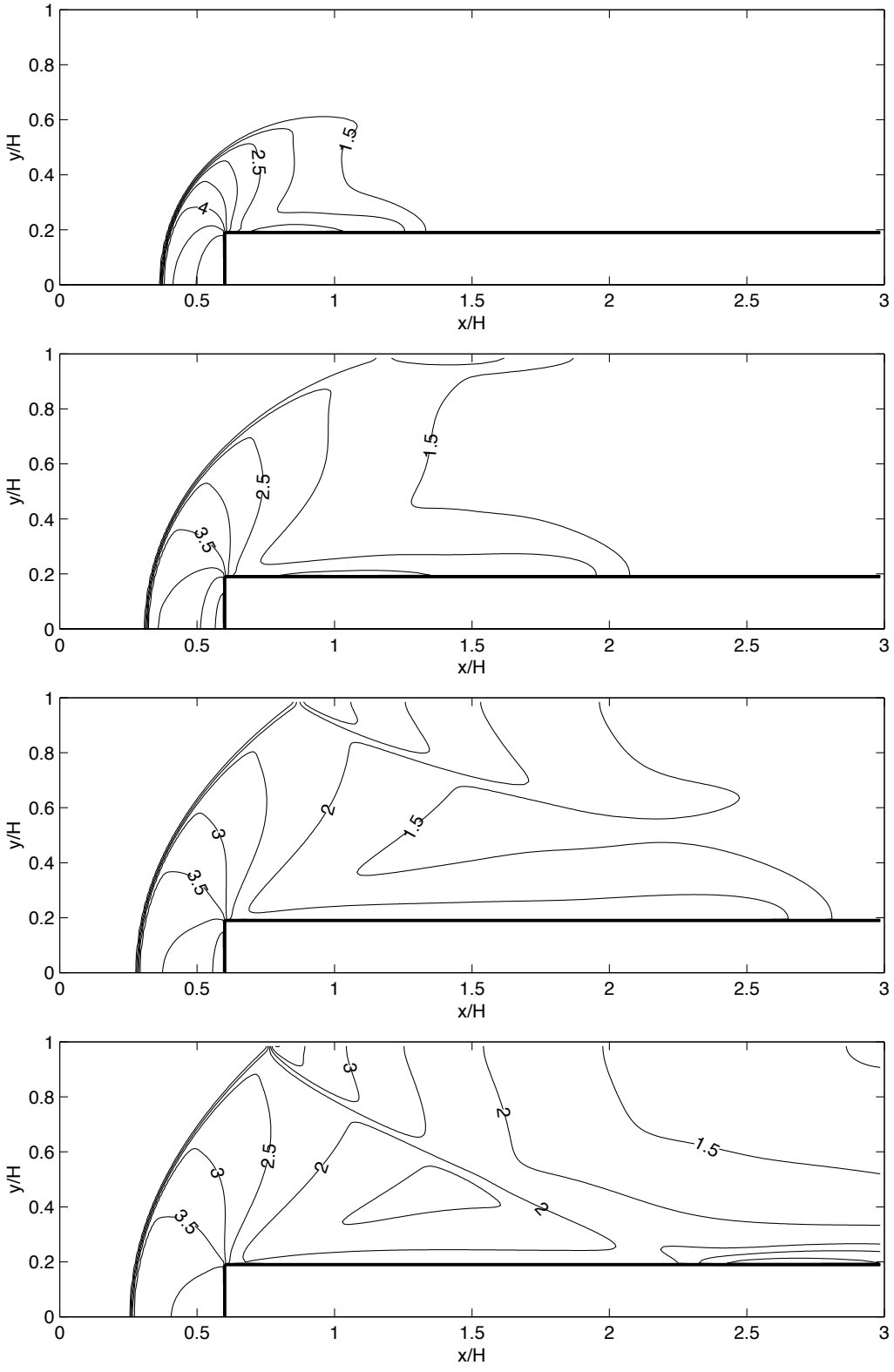

Figure 13.7: Temperature contours taken from the DTDEFM solution after 400, 800, 1200 and 1600 timesteps of $\Delta t = 0.0005$. The gas is ideal with $\gamma = 5/3$. The grid resolution was 100x300 cells. The freestream flow mach number is $M = 3$. The gas is inviscid and all bounaries are specularly reflective with exception to the inflow and outflow boundaries.

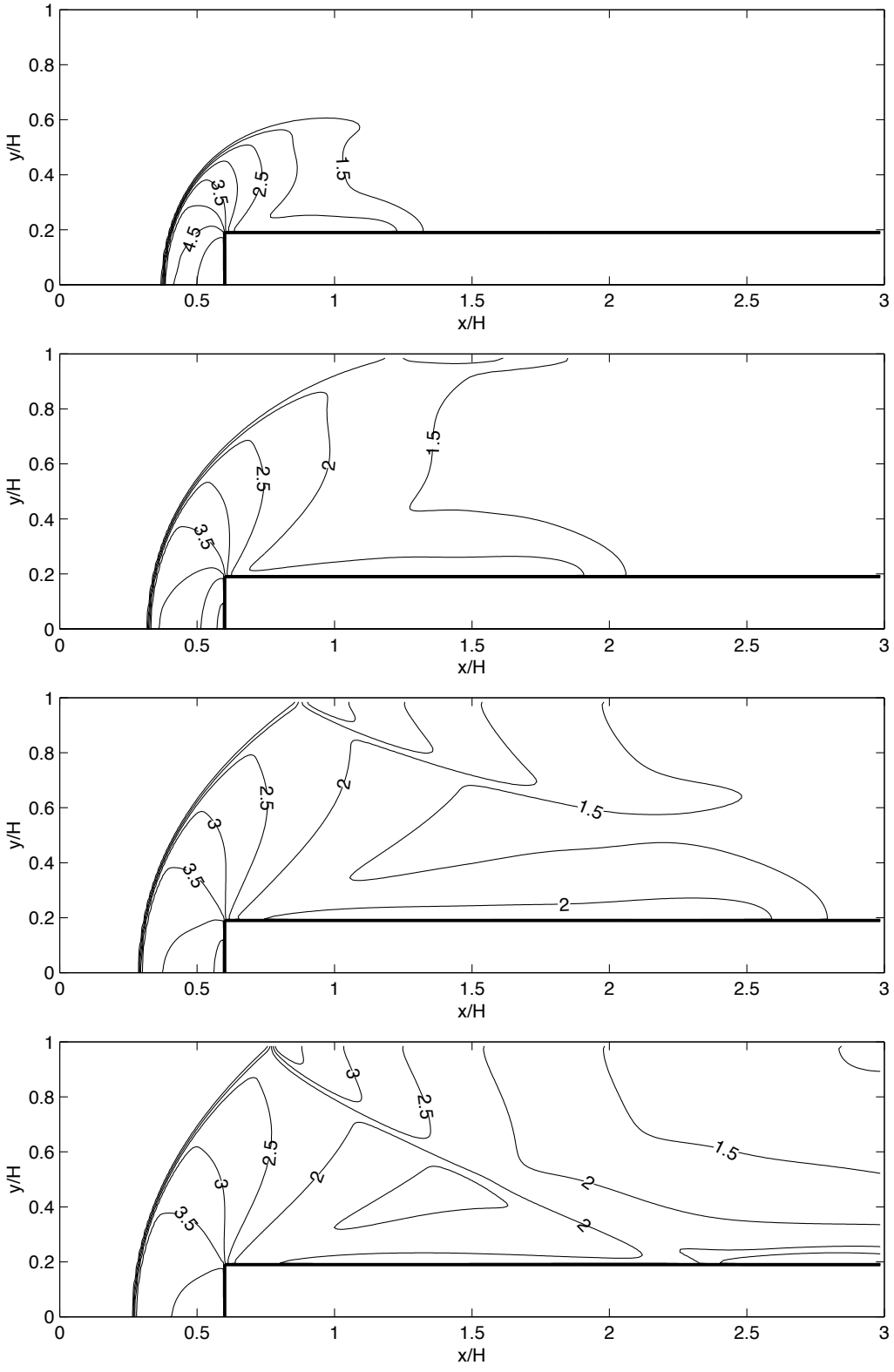

Figure 13.8: Temperature contours taken from the VTDEFM solution after 400, 800, 1200 and 1600 timesteps of $\Delta t = 0.0005$. The gas is ideal with $\gamma = 5/3$. The grid resolution was 100x300 cells. The freestream flow mach number is $M = 3$. The gas is inviscid and all bounaries are specularly reflective with exception to the inflow and outflow boundaries.

13.6 Shock wave interaction with I beam

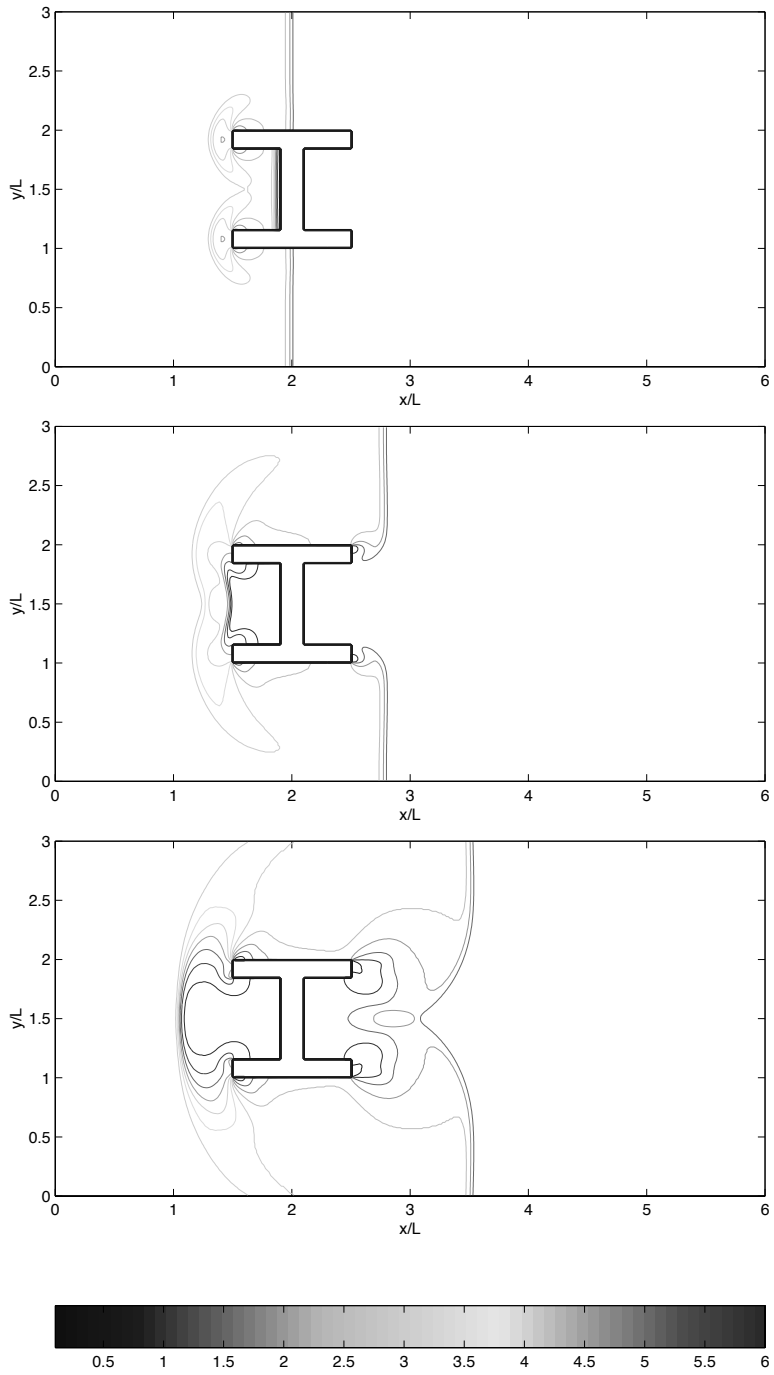

Figure 13.9: Density contours from EFM solution for a shock wave interaction with the I-beam shown in Figure 10.30 at times $\sqrt{RT}t/L = 0.05, 0.104$ and 0.154 where t is the flow time. The computational grid employs 400x200 cells. The gas in assumed inviscid with $\gamma = 1.4$. The approaching shock creates a pressure increase of $p_H/p_L = 4.5$ and a temperature increase of $T_H/T_L = 1.687$. All surface reflections off the I-beam are treated as specular. All boudaries encapsulating the computational grid are extraploated from internal cells.

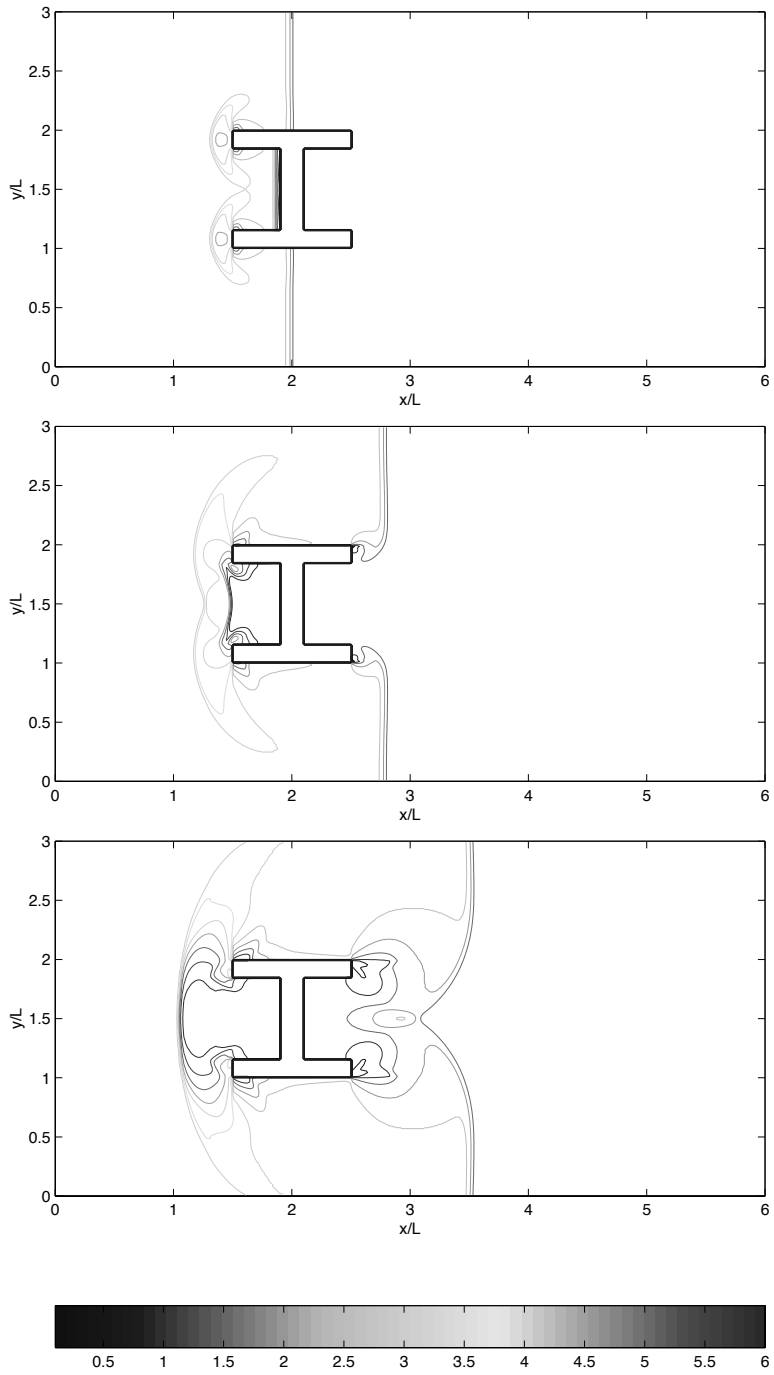

Figure 13.10: Density contours from the Riemann solution for a shock wave interaction with the I-beam shown in Figure 10.30 at times $\sqrt{RT}t/L = 0.05, 0.104$ and 0.154 where t is the flow time. The computational grid employs 400x200 cells. The gas in assumed inviscid with $\gamma = 1.4$. The approaching shock creates a pressure increase of $p_H/p_L = 4.5$ and a temperature increase of $T_H/T_L = 1.687$. All surface reflections off the I-beam are treated as specular. All boudaries encapsulating the computational grid are extraploated from internal cells.

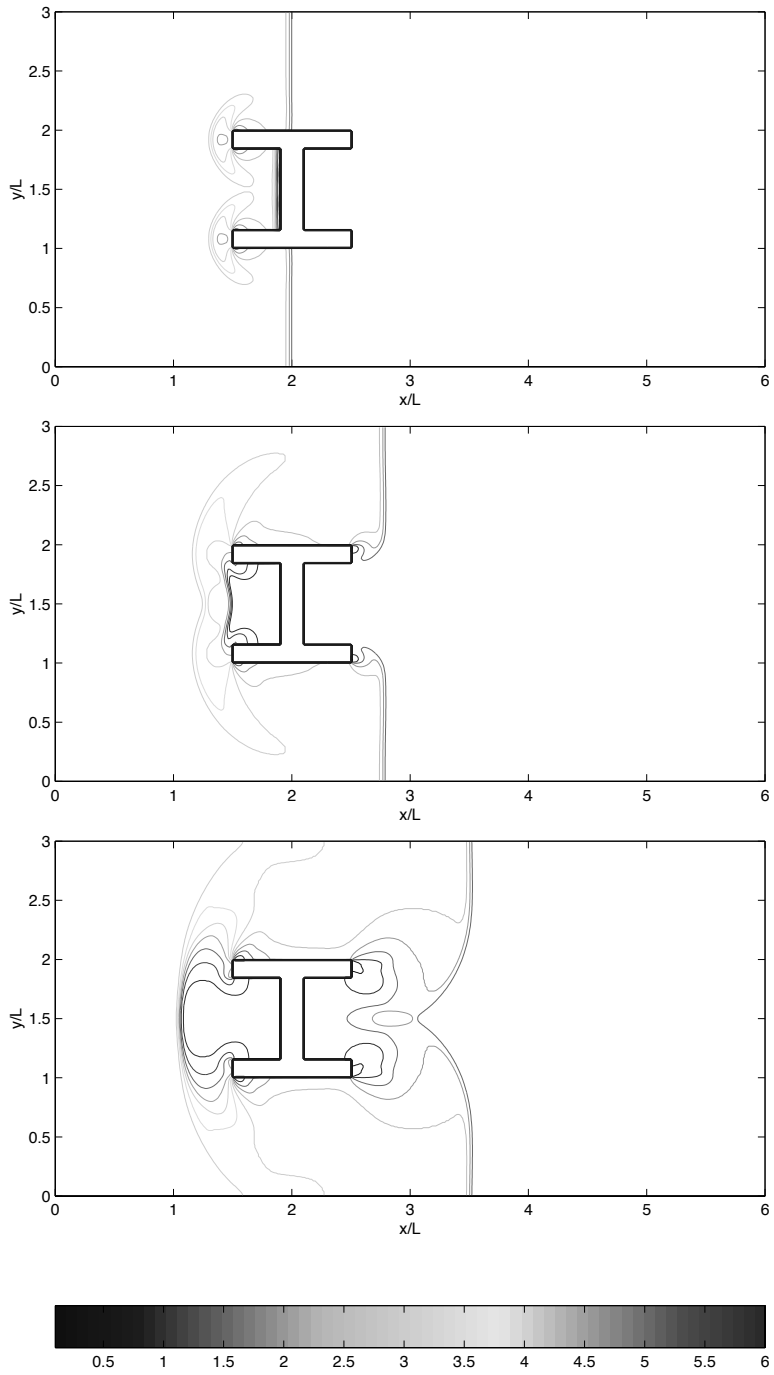

Figure 13.11: Density contours from the DTDEFM solution for a shock wave interaction with the I-beam shown in Figure 10.30 at times $\sqrt{RT}t/L = 0.05, 0.104$ and 0.154 where t is the flow time. The computational grid employs 400x200 cells. The gas in assumed inviscid with $\gamma = 1.4$. The approaching shock creates a pressure increase of $p_H/p_L = 4.5$ and a temperature increase of $T_H/T_L = 1.687$. All surface reflections off the I-beam are treated as specular. All boudaries encapsulating the computational grid are extraploated from internal cells.

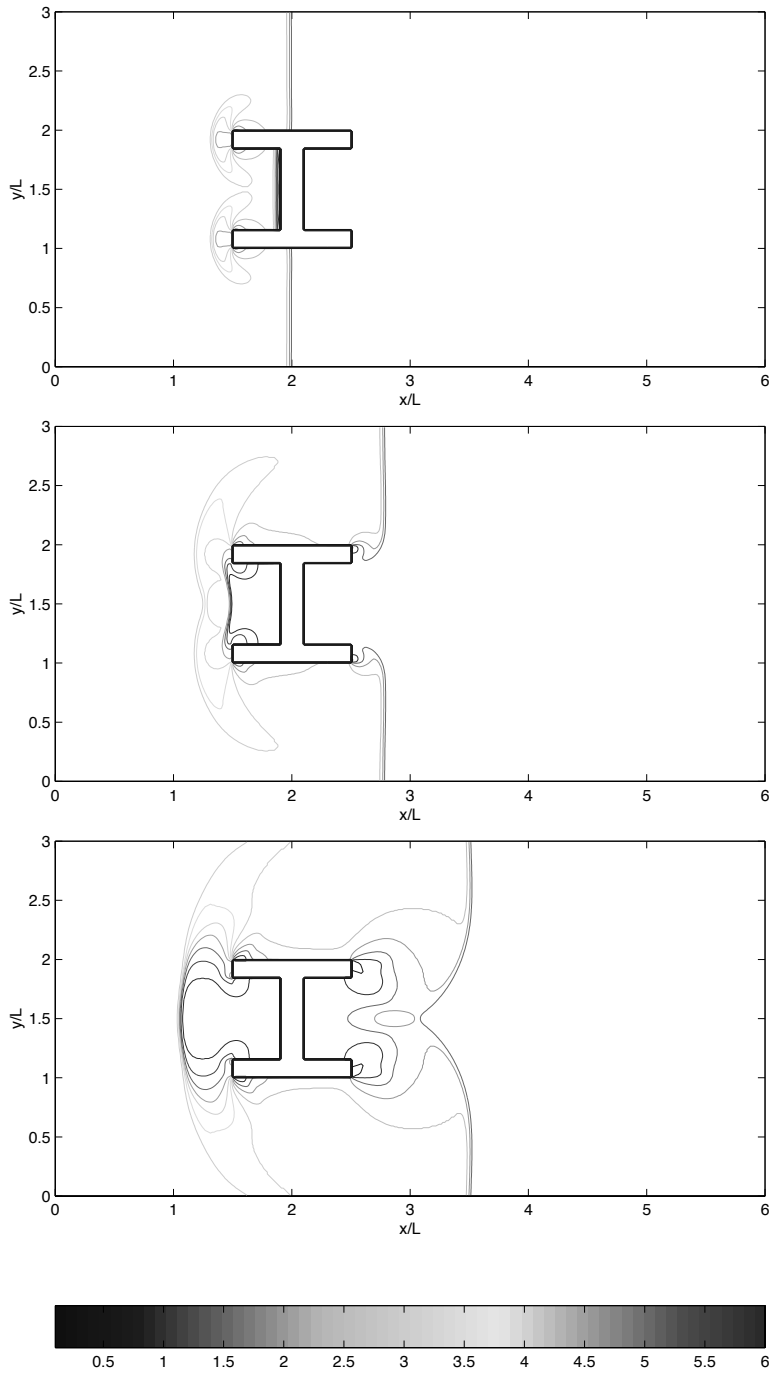

Figure 13.12: Density contours from the VTDEFM solution for a shock wave interaction with the I-beam shown in Figure 10.30 at times $\sqrt{RT}t/L = 0.05, 0.104$ and 0.154 where t is the flow time. The computational grid employs 400x200 cells. The gas in assumed inviscid with $\gamma = 1.4$. The approaching shock creates a pressure increase of $p_H/p_L = 4.5$ and a temperature increase of $T_H/T_L = 1.687$. All surface reflections off the I-beam are treated as specular. All boudaries encapsulating the computational grid are extraploated from internal cells.

13.7 Blast waves around city buildings

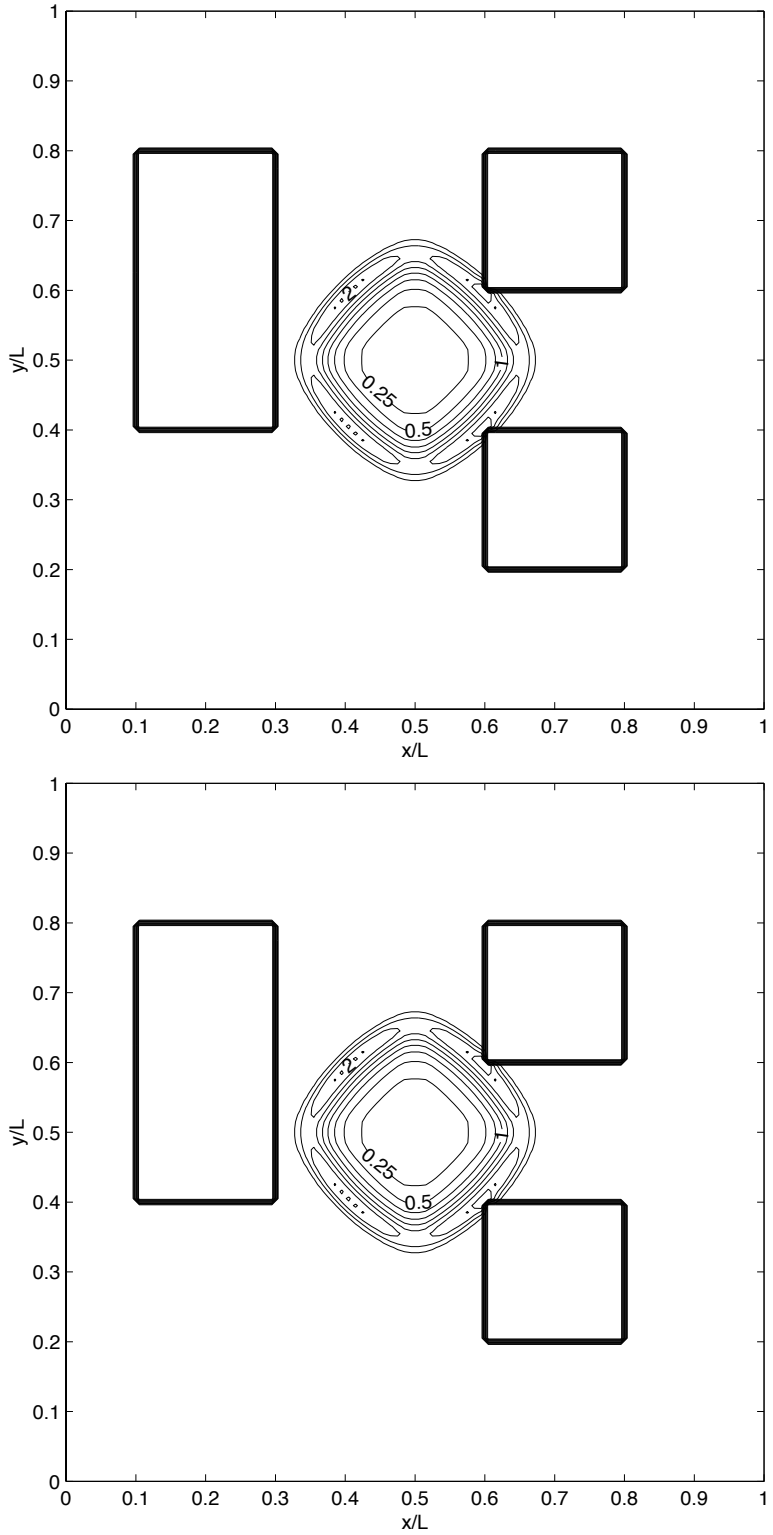

Figure 13.13: Results for a Roe solver [Top] and CFD-FASTRAN [Bottom] at time $t\sqrt{RT}/L = 0.01$. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extraploated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.40. The computational grid is a cartesian mesh of 100x100 cells. The CFD-FASTRAN computational grid employed 40 blocks with a total of 8400 cells.

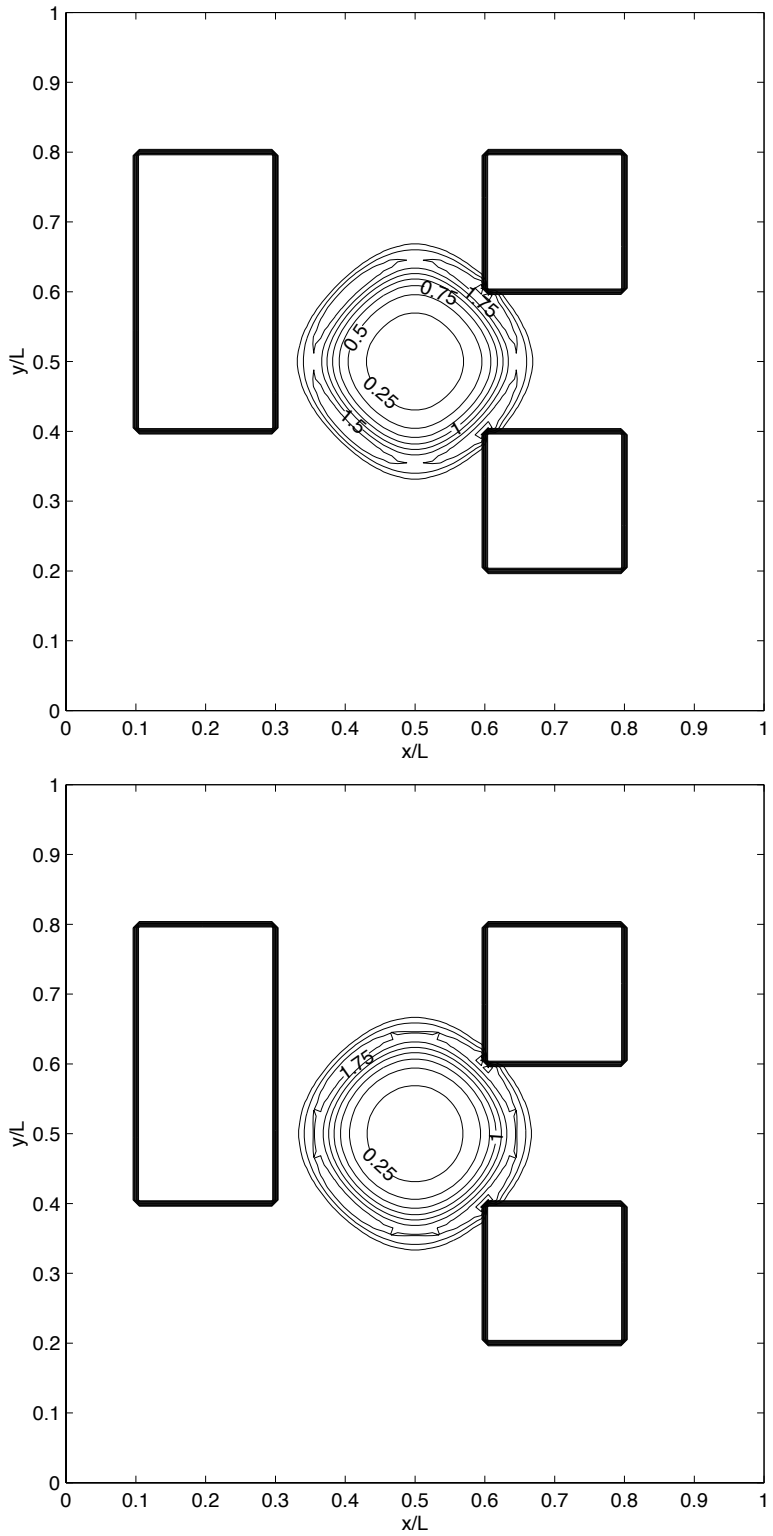

Figure 13.14: Results for EFM [Top] and TDEFM [Bottom] at time $t\sqrt{RT}/L = 0.01$. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extraploated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.40. The computational grid is a cartesian mesh of 100x100 cells.

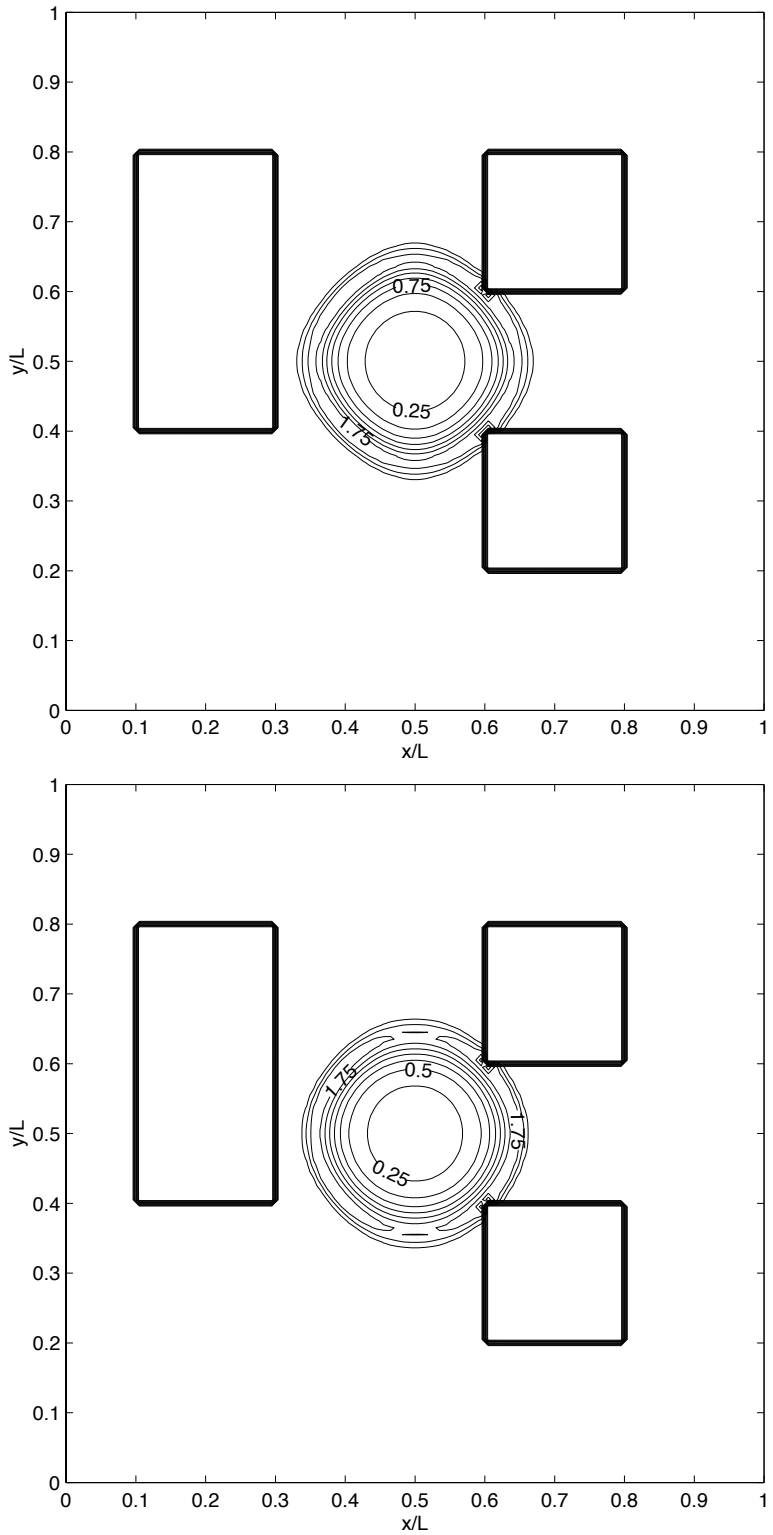

Figure 13.15: Results for DTDEFM [Top] and VTDEFM [Bottom] at time $t\sqrt{RT}/L = 0.01$. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extraploated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.40. The computational grid is a cartesian mesh of 100x100 cells.

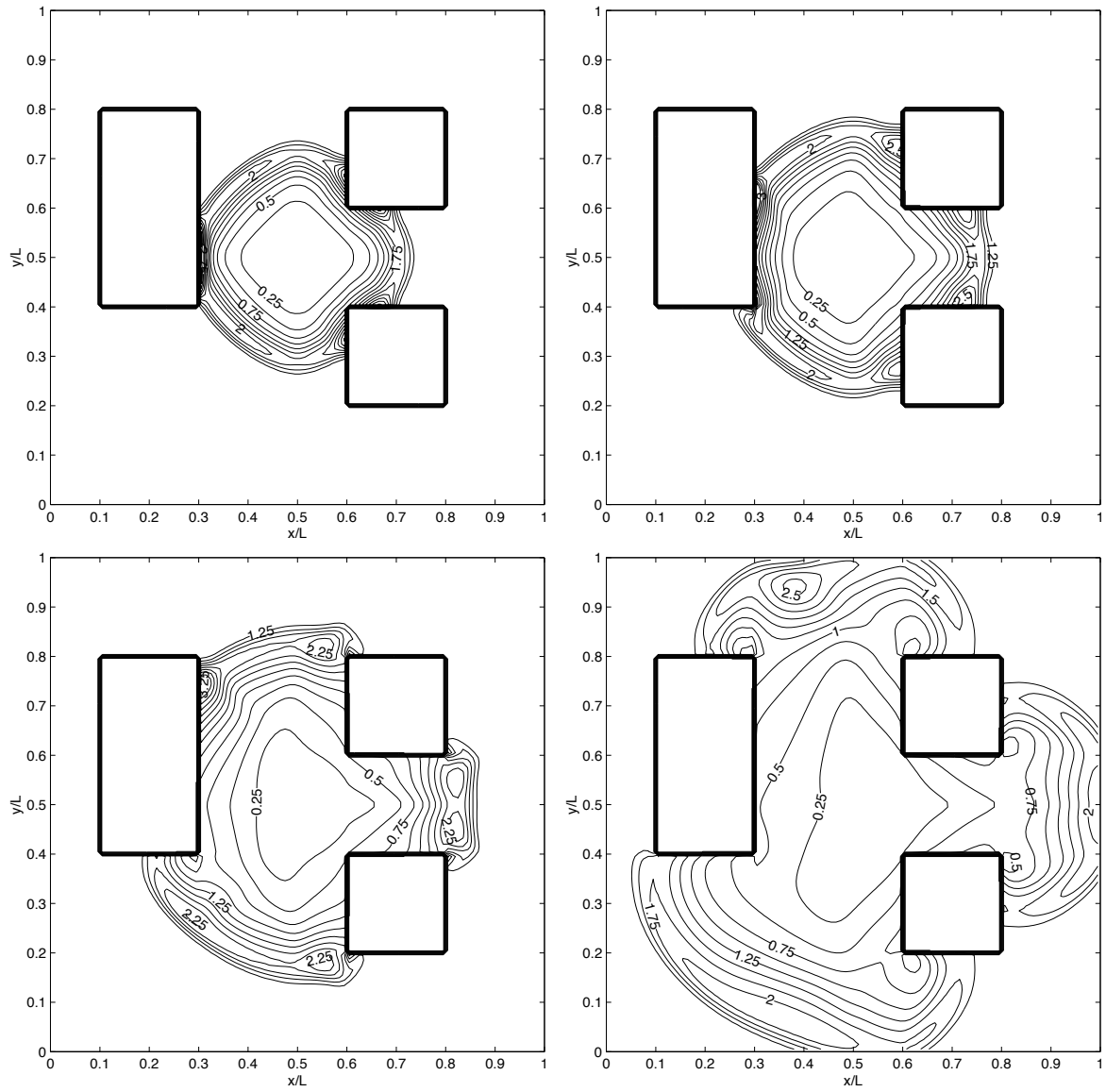

Figure 13.16: Density contours taken from the Riemann solver results at times $t\sqrt{RT}/L = 0.02, 0.03, 0.05$ and 0.1. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extraploated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.40. The computational grid is a cartesian mesh of 100x100 cells.

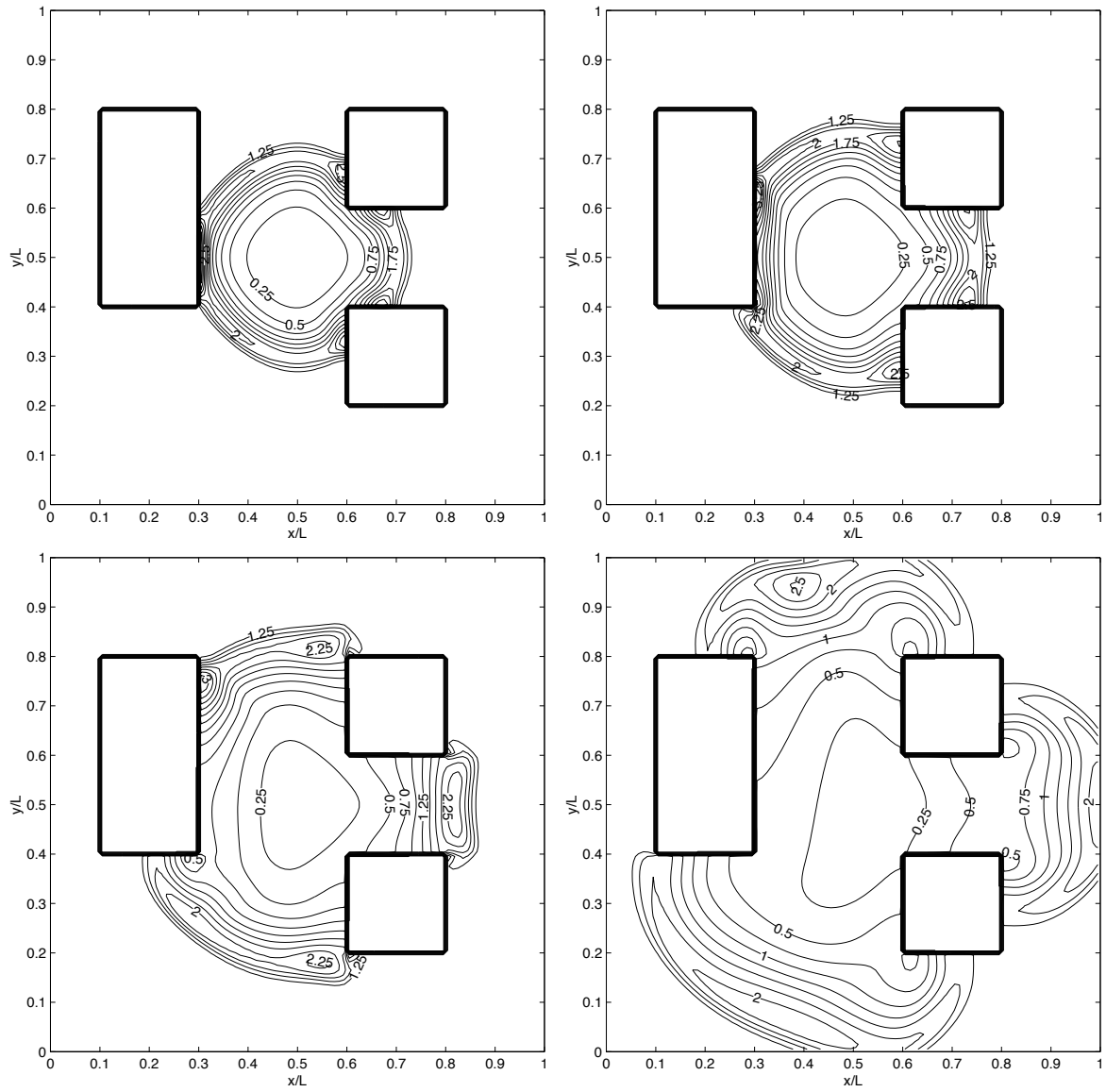

Figure 13.17: Density contours taken from EFM results at times $t\sqrt{RT}/L = 0.02, 0.03, 0.05$ and 0.1. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extraploated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.40. The computational grid is a cartesian mesh of 100x100 cells.

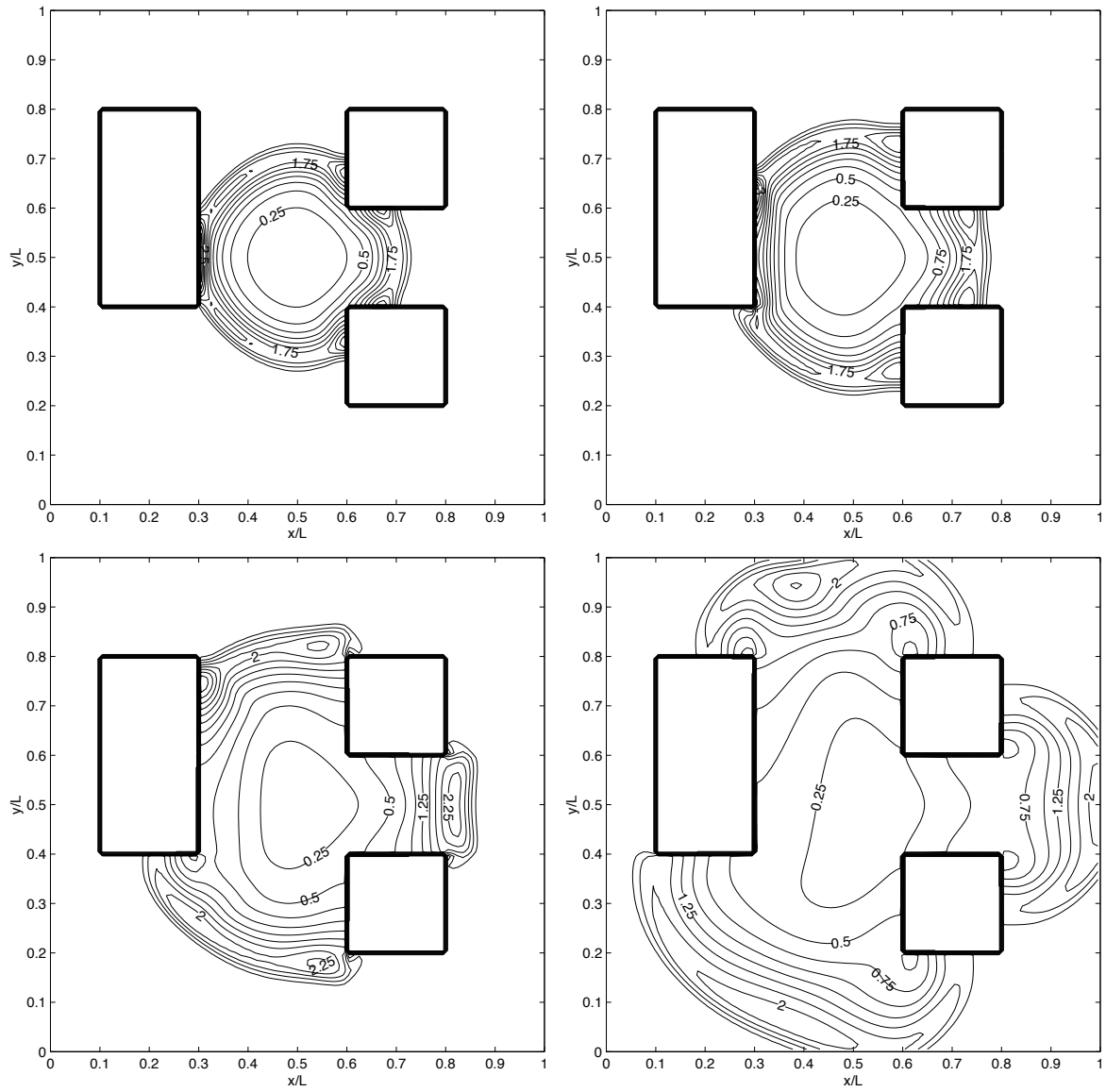

Figure 13.18: Density contours taken from TDEFM results at times $t\sqrt{RT}/L = 0.02, 0.03, 0.05$ and 0.1. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extraploated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.40. The computational grid is a cartesian mesh of 100x100 cells.

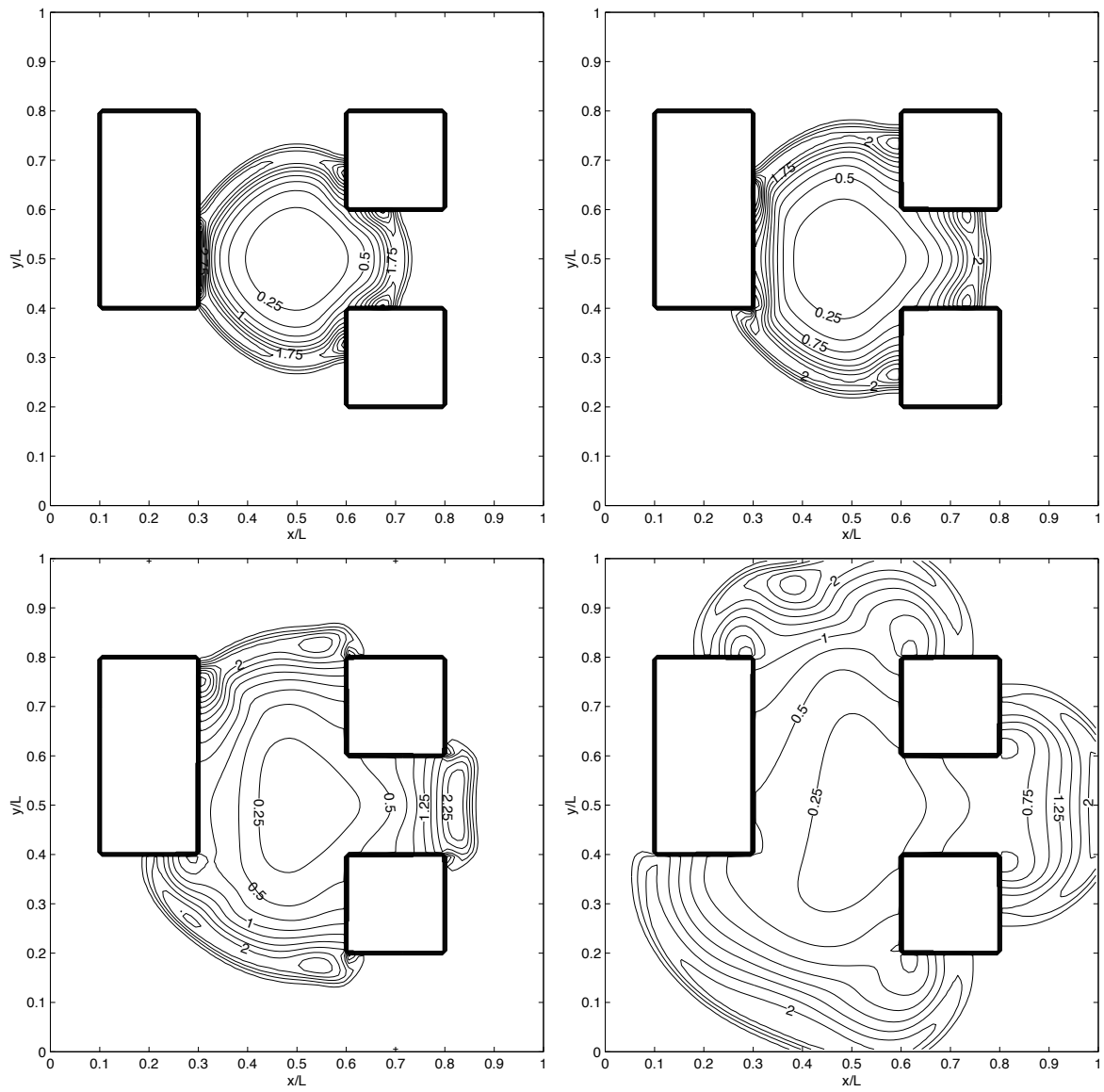

Figure 13.19: Density contours taken from DTDEFM results at times $t\sqrt{RT}/L = 0.02, 0.03, 0.05$ and 0.1. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extraploated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.40. The computational grid is a cartesian mesh of 100x100 cells.

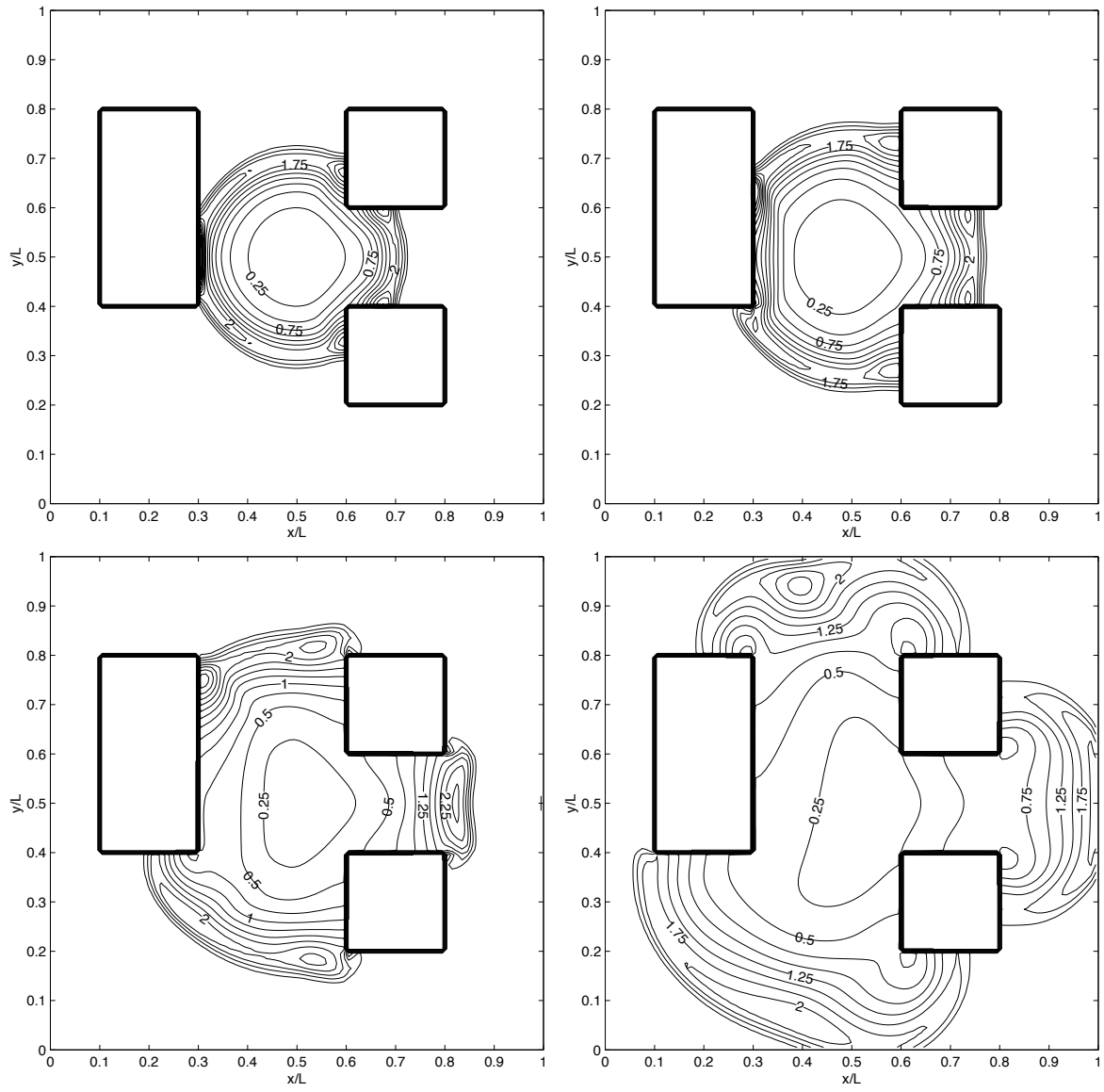

Figure 13.20: Density contours taken from VTDEFM results at times $t\sqrt{RT}/L = 0.02, 0.03, 0.05$ and 0.1. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extraploated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.40. The computational grid is a cartesian mesh of 100x100 cells.

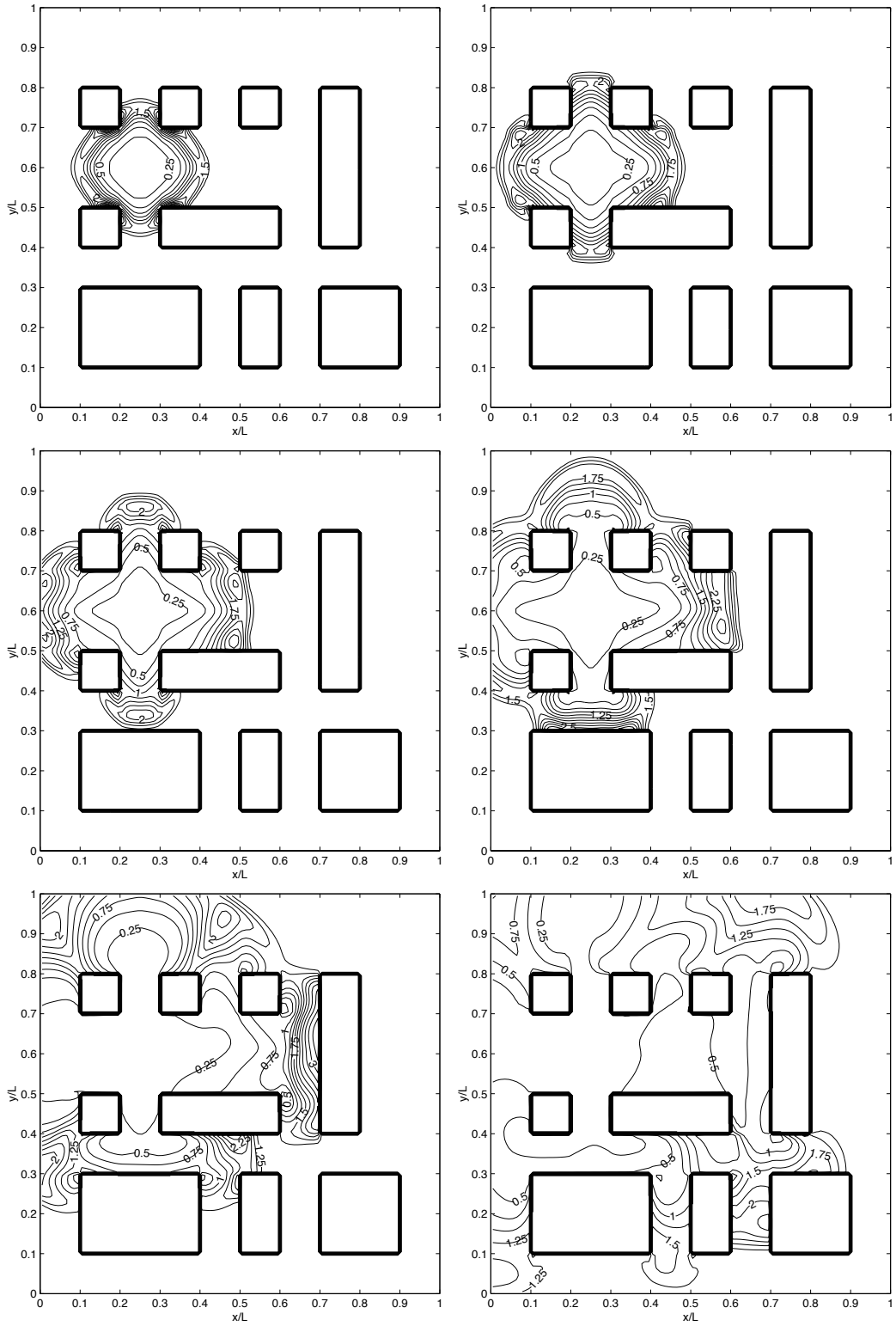

Figure 13.21: Density contours taken from the Riemann solver results at times $t\sqrt{RT}/L = 0.01, 0.02, 0.03, 0.05, 0.1$ and 0.2. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extraploated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.46. The computational grid is a cartesian mesh of 100x100 cells.

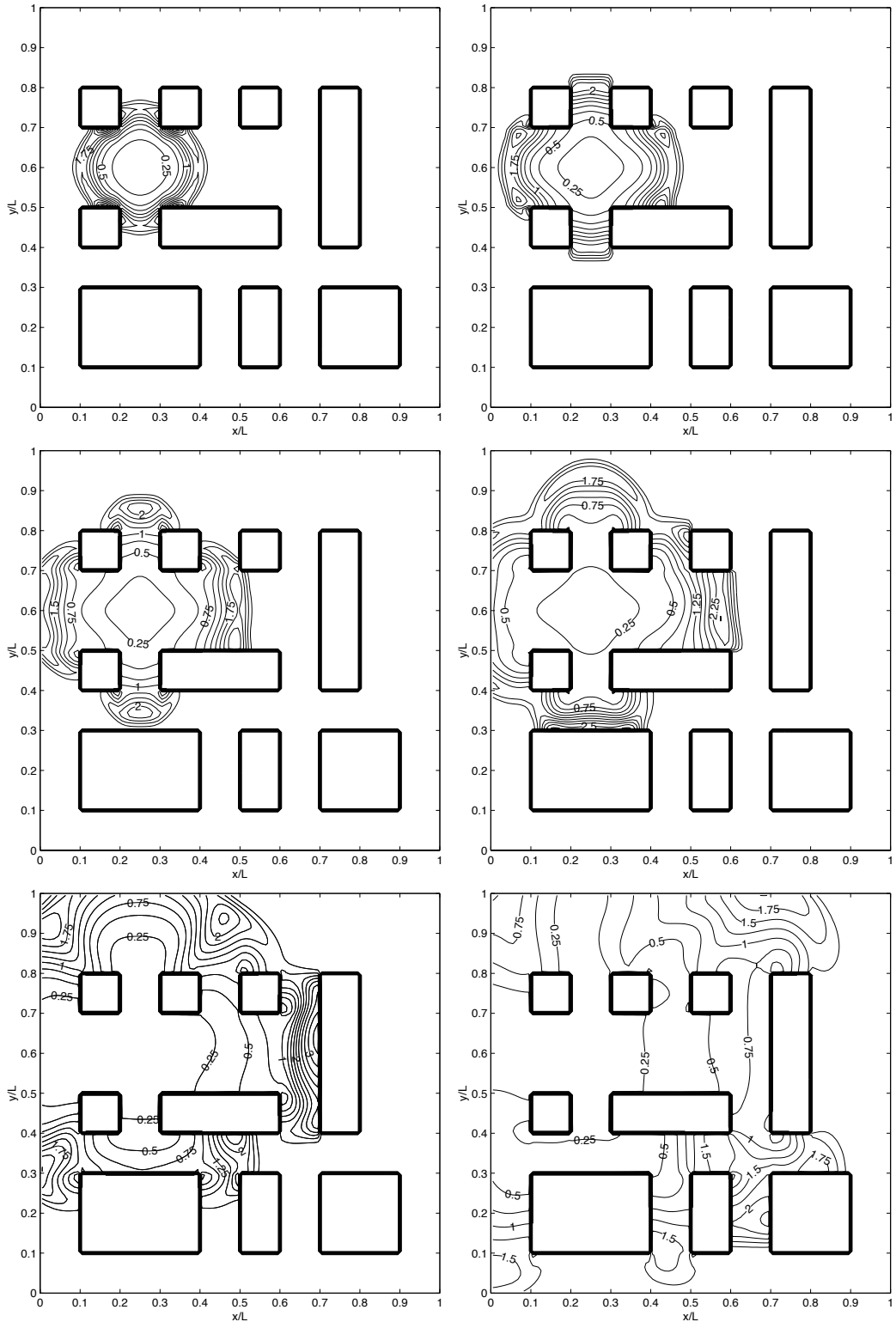

Figure 13.22: Density contours taken from EFM results at times $t\sqrt{RT}/L = 0.01, 0.02, 0.03, 0.05, 0.1$ and 0.2. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extraploated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.46. The computational grid is a cartesian mesh of 100x100 cells.

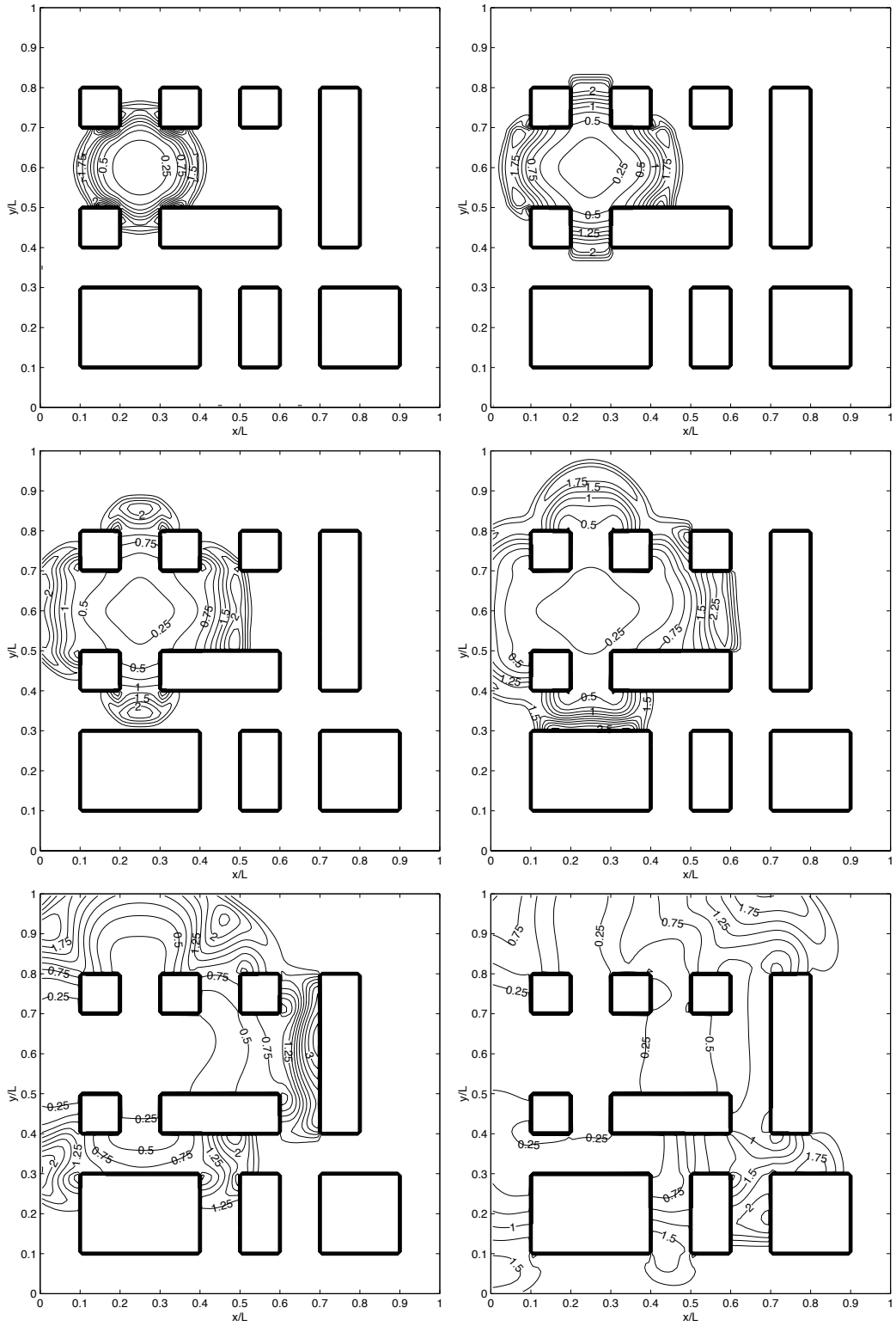

Figure 13.23: Density contours taken from TDEFM results at times $t\sqrt{RT}/L = 0.01, 0.02, 0.03, 0.05, 0.1$ and 0.2. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extraploated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.46. The computational grid is a cartesian mesh of 100x100 cells.

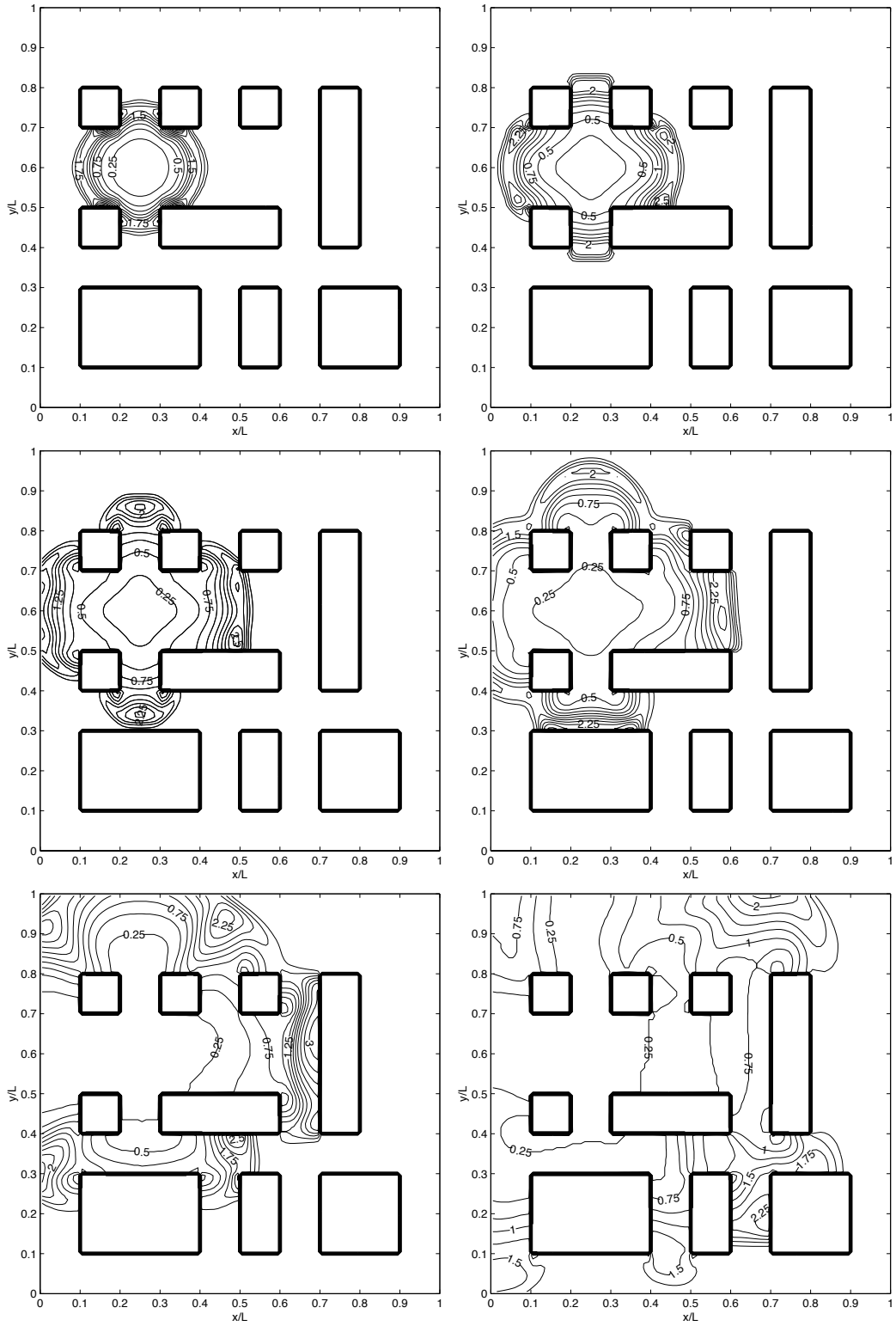

Figure 13.24: Density contours taken from DTDEFM results at times $t\sqrt{RT}/L = 0.01, 0.02, 0.03, 0.05, 0.1$ and 0.2. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extraploated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.46. The computational grid is a cartesian mesh of 100x100 cells.

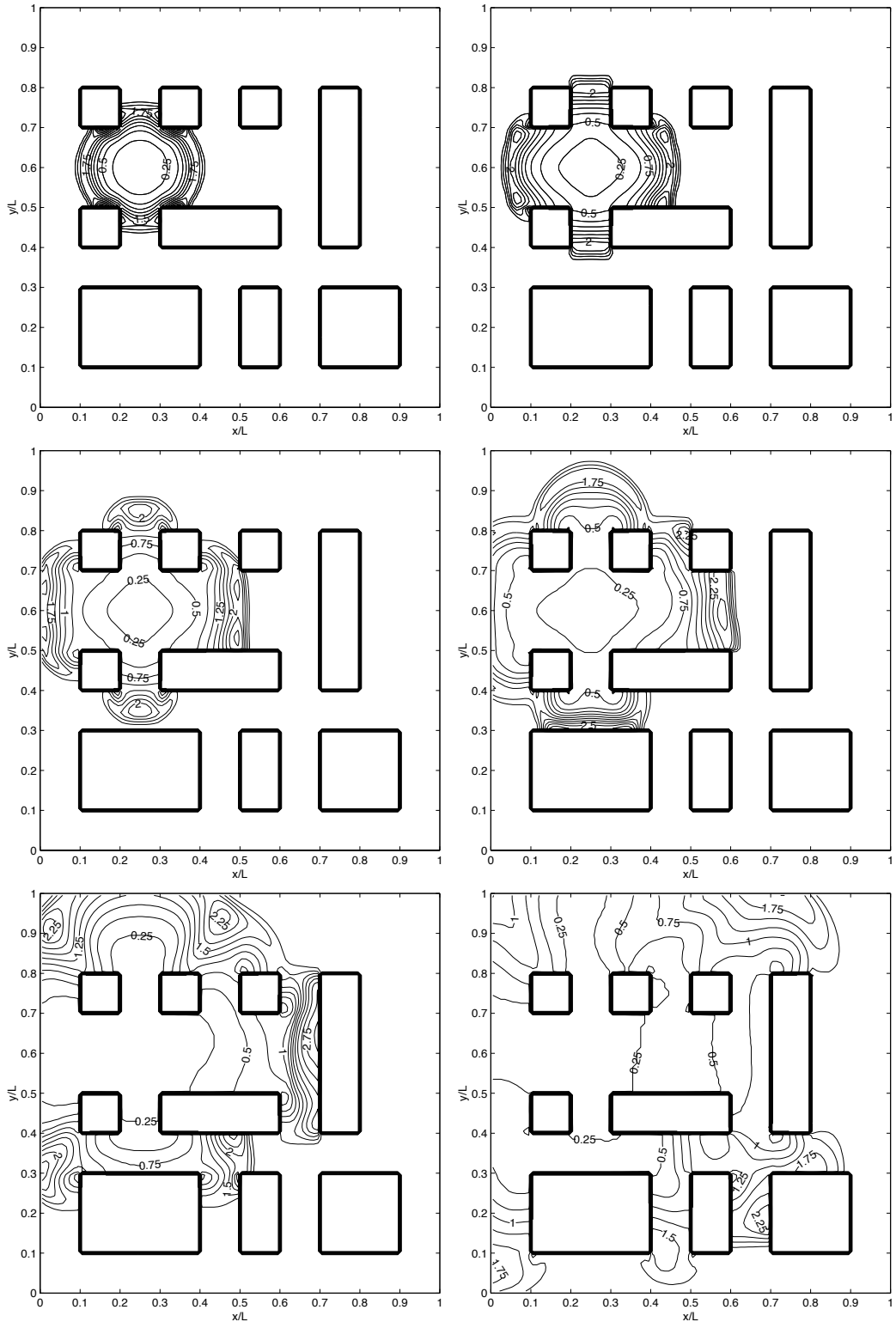

Figure 13.25: Density contours taken from VTDEFM results at times $t\sqrt{RT}/L = 0.01, 0.02, 0.03, 0.05, 0.1$ and 0.2. The gas is inviscid and monatomic. All building surfaces are treated as fully reflective while the edges bounding the computational domain are extraploated outflow. The initial 'bomb' region is the high temperature and pressure region shown in Figure 10.46. The computational grid is a cartesian mesh of 100x100 cells.

13.8 Properties of the Error Function

The Error function is defined as:

$$erf(z) = \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2} dt \tag{13.2}$$

And can also be defined in terms of the incomplete gamma function $\gamma(\alpha, \chi)$ as:

$$erf(z) = \pi^{-1/2} \gamma(\frac{1}{2}, Z^2) \tag{13.3}$$

As a Maclaurin series, the error function is defined as:

$$\begin{aligned} erf(z) &= \frac{2}{\sqrt{\pi}} \sum_{n=0}^{\infty} \frac{(-1)^n z^{2n+1}}{n!(2n+1)} \\ &= \frac{2}{\sqrt{\pi}} \left(z - \frac{1}{3} z^3 + \frac{1}{10} z^5 - \frac{1}{42} z^7 + \frac{1}{216} z^9 ... \right) \end{aligned} \tag{13.4}$$

13.9 C++ Code for TDEFM function

```
void tdefmflux(double xL, double xR, double xl, double xr,
double t, double m, double s, double *DATA) {

double m2, s2, t2, tm, denom;
double LMl, LMr, RMl, RMr, RML, LMR;
double expLMl, expLMr, expRMl, expRMr;
double erfLMl, erfLMr, erfRMl, erfRMr;
double expcoeffm, expcoeffp, expcoeffe;
double sdx, C, OD;
OD = 2; // Number of dimensions used in simulation
m2 = m*m; // Bulk velocity in source cell squared
s2 = s*s; // Variance (sqrt(RT)) squared
t2 = t*t; // Timestep squared
tm = t*m;
denom = sqrt(2)*t*s;
sdx = xR - xL; // 1D width of source cell

RMl = (xR-xl+tm)/denom;
RMr = (xR-xr+tm)/denom;
LMl = (xL-xl+tm)/denom;
LMr = (xL-xr+tm)/denom;
expRMr = exp(-RMr*RMr);
```

```
expRMl = exp(-RMl*RMl);
expLMl = exp(-LMl*LMl);
expLMr = exp(-LMr*LMr);
erfLMl = erf(LMl);
erfLMr = erf(LMr);
erfRMl = erf(RMl);
erfRMr = erf(RMr);
C = (1/OD)*(0.5*((2*Cv/R)-OD)*s*s);
//C = Amount of energy (per unit mass) each particle carries
expcoeffm = t*s/sqrt(2*MPI);
expcoeffp = t*m*s/sqrt(2*MPI);
expcoeffe = t*s*(m2+2*s2+2*C)/(2*sqrt(2*MPI));

// MASS FLUX
DATA[0] = (1/sdx)*(expcoeffm*(expRMl-expRMr-expLMl+expLMr)
+ 0.5*(tm + xR - xl)*erfRMl
- 0.5*(tm + xR - xr)*erfRMr
- 0.5*(tm + xL - xl)*erfLMl
+ 0.5*(tm + xL - xr)*erfLMr);

// MOMENTUM FLUX
DATA[1] = (1/sdx)*(expcoeffp*(expRMl-expRMr-expLMl+expLMr)
+ 0.5*(m*(tm + xR - xl) + s*s*t)*erfRMl
- 0.5*(m*(tm + xR - xr) + s*s*t)*erfRMr
- 0.5*(m*(tm + xL - xl) + s*s*t)*erfLMl
+ 0.5*(m*(tm + xL - xr) + s*s*t)*erfLMr);

    // ENERGY FLUX
DATA[2] = (1/sdx)*(expcoeffe*(expRMl-expRMr-expLMl+expLMr)
+ 0.25*(tm*(m2 + 3*s2 + 2*C) + (xR - xl)*(m2+s2+2*C))*erfRMl
- 0.25*(tm*(m2 + 3*s2 + 2*C) + (xR - xr)*(m2+s2+2*C))*erfRMr
- 0.25*(tm*(m2 + 3*s2 + 2*C) + (xL - xl)*(m2+s2+2*C))*erfLMl
+ 0.25*(tm*(m2 + 3*s2 + 2*C) + (xL - xr)*(m2+s2+2*C))*erfLMr);

} // End of TDEFM flux function
```