THE LATTICE BOLTZMANN METHOD FOR COMPUTATIONAL FLUID DYNAMICS APPLICATIONS

by

CHINMAY H. ADHVARYU

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE IN AEROSPACE ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON
August 2008

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Brian H. Dennis for his guidance and encouragement throughout the course of this thesis work. He has given all the freedom during my work in CFD lab at UTA. I would also like to thank Dr. Z. X. Han and Dr. A. Y. Tong for being on my committee.

I would like to take this opportunity to thank my friends at CFD lab for being supportive and kind. I have always enjoyed my discussions with Takahiro Sonoda and James Ruiz. I would like to thank Mr. Rajiv Kumar for sharing his knowledge of CFD and Aerodynamics with me. I would like to thank my friends at CFD lab, Wei Han, Weiya Jin, Ai Ueno, and Rachaneewan Charoenwat. I would like to thank Harsh Shah and Kamal Chauhan for being good friends. I would like to thank Simran Bhalla for being the best friend and always being very supportive.

I would like to thank my parent for supporting me and always encouraging me for this great opportunity of education. I am thankful to my younger sister for being very supportive.

July 11, 2008

ABSTRACT

THE LATTICE BOLTZMANN METHOD FOR
COMPUTATIONAL FLUID DYNAMICS
APPLICATIONS

Chinmay H. Adhvaryu, M.S.

The University of Texas at Arlington, 2008

Supervising Professor: Brian H. Dennis

An analysis of lattice Boltzmann method is presented in this thesis. This analysis contrast with the traditional lattice Boltzmann work, where complex fluids are considered with extended properties. Focus of the current thesis is on the validation and use of lattice Boltzmann method for traditional fluid dynamics.

In first part of the document a historical and theoretical background of lattice Boltzmann method is presented. In this part some conclusions are drawn about the stability and other physical characteristics of the lattice Boltzmann method. Second part deals with the computer implementation of the LBM. It also discusses some aspect about practical usage of this method as a CFD tool. Boundary conditions are discussed in the same sections. Different method for boundary conditions and their numerical characteristics are presented. In the last section two numerical case studies are given for the lattice Boltzmann. Purpose of these numerical studies is to validate lattice Boltzmann method for traditional CFD application. Results of this numerical studies also reveals some facts about the physical aspect of lattice Boltzmann method.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

The present thesis investigates lattice Boltzmann method for the Computational Fluid Dynamics application. Computational Fluid Dynamics, known today as CFD, is defined as the set of methodologies that enable the computer to provide us with a numerical *simulation* of fluid flows. Lattice Boltzmann method is relatively new approach for the hydrodynamic simulations. Unlike traditional CFD method this approach is based on kinetic theory. It is a bottom-up approach. In kinetic theory fluid motion is described at the particle collision level.

Recently much research effort has been invested in using lattice Boltzmann method to solve Navier-Stokes fluids. It has been an active area of research in computational fluid dynamics from last 15 years.

Aim of this thesis is to develop a basic understanding of lattice Boltzmann method as a theoretical and numerical tool for fluid dynamic simulations. Its parallel and differences with traditional CFD methods are discussed. This thesis investigates lattice Boltzmann method from the CFD point of view.

1.1 Content of the Thesis

Chapter 2 of this thesis describes the introduction to CFD and traditional CFD methods. A simple introduction to philosophy of lattice Boltzmann method is given in the same chapter. Historical background and historical background of lattice Boltzmann method is described in chapter 3. A multi-scale Chapmann-Enskog procedure to recover Navier-Stokes equation from discrete Boltzmann equation is described in chapter 3. Computer aspect of lattice Boltzmann are given in chapter 4. In the next chapter boundary conditions of lattice Boltzmann method are discussed. Last chapter, chapter 6 presents the result obtained from lattice Boltzmann code developed during this thesis work. These results are validated against the standard results.

CHAPTER 2

INTRODUCTION TO COMPUTATIONAL FLUID DYNAMICS

2.1 Introduction

Computational Fluid Dynamics, known today as CFD, is defined as the set of methodologies that enable the computer to provide us with a numerical *simulation* of fluid flows. We use the word *simulation* to indicate that we use the computer to solve numerically the laws that govern the movements of fluids, in or around material system.

The core of CFD research area is represented by a partial differential equation (PDE) widely known as the Navier-Stokes equation, which expresses a local conservation law for the momentum in the system. The Navier-Stokes equations are the basic governing equations for a viscous, heat conducting fluid. It is a vector equation obtained by applying Newton's Law of Motion to a fluid element and is also called the *momentum equation*. Solving the Navier-Stokes equation is actually a numerically very challenging task, in spite of the apparent simplicity of the PDE. Much effort is therefore invested in developing numerical approaches to the resolution of this equation, rather than adding even more complexity by additional physical terms. In general these numerical techniques require discretization of the partial differential equation (PDE). In next section we give an overview of most popular methods used for the discretization process.

2.2 Discretization Methods

2.2.1. Finite Difference Method

The finite Difference approximation is the oldest method applied to obtain numerical solution of differential equations. This method is based on the properties of Taylor expansions and on the straightforward application of the definition of derivatives. It is probably the simplest method to apply particularly on the uniform meshes. Limitation of structured grid makes it difficult to apply for problems with complex geometry.

2.2.2 Finite Volume Method

The finite volume method (FVM) The Finite Volume Method (FVM) is one of the most versatile discretization techniques used in CFD. It is most widely applied method in CFD. The reason behind the appeal to the FVM lies in its generality, its conceptual simplicity and its ease of implementation for arbitrary grids, structured as well as unstructured. In this technique integral formulation of the conservation laws is discretized directly in the physical space. FVM is based on cell averaged values, which appear as a most fundamental quantity in conservation equations. This distinguishes the FVM from finite element and finite difference, where the main numerical quantities are the local function values at the mesh points. Thus FVM has the great advantage that the conservative discretization is automatically satisfied, through the direct discretization of the integral form of the conservation laws.

2.2.3 Finite Element Method

The Finite Element Method originated from the field of structural analysis. In FEM original PDEs are multiplied by a test function and integrated over the domain, resulting in the weak formulation of the problem. Infinite dimension subspace which contains the unknown function is then replaced by a subspace of finite dimension, chosen so that the approximate solution consist of piecewise constant polynomial function. Suitable choice of test function leads to a system of algebraic equation. The FEM is based on the definition of function values attached to the nodes of the mesh, where the numerical value of the unknown functions, and eventually their derivatives, will have to be determined.

2.3 The Lattice Boltzmann Method

The Lattice Boltzmann method is relatively new. The Method of lattice Boltzmann equation (LBE) is an innovative numerical method based on kinetic theory to simulate various hydrodynamic systems. The lattice Boltzmann equation was introduced to overcome some serious deficiencies of its historic predecessor: the lattice gas automata. The lattice Boltzmann

equation overcomes two major shortcomings of the lattice gas automata: intrinsic noise and limited values of transport coefficients, both due to the Boolean nature of the LGA method. This method was proposed by McNamara and Zanetti [1]. The fundamental idea of the LBM is to construct simplified kinetic models that incorporate the essential physics of microscopic or mesoscopic processes so that the macroscopic averaged properties obey the desired macroscopic equations. The basic premise for using these simplified kinetic-type methods for macroscopic fluid flows is that the macroscopic dynamics of a fluid is the result of the collective behavior of many microscopic particles in the system and that the macroscopic dynamics is not sensitive to the underlying details in microscopic physics. By developing a simplified version of the kinetic equation, one avoids solving complicated kinetic equations such as the full Boltzmann equation, and one avoids following each particle as in molecular dynamics simulations. Lattice Boltzmann scheme is based on microscopic picture but it focuses on averaged macroscopic behavior of the fluid. That gives the simplicity in implementation, clear physical picture and fully parallel algorithm.

CHAPTER 3

THEORY OF LATTICE BOLTZMANN METHOD

3.1 History of Lattice gas cellular automaton to Lattice Boltzmann Method

Lattice gas cellular automaton models were the harbingers of LBM. The Lattice Boltzmann Equation developed in the wake of Lattice gas cellular automaton method and was generated precisely in response to its initial drawbacks. Simple cellular automaton obeying nothing but conservation laws at microscopic level can be used to simulate hydrodynamics system was first proposed by Frish, Hasslacher, and Pomeau [2] in 1986. The Lattice gas automata is based on the simple evolution rules of *occupation number* n_i on the lattice. *occupation number* n_i is borrowed from statistical mechanics. It represents two states:

$$\begin{aligned} n_i(\vec{x},t)&=0 \quad \textit{particle absence at site } \vec{x}\textit{, time } t\textit{,} \\ n_i(\vec{x},t)&=0 \quad \textit{particle presence at site } \vec{x}\textit{, time } t\textit{.} \end{aligned} \tag{3.1}$$

The collection of occupation numbers $n_i(\vec{x},\ t)$ over the entire lattice defines a 6N-dimensional time dependent boolean field whose evolution takes place in a Boolean phase-space consisting of 2^{6N} discrete states in the case of FHP hexagonal lattice shown in Fig 3.1.

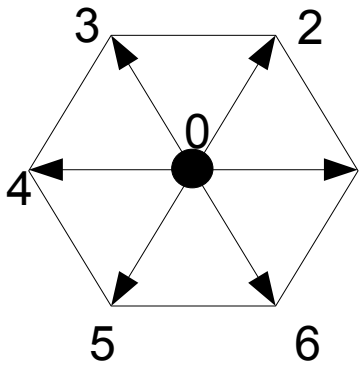

Figure 3.1: FHP Hexagonal lattice

Prescribed evolution rules for this Lattice gas automata should cater two basic mechanism.

- Free streaming
- Collision

Free streaming consists of simple particle transfers from site to site according to set of discrete speeds c_{ia} . Thus, a particle sitting at site $\vec{x}$ at time t with speed c_{ia} will move to site $\vec{x}+c_{ia}$ at time $t+1.$ In equations :

$$n_i(\vec{x}+c_{ia},\ t+1)=n_i(\vec{x},\ t). \tag{3.2}$$

This defines the discrete free streaming operator Δ_i as

$$\Delta_i n_i \equiv n_i(\vec{x}+c_{ia}, t+1)-n_i(\vec{x}, t). \tag{3.3}$$

Collision operator changes occupation number from $n_i \ to \ n_i^{'}$ on the same site. So collision operator $C_i(\underline{n})$*:*

$$n'_i - n_i = C_i(\underline{n}) \tag{3.4}$$

Where $\underline{n} \equiv [n_1,\ n_2\ ,\ .....,\ n_b]$ denotes the set of occupation numbers at a given lattice site. So, the final Lattice gas cellular automata update rule follows

$$\begin{gathered} \Delta_i n_i = C_i \\ \text{or} \\ n_i(\vec{x}+\vec{c}_i, t+1)=n'_i(\vec{x}, t), \end{gathered} \tag{3.5}$$

The equation (3.4) and (3.5) represent the micro dynamic equation for the lattice gas. The Lattice gas cellular automata suffers from a major problem of statistical noise. Like any particle particle method LGCA are exposed to a fair amount of statistical fluctuation [4].

As an alternative approach to Lattice gas automata the Lattice Boltzmann method was developed by statistical averaging the dynamics of former method. This was apparently to cure the statistical noise present in the Lattice gas automata due to the *occupation number* n_i. The earliest LBM was first proposed by G. McNamara and G. Zanetti in 1988, with the explicit intent of curing the statistical noise problem plaguing Lattice gas automata [3]. The basic idea is

simple: Just replace the boolean occupation number n_i with the corresponding ensemble-averaged populations

$$f_i = \langle n_i \rangle, \tag{3.6}$$

where the brackets stand for suitable ensemble averaging. The change in perspective is; instead of tracking single boolean molecules we follow the time history of a collective population representing a microscopic degree of freedom of group of particle.

The whole intension of developing Lattice gas cellular automata for hydrodynamic simulation was to use use underlaying boolean microdynamics. This would allow the idea of "exact computing" since the occupation number is represented by boolean string of 0 and 1. LBE is a direct transcription of LGCA microdynamics with the plane replacement $n_i \rightarrow f_i$, the trade-off is clear: noise is erased because f_i is by definition an averaged, smooth, quantity. On the other hand , being the real numbers, the f_i are no longer amenable to exact Boolean algebra.

It is interesting to note that LBE is an intermediate step in deriving Navier-Stokes from LGCA. Theoretical framework of LBE was based on LGCA models. In other words our basic understanding of LBE was restricted by our knowledge of the statistical mechanics of LGCA. However X. He and L.-S. Luo proved that LBE is a finite difference form of the continuous Boltzmann equation [5,6]. In next section LBE is established on the basis of Boltzmann equation.

3.2 From the Boltzmann Equation to the lattice Boltzmann Equation

Lattice Boltzmann Equation models evolved from the boolean arithmetic based , the lattice gas automata. But the model presented here shows the LBE as a specially discretized form of the continues Boltzmann equation in time and phase space. It shows that the LBE is finite difference form of the continues Boltzmann equation. We start with Boltzmann equation with single relaxation BGK collision operator [7, 8].

3.2.1 Boltzmann Equation and Distribution function

The Boltzmann equation also known as the Boltzmann transport equation, devised by Ludwig Boltzmann, describes the statistical distribution of particles in a fluid. It is an equation for the time evolution of $f_i(\vec{x}, \vec{p}, t)$, the particle distribution function in the phase space. Phase space here can be viewed as a space in which coordinates are given by the position and momentum vectors at the time. The distribution function $f_i(\vec{x}, \vec{p}, t)$, gives the probability of finding a particular molecule with a given position and momentum.

3.2.2 Derivation of Lattice Boltzmann Equation

The Boltzmann BGK equation in the form of ordinary differential equation [7,9].

$$\frac{\partial f}{\partial t} + \xi \cdot \nabla f = \frac{-1}{\lambda}(f - g) \tag{3.7}$$

Where $f_i(\vec{x}, \xi, t)$ is the single particle distribution function, ξ is the microscopic velocity, λ is the relaxation time due to collision, and g is the Boltzmann-Maxwellian distribution function:

$$g \equiv \frac{\rho}{(2\pi R T)^{D/2}} \exp\left[\frac{-(\xi - u)^2}{2RT}\right] \tag{3.8}$$

In which R is the ideal gas constant, D is the dimension of the space, and $\rho, u,$ and T are the microscopic density of mass, velocity and temperature, respectively. The macroscopic variables are the (microscopic velocity) moments of the distribution function of $f_i(\vec{x}, \xi, t)$:

$$\rho = \int f \, d\xi = \int g \, d\xi, \tag{3.9a}$$

$$\rho u = \int \xi f \, d\xi = \int \xi g \, d\xi, \tag{3.9b}$$

$$\epsilon \rho = \frac{1}{2}\int (\xi - u)^2 f \, d\xi = \frac{1}{2}\int (\xi - u)^2 g \, d\xi, \tag{3.9c}$$

In equation (3.9), an assumption of Chapmann-Enskog [9] has been applied:

$$\int h(\xi) f(x,\xi,t)\, d\xi = \int h(\xi) g(x,\xi,t)\, d\xi, \tag{3.10}$$

Where $h(\xi)$ is a linear combination of collisional invariants. Collisional invariant is a microscopic property which does not charge under the effect of collision.

The energy in equation(3.9), can also be written in terms of temperature T:

$$\epsilon = \frac{D_0}{2} RT = \frac{D_0}{2} N_A k_B T \tag{3.11}$$

Where $D_0, N_A,$ and k_B are the number of degrees of freedom of a particle, Avogadro's number, and the Boltzmann constant, respectively.

3.2.2.1 Discretization of Time

Equation (3.7) can be rewritten in the form of an ordinary differential equation:

$$D_t f + \frac{1}{\lambda} f = \frac{1}{\lambda} g \tag{3.12}$$

$D_t = \frac{\partial}{\partial t} + \xi \cdot \nabla$ is the time derivative along the characteristic line ξ. The above equation can be formally integrated over a time step of δ_t:

$$f(x+\xi\,\delta t, \xi, t+\delta t) = e^{(-\delta t/\lambda)} f(x,\xi,t) + \frac{1}{\lambda} e^{(-\delta t/\lambda)} \int_0^{\delta t} e^{t'/\lambda} g(x+\xi t', \xi, t+t')\, dt' \tag{3.13}$$

Assuming that δ_t is small enough and g is smooth enough locally, and neglecting the terms of order of $\Theta(\delta_t^2)$ or smaller in the Taylor expansion of the right hand side of the equation(3.13) we obtain

$$f(x+\xi\,\delta t, \xi, t+\delta t) - f(x,\xi,t) = \frac{-1}{\tau}[f(x,\xi,t) - g(x,\xi,t)], \tag{3.14}$$

Where $\tau = \lambda / \delta_t$ is the dimensionless relaxation time.

3.2.2.2 Equilibrium Function g

In the lattice Boltzmann equation, the equilibrium distribution function is obtained by a truncated small velocity expansion of often called low-Mach-number approximation [10].

$$
\begin{aligned}
g &= \frac{\rho}{(2\pi RT)^{D/2}} \exp\left[\frac{-(\xi-u)^2}{2RT}\right] \\
&= \frac{\rho}{(2\pi RT)^{D/2}} \exp\left[\frac{-\xi^2}{2\mathrm{RT}}\right] \exp\left[\frac{\xi\cdot u}{RT} - \frac{u^2}{2\mathrm{RT}}\right] \\
&= \frac{\rho}{(2\pi RT)^{D/2}} \exp\left[\frac{-\xi^2}{2\mathrm{RT}}\right] \times \left\{1 + \frac{(\xi\cdot u)}{RT} + \frac{(\xi\cdot u)^2}{2(RT)^2} - \frac{u^2}{2\mathrm{RT}}\right\} + \Theta(u^3)
\end{aligned}
$$

For convenience, the following notation for the equilibrium distribution function with truncated small velocity expansion shall be used from now on.

$$
f^{(eq)} = \frac{\rho}{(2\pi RT)^{D/2}} \exp\left[\frac{-\xi^2}{2\mathrm{RT}}\right] \times \left\{1 + \frac{(\xi\cdot u)}{RT} + \frac{(\xi\cdot u)^2}{2(RT)^2} - \frac{u^2}{2\mathrm{RT}}\right\} + \Theta(u^3) \tag{3.15}
$$

Although $f^{(eq)}$ only retains the terms up to $\Theta(u^2)$, it is trivial to maintain high order terms of u in the above expansion if necessary.

3.2.2.3 Discretization of phase space

For the discretization of phase space, we need two consideration. First, the discretization of momentum space is coupled to that of configuration space such that a lattice structure is obtained. This is a special characteristic of lattice Boltzmann equation. Second quadrature must be accurate enough so that not only the conservation constraints are preserved exactly, but also the necessary symmetries required by the Navier-Stokes equations are retained.

To derive the Navier-Stokes equation from the Boltzmann equation using Chapman-Enskog analysis, first two order approximation of the distribution function should be considered, i.e. $f^{(eq)}$ and $f^{(1)}$ [9]. Calculating the hydrodynamic moments of equation (3.9) is equivalent to evaluating the following integral in general:

$$I=\int \psi(\xi) f^{(eq)} d\xi$$
$$=\frac{\rho}{(2\pi RT)^{D/2}} \int \psi(\xi) \exp\left[\frac{-\xi^2}{2\mathrm{RT}}\right] \times \left\{1+\frac{(\xi\cdot u)}{RT}+\frac{(\xi\cdot u)^2}{2(RT)^2}-\frac{u^2}{2\mathrm{RT}}\right\} d\xi \tag{3.16}$$

Where $\psi(\xi)$ is a polynomial of ξ . The above integral has following structure:

$$\int e^{-x^2} \psi(x)\, dx\,,$$

which can be calculated numerically with Gaussian-type quadrature. Our objective is to use quadrature to evaluate hydrodynamic moments ρ, u, and T .

To recover the two dimensional 9-velocity LBE model on square lattice space, if we use following values for $\psi(\xi)$:

$$\psi_{m,n}(\xi)=\xi_x^m \xi_y^n,$$

where ξ_x and ξ_y are the x and y components of ξ . The integration of equation (3.16) using these values gives us the following equation for the equilibrium distribution function for two dimensional, 9- velocity LBE model:

$$f_\alpha^{(eq)}=w_\alpha \rho \left\{1+\frac{3(e_\alpha\cdot u)}{c^2}+\frac{9(e_\alpha\cdot u)^2}{2c^4}-\frac{3u^2}{2c^2}\right\} \tag{3.17}$$

Where weights,

$$\begin{aligned} &w_\alpha=4/9, && \alpha=0 \\ &w_\alpha=1/9, && \alpha=1,2,3,4 \\ &w_\alpha=1/36, && \alpha=5,6,7,8 \end{aligned} \tag{3.18}$$

and microscopic velocities

$$\begin{aligned} &e_\alpha=(0,\,0), && && \alpha=0 \\ &e_\alpha=(\cos\theta_\alpha,\,\sin\theta_\alpha)c && \theta_\alpha=(\alpha-1)\pi/2 && \alpha=1,2,3,4 \\ &e_\alpha=\sqrt{2}(\cos\theta_\alpha,\,\sin\theta_\alpha)c && \theta_\alpha=(\alpha-1)\pi/2+\pi/4 && \alpha=5,6,7,8 \end{aligned} \tag{3.19}$$

Here is $c=\sqrt{3\mathrm{RT}}=\delta_x/\delta_t$ or $RT=c_s^2=c^2/3$ where c_s is the sound speed of the model. Here the phase space is discretized in to a square lattice space with a lattice constant

$\delta_x = \sqrt{3\mathrm{RT}}\,\delta_t$. It should be noted that temperature *T* has no physical significance since we are dealing with isothermal model.

Following the same procedure different configuration for the phase space can be achieved. For example two-dimensional 6-velocities and 7-velocities triangular lattice can be constructed. Similar way three dimensional 27 velocities square lattice model can be constructed.

3.3 Lattice Boltzmann Equation to Navier Stokes equation

Application of Taylor series expansion of the lattice Boltzmann equation followed by the a Chapmann-Enskog expansion results in the typical hierarchy of equation; Euler, Navier-Stokes, Burnett, etc. By selecting the appropriate number of speeds and the appropriate form of the equilibrium distribution function, one may match the equations that result from the LB method with those of the traditional kinetic theory to the desired level. Higher order terms that are not matched represent behavior of the lattice gas that differs from a Maxwellian gas. The Chapmann-Enskog theory yields correct behavior to the Euler level but the Navier-Stokes level is valid only in incompressible limit. In other terms, the incorrect terms become small as the square of the Mach number becomes small. Here we should a make note that in these models only mass and momentum are conserved. So we are discussing isothermal incompressible applications. Complete energy conserving models can yield the correct form of the compressible continuity, momentum and energy equation. We limit our discussion to isothermal incompressible LB schemes with low Knudsen number and Mach number limit.

This section provides a description of the Chapmann-Enskog expansion of the discrete-velocity Boltzmann equation and application of the lattice Boltzmann discretization. We are using BGK collision operator where τ is relaxation parameter, which is inversely proportional to density. With this condition discrete velocity Boltzmann equation becomes :

$$\frac{\partial f_i}{\partial t} + e_i \cdot \nabla f_i = \frac{-1}{\tau}\left(f_i - f_i^{(eq)}\right) \tag{3.20}$$

Macroscopic flow variables are defined by moments of distribution function:

Mass,

$$\rho \equiv \sum_i f_i; \tag{3.21}$$

Momentum,

$$\rho u \equiv \sum_i f_i \cdot e_i. \tag{3.22}$$

The Chapmann-Enskog procedure is based on a double expansion in the smallness parameter of both dependent $f_i(\vec{x}, t)$ and independent space time variable. This method requires the expansion of the distribution function in the small parameter such that $f_i \equiv f^0 + \epsilon f^1 \ldots$, where subscript 0 denotes local equilibrium $f^0 = f^{eq}$ and subscript 1 departure from this local equilibrium. So for Chapmann-Enskog procedure other differential operators will be evaluated like following:

$$\begin{aligned} f_i &= f_i^0 + \epsilon f_i^1 + \epsilon^2 f_i^2 + \ldots, \\ \frac{\partial}{\partial t} &= \epsilon \frac{\partial}{\partial t_1} + \epsilon^2 \frac{\partial}{\partial t_2} + \ldots, \\ \frac{\partial}{\partial x} &= \epsilon \frac{\partial}{\partial x_1}, \end{aligned} \tag{3.23}$$

It is already shown that to first order in ϵ we have the following continuity and momentum equation [2].

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho u) = 0, \tag{3.24}$$

$$\frac{\partial}{\partial t}(\rho u) + \nabla \cdot (\Pi) = 0, \tag{3.25}$$

where $\Pi = \Pi^{(0)} + \Pi(1); \Pi^{(0)}$ and $\Pi^{(1)}$ are the equilibrium and non-equilibrium momentum flux tensors, respectively, and are defines as,

$$\Pi^{(l)}_{\alpha\beta}=\sum_i e_{i\alpha}e_{i\beta}f^{(l)}_i \tag{3.26}$$

with *l*=0,1.

If we use value of $f^{(0)}=f^{eq}$ from equation (3.17) and other values in equation (3.18-3.19). The above yields ,

$$\Pi^{(0)}_{\alpha\beta}=p\,\delta_{\alpha\beta}+\rho\,u_\alpha u_\beta \tag{3.27}$$

where *p* is the pressure, $p=\rho/3,$ the equation of state for an ideal gas with constant sound speed, $C_s=1/\sqrt{3}.$ First term on the right hand side of the equation (3.27) is isotopic part of this tensor, with pressure term. Second term on the right hand side of the equation (3.27) gives the Galilean invariant convective term in momentum equation.

Using the standard Chapmann-Enskog procedure, $\Pi^{(1)}$ can be calculated,

$$\Pi^{(1)}_{\alpha\beta}=-\nu(\nabla_\alpha(\rho\,u_\beta)+\nabla_\beta(\rho\,u_\alpha)) \tag{3.28}$$

The resulting momentum equation is,

$$\rho\frac{\partial u_\alpha}{\partial t}+\nabla_\beta\cdot(\rho\,u_\alpha u_\beta)=-\nabla_\alpha p+\nu\,\nabla_\beta\cdot(\nabla_\alpha\rho\,u_\beta+\nabla_\beta\rho\,u_\alpha)$$

or

$$\begin{aligned}\rho\frac{\partial u_\alpha}{\partial t}+\rho\,u_\beta\frac{\partial u_\beta}{\partial x_\beta}&\\ =-\frac{\partial p}{\partial x_\alpha}+\frac{\partial}{\partial x_\beta}\left(\frac{\lambda}{\rho}\left(\frac{\partial\rho\,u_\gamma}{\partial x_\gamma}\delta_{\alpha\beta}+u_\alpha\frac{\partial\rho}{\partial x_\beta}+u_\beta\frac{\partial\rho}{\partial x_\alpha}\right)\right)&\\ +\frac{\partial}{\partial x_\beta}\left(\mu\left(\frac{\partial u_\beta}{\partial x_\alpha}+\frac{\partial u_\beta}{\partial x_\beta}\right)\right),&\end{aligned} \tag{3.29}$$

where $\nu=\frac{\tau}{3}$ is the kinematic viscosity in first equation. Note that these equations are not standard Navier-Stokes equations. In the second equation, the second term on the right hand side has derivatives of density. If these gradients of density are negligible, then this discrete lattice Boltzmann equation should behave like Navier-Stokes equations. If we make an assumption of low Mach number then Navier-Stokes equation can be recovered in nearly

incompressible limit i.e $M = u/C_s \ll 1$ and $u' \sim \rho' \sim p' \sim \Theta(M^2)$. Where primes quantities denotes fluctuations.

3.4 Lattice Boltzmann BGK model

In this section all the possible lattice structures are presented and there constants. Lattice Boltzmann equation with single relaxation time BGK collision operator can be written as following:

$$f_i(x+e_i, t+1) - f_i(x,t) = -\frac{1}{\tau}\left(f_i(x,t) - f_i^{eq}(x,t)\right) \tag{3.30}$$

where, $$f_i^{(eq)} = w_i \rho \left\{1 + \frac{3(e_i \cdot u)}{c^2} + \frac{9(e_i \cdot u)^2}{2c^4} - \frac{3u^2}{2c^2}\right\} \tag{3.31}$$

With hydrodynamic moments are,

$$\rho = \sum_i^q f_i, \quad \text{and} \quad \rho u = \sum_i^q f_i e_i. \tag{3.32}$$

It should be noted here that since equation (3.30) has a Lagrangian nature in space discretization, and because of the implicit nature in the time, the discretization error in equation (3.30) has a special form which can be included in viscosity terms. It turns out that equation (3.30) is a secind order method both in space and time, if replace ν by,

$$\nu = \frac{2\tau - 1}{6} \tag{3.33}$$

i.e., the truncation error can be absorbed into the viscosity to increase the accuracy of lattice Boltzmann method.

There are different types of lattice structures. With substituting corresponding parameters in equation (3.30 – 3.33) different lattices can be constructed. These parameters are shown in table (3.1)

Table 3.1 Parameters of some DnQm BGK lattices

Lattice Name	C_s^2	Weight w_i
D1Q3	1/3	4/6 1/6
D1Q5	1	6/12 2/12 1/12
D2Q9	1/3	16/36 4/36 1/36
D3Q15	1/3	16/72 8/72 1/72

3.5 Summary

In this chapter theoretical development of lattice Boltzmann is discussed. Lattice Boltzmann equations are hyperbolic subset of the Navier-Stokes equations. It is a simple advection and diffusion equation. As opposed to nonlinear $u \nabla u$ convective term in Navier Stokes equation, LBE has a linear streaming operator.

CHAPTER 4

COMPUTER IMPLEMENTATION OF LATTICE BOLTZMANN METHOD

4.1 Introduction

Focus of this chapter is use of lattice Boltzmann as a computational fluid dynamics tool. Having discussed theory of lattice Boltzmann equation in previous chapters, mathematical implementation of this method is discussed in this chapter. Lattice Boltzmann equation with single relaxation time BGK approximation is following,

$$f_i(x+e_i\delta t, t+\delta t)-f_i(x_i,t)=-\frac{1}{\tau}\left[f_i(x,t)-f_i^{eq}(x,t)\right] \quad (4.1)$$

where τ is relaxation time and f^{eq} is equilibrium distribution function. It is a general practice to divide equation (4.1) in to two steps: collision and streaming

collision step : $f_i^t(x,t+\delta t)=f_i(x,t)-\frac{1}{\tau}\left[f_i(x_i,t)-f_i^{eq}(x,t)\right]$ (4.2a)

streaming step: $f_i(x+e_i\delta t, t+\delta t)=f^t(x,t+\delta t),$ (4.2b)

Where $f_i^t(x,t+\delta t)$ is post collision distribution function. It should be noted that with such a splitting in the computational procedure, there is no need to store both $f_i^t(x,t+\delta t)$ and $f_i(x,t)$

4.2 Lattice Boltzmann Algorithm and its Computer Implementation

4.2.1 Lattice Boltzmann algorithm and accuracy

In this section general computer implementation of lattice Boltzmann method is explained. In this section computer coeds are given with the appropriate part. The algorithm consists of iterative procedure using equation (4.2).

In a numerical simulation, the distribution functions f_i are stored in q single or double precision floating point variables. This means that additionally to the precision trade off

due to the discretization of the 6-D phase space and the time, one introduces a numerical error due to the fixed-width representation of real numbers. In its current form, the model has actually a numerical deficiency that may keep the latter numerical error needlessly high. This is because we store the sum of both large and small numbers in the same variable. This is because we store the sum of both large and small numbers in the same variable. The density ρ can namely be split in two contributions: $\rho_0+\rho^{dyn}; \rho_0$ is the constant density of the incompressible flow, and ρ^{dyn} , a measure of the pressure variations, scales at at a second order with respect to the Mach number of the system. For low Mach-number flows, the contribution ρ^{dyn} can therefore not be stored in the variable with sufficient accuracy. The same is true for the variables f_i^{eq} that contain contributions at several orders of magnitude. This situation can be improved by simulating the dynamics of $f_i^{dyn}=f_i-\rho_0 t_i$ instead of f_i . They have the same dynamics as the f_i :

$$f_i^{dyn}(x+e_i\delta t, t+\delta t)-f_i^{dyn}(x,t)=-\omega(f_i^{dyn}(x,t)-f_i^{dyn,eq}(x,t)) \tag{4.3}$$

where ,

$$f_i^{dyn,eq}=f_i^{eq}-\rho_0 w_i=\rho^{dyn}w_i+\rho w_i\left(3\vec{e}_i\cdot\vec{u}+\frac{9}{2}(e_i\cdot\vec{u})^2-\frac{3}{2}|\vec{u}|^2\right), \tag{4.4}$$

$$\rho^{dyn}=\sum_i^9 f_i^{dyn},\quad \rho=\rho^{dyn}+\rho_0 \text{ and } \rho\vec{u}=\sum_i^9 \vec{e}_i\cdot f_i^{dyn} \tag{4.5}$$

In a simulation, the value of ρ_0 is fixed when you implement the initial and the boundary conditions. In the following, we will always work with f_i^{dyn} , but we skip the superscript (dyn). Finally, we implement the dynamics in a slightly different form in which the following discussion of the numerical implementation is simplified. For this, the f_i are renamed to f_i^{in} , and we introduce the temporary variables f_i^{out} . The dynamics consists then of

a collision step: $f_i^{out} = (1-\omega) f_i^{\text{in}} + \omega f_i^{eq}$

a streaming step: $f_i^{\text{in}}(x + e_i \delta t, t + \delta t) = f_i^{out}(x, t)$

It is interesting to note that the collision step is purely local, that means, it involves no transfer of information between lattice cells. The streaming step on the other hand involves no computations, but performs only copies between adjacent cells. This clear distinction helps writing simple codes with advanced features such as parallel program execution.

4.2.2 Computer implementation

Simulations of fluid dynamics are resource consuming, both in the usage of CPU time and memory storage. It is therefore crucial to design the code in such a manner as to allocate as little memory as possible, and to arrange the variables in memory in an appropriate way. During accesses to the central memory, data chunks are regularly pre fetched into the cache, where they should be utilized as extensively as possible, thus avoiding new accesses to the central memory. A good result is for example obtained when you are able to traverse linearly through contiguous memory chunks during the execution of the program. For this matter distribution functions are stored in multidimensional arrays as shown below.

```
double f[lx][ly][q];        // lx * ly * q double-precision matrix (2D case)
double f[lx][ly][lz][q];    // lx * ly * lz * q double-precision matrix (3D case)
```

Arrays to store velocity and density can be avoided in order to save memory. Those quantities can be computed locally and can be stored in double precision variable at each collision step. The lattice constants are stored in a constant, globally accessible array, as in the following example for the D2Q9 lattice:

```
// lattice weights
static const double w[9] = { 4./9., 1./36., 1./9., 1./36., 1./9., 1./36., 1./9., 1./36., 1./9. };

// lattice velocities
static const int c[9][2] = {
{0,0},{-1,1}, {-1,0}, {-1,-1}, {0,-1},{1,-1}, {1,0}, {1,1}, {0,1}};
```

They are stored as constants as to avoid any possible change during the execution.

4.2.2.1 Collision step

In order to carry out collision step, we need equilibrium distribution function. To calculate value of distribution function we need values of density ρ and velocity *u* calculated from distribution functions from the last time step. We can calculate density ρ locally, making a function as following. We can do the same thing to calculate velocity *u* locally.

```
// Compute the local particle density on a lattice site
void computeRho(double f[q], double* rho) {
int iF;
*rho = 0.;
for (iF=0; iF<q; ++iF) {
                        rho += f[iF];
                }
}

// Compute the local flow velocity, and its norm-square on a lattice site
void computeU(double f[q], double rho, double u[d], double* uSqr) {
Int iD, iF;
*uSqr = 0.;
for(iD=0; iD<d; ++iD) {
        u[iD] = 0.;
        for (iF=0; iF<q; ++iF) {
                        u[iD] += f[iF] * c[iF][iD];
        }
        u[iD] /= rho;
        *uSqr += u[iD]*u[iD];
        }
}
```

Once density and velocity is determined, equilibrium distribution function can be calculated in a function as following.

```
// Compute local equilibrium term from rho and u
double fEq(int iF, double rho, double u[d], double uSqr) {
int iD;
double c_u = 0.; // scalar product between c_{iF} and u
for (iD=0; iD<d; ++iD) {
        c_u += c[iPop][iD] * u[iD];
}
// remember that we are working with f^{dyn} = f-rho0
return rho*w[iPop] +
(rho+rho0)*w[iPop]*(3.*c_u + 4.5*c_u*c_u - 1.5*uSqr);
}
```

Once we have value of equilibrium distribution function we can carry out BGK collision step.

```
// Apply the BGK collision step on a lattice site
void bgkCollideSite(double f[q]) {
double rho, u[d], uSqr;
int iF;
computeRho(f, &rho);
computeU(f, rho, u, &uSqr);
for (iF=0; iF<q; ++iF) {
        f[iF] *= (1.-omega);
        f[iF] += omega * fEq(f, iF, rho, u, uSqr);
        }
}
```

4.2.2.2 Streaming step

The implementation of the streaming step contains a subtlety. Suppose the values of f^{i} and f^{out} were stored in two different arrays. Then the streaming step, applied to all lattice sites would be straightforward:

```
// streaming step, version using a temporary array
void stream2D(double fIn[lx][ly][q], double fOut[lx][ly][q]) {
int iX, iY, iF, nextX, nextY;
for (iX=0; iX<lx; ++iX) {
        for (iY=0; iY<ly; ++iY) {
                for (iF=0; iF<q; ++iF) {
                        nextX = iX + c[iF][0];
                        nextY = iY + c[iF][1];
                        if (nextX>=0 && nextY>=0 && nextX<lx && nextY<ly) {
                        fIn[nextX][nextY][iF] = fOut[iX][iY][iF];
                        }
                }
        }
}
}
```

This does not work though when fIn and fOut are identical, because during the loop traversal, one overwrites values that are still needed further ahead. There are several possible solutions to that problem, the one shown here proposes to store the values of f_{in} and f_{out} in the same collection of q population functions, but at a different index. For this, we define an opposite opp(i) of an index $i \epsilon [0 \cdots q-1]$ in such a way that $c_{opp}(i) = -c_{i.}$. That means that all the f's must be swapped after collision:

```
// swap two double precision values
void swap(double* val1, double* val2) {
double tmp = *val1;
*val1 = *val2;
*val2 = tmp;
}
// apply this step after the collision step, to store fOut in the opposite
// location of fIn
void swapAfterCollision(double f[q]) {
int iF;
for (iF=1; iF<q/2; ++iF) {
        swap(&f[iF], &f[iF+q/2]);
        }
}
```

So streaming step using a single array:

```
void stream2D(double f[lx][ly][q]) {
int iX, iY, iF, nextX, nextY;
for (iX=0; iX<lx; ++iX) {
        for (iY=0; iY<ly; ++iY) {
                for (iF=1; iF<q/2; ++iF) {
                        nextX = iX + c[iF][0];
                        nextY = iY + c[iF][1];
                        if (nextX>=0 && nextY>=0 && nextX<lx && nextY<ly) {
                        swap(&f[iX][iY][iF+q/2], &f[nextX][nextY][iF]);
                        }
                }
        }
   }
}
```

4.2.2.3 Final code

```
// Execute one BGK iteration step (without boundary conditions)
void bgkIteration(double f[lx][ly][q]) {
int iX, iY;
// Apply collision over entire space
for (iX=0; iX<lx; ++iX) {
        for (iY=0; iY<ly; ++iY) {
                bgkCollideSite(f[iX][iY]);
                swapAfterCollision(f[iX][iY]);
        }
}
// Apply streaming over entire space
stream2D(f);
}
```

Final code using above mentioned procedure will look like this. It is worth noting here that no boundary condition has been implemented. In a case with boundary conditions, collision step is the place to introduce it.

4.3 Choice of units in simulation setup

When simulating real life fluid dynamics, all the variables and simulation parameters are represented in the form of physical unit. On the other hand lattice Boltzmann models express everything in terms of lattice units. To use lattice Boltzmann method effectively for CFD, one should translate physical unit to lattice unit and vice versa. In this section a simple procedure is discussed to achieve this translation.

The approach presented here consists of two steps. A physical system is first converted into a dimensionless one, which is independent of the original physical scales, but also independent of simulation parameters. In a second step, the dimensionless system is converted into a discrete simulation. The correspondence between these three systems (the physical one (P), the dimensionless one (D), and the discrete one (LB)) is made through dimensionless, or scale-independent numbers. The solutions to the incompressible Navier-Stokes equations for example depend only on one dimensionless parameter, which is the Reynolds number (Re). Thus, the three systems (P), (D), and (LB) are defined so as to have the same Reynolds number. The transition from (P) to (D) is made through the choice of a characteristic length scale l_0 and time scale t_0 , and the transition from (D) to (LB) through the choice of a discrete space step x and time step t.

For example we want to simulate flow in a cavity. For this, a length scale l_0 and a time scale t_0 are introduced which are representative for the flow configuration. The length l_0 could length or width of the cavity, and t_0 could be the time needed by a passive scalar in the fluid to travel a distance l_0 . The physical variables such as the time t_p and the position vector x_p , are replaced by their dimensionless counterparts:

$$t_d = \frac{t_p}{t_{0,p}} \quad \text{and} \quad x_d = \frac{x_p}{l_{0,p}} \tag{4.6}$$

where the dimensionless Reynolds number has been defined as

$$Re = \frac{l_0^2}{t_o \nu} \tag{4.7}$$

Note that by expressing reference variables in the dimensionless system, one finds that $l_{0,d} = 1$ and $t_{0,d} = 1$. One may therefore consider the dimensionless system as the system in which l_0 and t_0 are unity. It is worth mentioning that the viscosity in the dimensionless system is $\nu_d = 1/Re$. Now dimensionless space can be discretized. The discrete space interval δx is defined as the reference length divided by the number of cells N used to discretize this length. In the same way, δt is defined as the reference time divided by the number of iteration steps Niter needed to reach this time.

$$\begin{aligned} \delta x &= \frac{l_{d,0}}{N} \\ \delta t &= \frac{t_{d,0}}{N_{iter}} \end{aligned} \tag{4.8}$$

Other variables, such as velocity and viscosity, are easily converted between (D) and (LB) through a dimensionless analysis:

$$\begin{aligned} u_d &= \frac{\delta x}{\delta t} u_{lb} \\ \nu_l &= \frac{1}{Re} = \frac{\delta_x^2}{\delta t} \nu_{lb} \end{aligned} \tag{4.9}$$

So, if we take $u_0 = l_0/t_o$ we find

$$u_{0,d} = 1 \quad \text{and} \quad u_{0,lb} = \delta t / \delta x \tag{4.10}$$

There is no straightforward intuitive way to chose δt . In other numerical schemes than LB, δt is often linked to δx by stability considerations. In explicit time-stepping schemes, it is common to use the relation $\delta t \sim \delta x^2$ to keep the model numerically stable.

4.4 Differences between LBE and other CFD methods

As a CFD tool lattice Boltzmann method differs from other CFD methods in various aspects. There are two major differences. They are mentioned below.

4.4.1 Convection

One major difference is absence of convective non-linear term in lattice Boltzmann method. Other Navier-Stokes based CFD methods has to inevitably treat the non linear convective term $u \cdot \nabla u;$. In lattice Boltzmann this nonlinear convection simply becomes advection in linear streaming step. Non linearity of the convection is absorbed in to the collision term.

4.4.2 Pressure

In traditional CFD methods, for the incompressible Navier-Stokes solver, pressure is calculated by solving Poisson's equation. While in the case of lattice Boltzmann method pressure is determined from equation of state.

CHAPTER 5

BOUNDARY CONDITIONS

5.1 Introduction

Boundary conditions and Initial conditions are essential for any computational fluid dynamic methods. For traditional CFD methods, for every boundary and initial conditions Navier-Stokes equations has a unique solution. There are different types of boundary conditions depending on the problem description and given data.

One major difficulty with implementation of boundary conditions in LB is, one has to translate given information from macroscopic variables to distribution function f_i , since it is the only variable to be evaluated in lattice Boltzmann method. In this section various types of boundary conditions are discussed and how to implement them in terms of lattice Boltzmann equations.

5.2 Boundary Condition

5.2.1 Dirichlet Boundary Conditions

Dirichlet class of boundary conditions are probably the simplest types of boundary conditions in terms of mathematics and implementation. These kind of boundary conditions specifies value of the solution or other variable on the boundary. In this section we discuss various methods to implement Dirichlet boundary condition. Following discussion is considering D2Q9 square lattice.

5.2.1.1.Velocity Boundaries

For example consider a boundary node in figure 5.1. Boundary is aligned with x direction. Distribution functions f_4, f_7, f_8 are inside the wall. While f_3, f_0, f_1 are on the wall. After streaming $f_0, f_{1,} f_{3,} f_4, f_{7,} f_8$ are known. Suppose that x and y

components of velocity u_x and u_y are specified on the wall. We can use hydrodynamic moments to determine the values of $f_{2,}f_{5,}f_6$ and ρ .

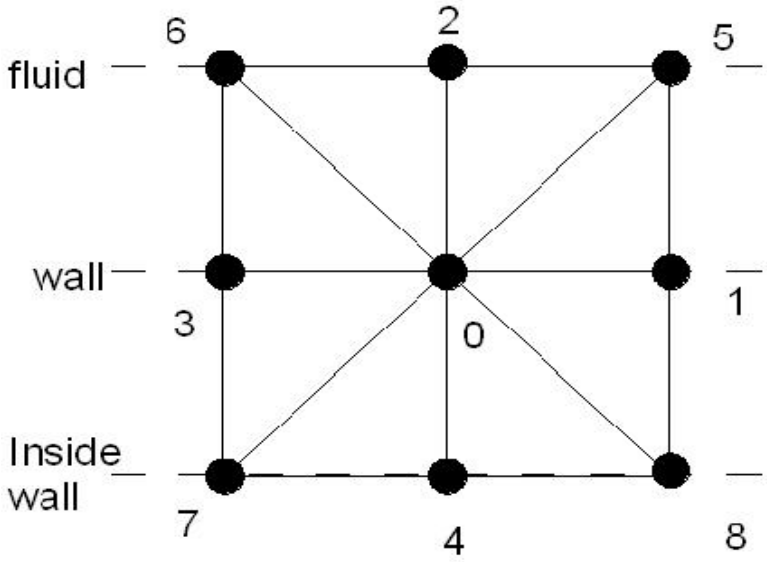

Figure 5.1 : Boundary node aligned with X direction

We can express fluid density in following manner

$$\rho=\sum_i^9 f_i$$
$$So,\ f_2+f_5+f_6=\rho-(f_0+f_1+f_3+f_4+f_7+f_8), \quad (5.1)$$

$$\rho u=\sum_i^9 f_i e_i$$
$$So,\ f_5-f_6=\rho u_x-(f_1-f_3-f_7+f_8) \quad (5.2)$$

$$\text{and } \rho u_y=(f_2+f_5+f_6)-(f_4+f_7+f_8),$$
$$So,\ (f_2+f_5+f_6)=\rho u_y+(f_4+f_7+f_8). \quad (5.3)$$

Comparing equation (4.1) and (4.3) we get

$$\rho=\frac{1}{1-u_y}\left[f_0+f_1+f_3+2(f_4+f_7+f_8)\right] \quad (5.4)$$

However $f_{2,}f_5$ and f_6 remain undetermined. If we assume that bounceback rule is still holds true then,

$$f_2 = f_4 + \frac{2}{3}\rho u_y,$$
$$f_5 = f_7 - \frac{1}{2}(f_1 - f_3) + \frac{1}{2}\rho u_x + \frac{1}{6}\rho u_y, \quad (5.5)$$
$$f_6 = f_8 + \frac{1}{2}(f_1 - f_3) - \frac{1}{2}(\rho u_x) + \frac{1}{6}\rho u_{y.}$$

The collision is applied to all boundary nodes. The same boundary conditions can be developed for other walls as well. The above described scheme is also known as Zou and He boundary conditions suggesting the name of the original author who proposed this idea [14].

The only difficulty with this approach is orientation of the wall. All boundaries can not be treated with the same equations. While implementation of lattice Boltzmann method one has to take care of the different walls using different equations. To overcome this shortcoming there have been some new methods based on extrapolation principal are proposed [12]. These methods will be briefly discussed later in this chapter.

5.2.1.2 Pressure boundaries

Pressure boundary conditions are essentially specify density on the boundaries in lattice Boltzmann methods. In lattice Boltzmann method pressure is defined by the equation of state $p = \rho / C_s^2$. Where $C_s = \sqrt{RT}$ is speed of sound. Consider the figure 5.2

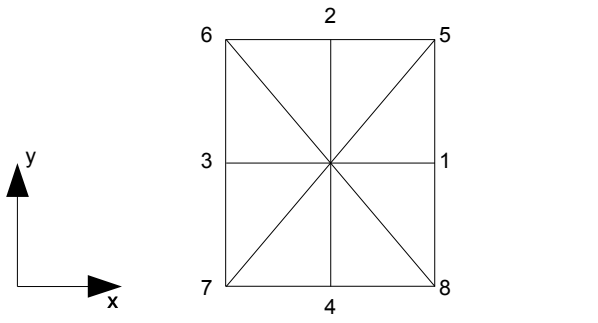

Figure 5.2: D2Q9 lattice on inlet

Pressure is to be specified on the flow boundary, in this case it is inlet. Pressure should be specified along the y direction on the inlet, and that u_y is also specified. After streaming, $f_{2,}f_{3,}f_{4,}f_{6,}f_{7}$ are known, in addition to $\rho=\rho_{\text{in}}$, $u_y=0.$ We need to determine u_x and $f_{1,}f_{5,}f_{8}$ as follows.

$$f_1+f_5+f_8=\rho_{\text{in}}-(f_0+f_2+f_3+f_4+f_6+f_7), \tag{5.6}$$

$$f_1+f_5+f_8=\rho_{\text{in}}u_x+(f_3+f_6+f_7) \tag{5.7}$$

$$f_5-f_8=-f_2+f_4-f_6+f_7 \tag{5.8}$$

Comparing equation (5.6) and (5.7) we get,

$$u_x=1-\frac{\left[f_0+f_2+f_4+2(f_3+f_6+f_7)\right]}{\rho_{\text{in}}} \tag{5.9}$$

If we use bounceback rule for the non-equilibrium part of the particle distribution function, normal to the inlet, to find $f_1-f_1^{(eq)}=f_3-f_3^{(eq)}$. With f_1 known, $f_{5,}f_8$ are obtained by the remaining two questions:

$$\begin{aligned} f_1&=f_3+\frac{2}{3}\rho_{\text{in}}u_x \\ f_5&=f_7-\frac{1}{2}(f_2-f_4)+\frac{1}{6}\rho_{\text{in}}u_x, \\ f_8&=f_6+\frac{1}{2}(f_2-f_4)+\frac{1}{6}\rho_{\text{in}}u_{x.} \end{aligned} \tag{5.10}$$

The corner node at the inlet needs some special treatment. Take the bottom node at the inlet as an example. After streaming, $f_{3,}f_{4,}f_{7}$ are known, ρ is specified, and $u_x=u_y=0.$ We need to determine $f_{1,}f_{2,}f_{5,}f_{6,}f_{8.}$ We use the bounceback rule for the non-equilibrium particle distribution function normal to the inlet and the boundary to find

$$\begin{aligned} f_1&=f_3+(f_1^{(eq)}-f_3^{(eq)})=f_{3,} \\ f_2&=f_4+(f_2^{(eq)}-f_4^{(eq)})=f_4 \end{aligned}$$

(5.11)

using $f_{1,}f_2$ in equations (5.7) and (5.8), we find

$$f_5=f_{7,}f_6=f_8=\frac{1}{2}\left[\rho_{\text{in}}-(f_0+f_1+f_2+f_3+f_4+f_5+f_7)\right] \tag{5.12}$$

We can repeat the same procedure for top inlet node and outlet nodes.

The specification of velocities u_x, u_y at a flow boundary is actually equivalent to a velocity wall boundary condition. The effect of specifying velocity at the inlet is similar to specifying pressure at the inlet, since both conditions will generate a density difference in the flow.

5.2.2. Curved and other complex boundaries

Boundary condition implementation methods presented in the last sections are effective for the simple geometries and boundaries. They yield second order accuracy [13]. However these schemes can not be extended for complex geometries and other types of complex boundaries such as porous media, moving boundaries.

In this section an extrapolation scheme is discussed for curved boundaries. This section is largely based on reference [13]. This scheme can be extended for other more complex boundaries as well. Consider a curved boundary shown in figure 5.3. Here we are considering velocity boundary condition.

As shown in figure 5.3. , the link between the fluid node X_f and the wall node X_w interacts the physical boundary at X_b, and $X_f=X_w+e_i\delta$. The fraction of the intersected link in the fluid region is $\Delta=(\|X_f-X_b\|)/(\|X_f-X_w\|)$. This scheme is based on specifying unknown distribution after the streaming with, decomposing it in to two parts just as Chapmann-Enskog analysis: where $f_i^{(eq)}$ and $f_i^{(ne)}$ are the equilibrium and unsteady part of the distribution function respectively. Guo et al. 2002 suggested a fictitious value of equilibrium part as following [13].

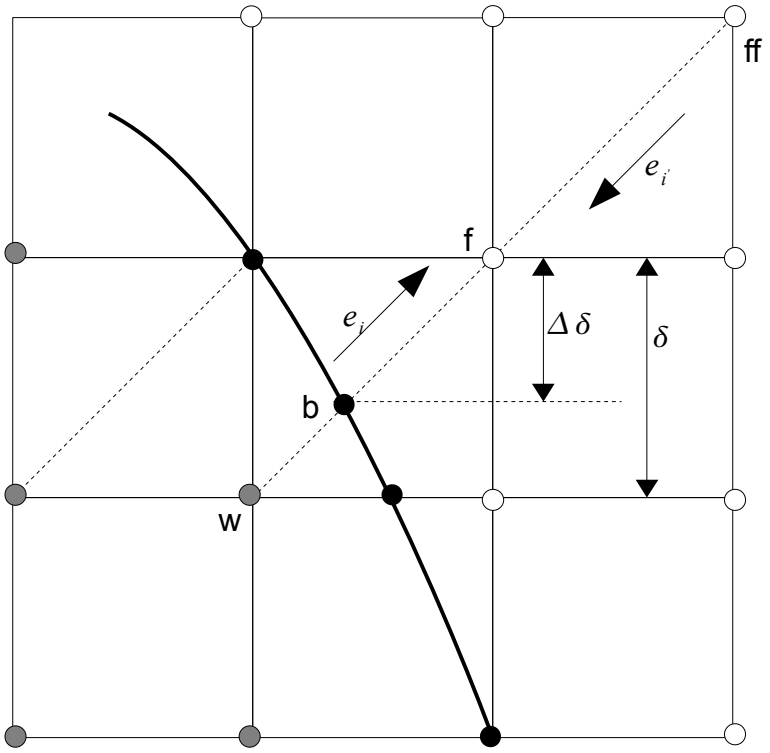

Figure 5.3: Curved boundary

$$f^{\bar{(eq)}}(X_w) = w_i\left[\bar{\rho}_w + \rho_0\left(\frac{e_i\cdot\bar{u}_w}{c_s^2} + \frac{(e_i\cdot\bar{u}_w)^2}{2c_s^4} - \frac{\bar{u}_w^2}{2c_s^2}\right)\right] \tag{5.13}$$

where $\bar{\rho}\equiv\rho(X_w)$ is an approximation of $\rho_w\equiv\rho(X_w)$, and $\bar{u_w}$ is an approximation of $u_w = u(X_w)$ to be chosen. Note that the LBM can be viewed as a special finite difference scheme for the discrete Boltzmann equation [15]. With this knowledge it is reasonable to a determine $\bar{u_w}$ by a linear extrapolation using,

$$\begin{aligned}&\bar{u_w} = u_{w1} \equiv (u_b + (\Delta - 1)u_f)/\Delta \\ &\qquad\qquad \text{or} \\ &\bar{u_w} = u_{w2} \equiv (u_b + (\Delta - 1)u_{ff})/(1+\Delta)\end{aligned} \tag{5.14}$$

It is usually more accurate using u_{w1} than using u_{w2} to approximate $\bar{u_{wf}}$ since X_f is closer to X_w than X_{ff}. It should be noted here that if Δ is small, the denominator in

the expression of u_{wl} will be too large, and it will lead to numerical instability in the computation. Therefore, it is better to use $\bar{u_w} = u_{wl}$ for $\Delta \geq 0.75$. Based on the expression of shear viscosity ν, we can obtain that $M = u_0 / C_s = C_s Re(\tau - 0.5)\delta / L$, where u_0 and *L* are characteristic velocity and length, respectively, and Re is the Reynold's number. Now if we choose τ such that $C_s Re(\tau - 0.5)/L = \Theta(1)$, then Mach number *M* is of the same order of the lattice spacing δ . It is well under stood that in the incompressible limit, the density fluctuation is of order $\Theta(M^2)$, therefore $\bar{\rho_w} = \rho_w + \Theta(\delta M^2)$. From this we can conclude the following :

$$\bar{f_i^{(eq)}}(x_w) - f_i^{(eq)}(x_w) = \Theta(\delta^2) \tag{5.15}$$

To determine the non-equilibrium part, same procedure can be followed. Finally post collision distribution function at the boundary can be prescribed as

$$f_i(X_w + e_i \delta t, t + \delta t) = \bar{f_i^{(eq)}}(X_w, t) + (1 - \tau^{-1}) f_i^{(ne)}(X_w, t). \tag{5.16}$$

This boundary condition scheme is second order in space and time. There are other schemes as well based on idea of finite difference type extrapolation [12].

5.2.3 Neumann boundary condition

Neumann type boundary conditions specifies the value of derivative of the solution takes at the boundaries. Since lattice Boltzmann equation is evolution of distribution function in discrete space and time, It becomes difficult to apply Neumann boundary in terms distribution function. As explained in last two sections Dirichlet boundary conditions for lattice Boltzmann equations are very well studies. If we can represent derivatives in terms of Taylor's expansion and then we can find the solution on the boundary using extrapolation, we should be able to use Neumann condition. To explain this scheme we use an example. Suppose we want to impose following boundary condition.

$$\vec{n} \cdot (\nabla u)|_{\partial\Omega} = 0, \tag{5.17}$$

Where *u* is velocity, $\vec{n}$ is normal vector pointing outwards the computational domain Ω and $\partial\Omega$ is the boundary. For simplicity let us assume $\vec{n}=(-1,0)$. So equation (5.17) reads,

$$\partial_x u|_{\partial\Omega}=0. \tag{5.18}$$

Consider the following figure. Lets say wall is situated at x0 for all y. The value of *u* for x>x0 is known.

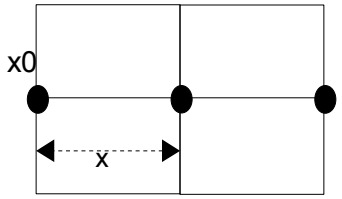

Figure 5.4: Neumann boundary

To use the Dirichlet boundary condition to apply Neumann boundary let us do a Taylor expansion of u up to second order.

$$u(x_0+x)=u(x_0)+x\,u'(x_0)+\frac{x^2}{2}u''(x_0)+\Theta(x^3) \tag{5.19}$$

$$u(x_0+2\mathrm{x})=u(x_0)+2\mathrm{x}\,u'(x_0)+2\mathrm{x}^2u''(x_0)+\Theta(x^3) \tag{5.20}$$

We want to impose $\partial_x u|_{\partial\Omega}=0$. We have two equations and two unknowns. We can solve for,

$$u(x_0)=\frac{4u(x_0+1)-u(x_0+2)}{3}. \tag{5.21}$$

Now imposing this velocity as a Dirichlet boundary condition we can ensure Neumann boundary.

CHAPTER 6

NUMERICAL RESULTS

6.1 Introduction

In this chapter numerical results of lattice Boltzmann method are presented. They are compared with analytical solutions or numerical solutions using another methods such as finite difference or finite element. Comparison is made and, it is shown that lattice Boltzmann method gives satisfactory results within given range of Reynold's number for incompressible flows. Mainly two numerical studies are conducted 1. Poiseuille flow and 2. lid driven cavity flow. For both the problems considered lattice Boltzmann yields satisfactory results. In the end small description of error in lattice Boltzmann is given.

6.2 Poiseuille Flow

Poiseuille flow is an incompressible flow between two parallel stationary plate. In an incompressible fluid Poiseuille flow is created between two stationary walls when pressure gradient or body force is aligned with the walls. Numerical simulation for plain Poiseuille flow driven by either a pressure gradient or a fixed velocity profile at the entrance of channel were carried out to test the validity of the lattice Boltzmann method. Since the analytical solution to Poiseuille flow is available it is an ideal case to check the validity of the lattice Boltzmann equations.

6.2.1 Problem setup

At the entrance of channel, two types of boundary conditions are implemented. One is constant pressure boundary condition, other is a fixed velocity profile. At the outlet , a constant pressure boundary condition is applied.

The velocity solution of the incompressible Navier-Stokes equations corresponding to Poiseuille flow is,

$$u(y)=u_0\left[1-\left(\frac{y}{l}\right)^2\right] \tag{6.1}$$

Where u_0 is the maximum velocity. l Is width of the channel. u_0 can be determined by, $u_0=2FL^2/\rho\nu$, where ν is kinematic viscosity. The channel width is *2L*. The distribution function f_i is evolved by the standard procedure of streaming and collision. In addition, the momentum in channel direction is incremented by adding the amount *F* to f_1 , and subtracting it from f_3 . On the top and bottom No-slip condition was implemented. The density on this boundaries was set to 2.7.

For pressure inlet condition inlet pressure (density) was set to 2.0 and outlet 1.0. Relaxation parameter τ was another variable. For the range of $0.7<\tau<10.0$ results were found matching well with the analytical solution. All the results presented here were taken once the steady state is achieved. Criteria for steady is

$$\sum\frac{|u(x_i,t+1)-u(x_i,t)|}{|u(x_i,t)|}\ll 10^{-6} \tag{6.2}$$

where summation is over entire system. Figure 6.1 shows the velocity profile of u_x . Here u_x is normalized with the maximum velocity at the inlet. Grid size used here was 30X6 for different values of τ . Reynold's number is kept constant to 20.0. Pressure contours along the channel are shown in figure 6.2

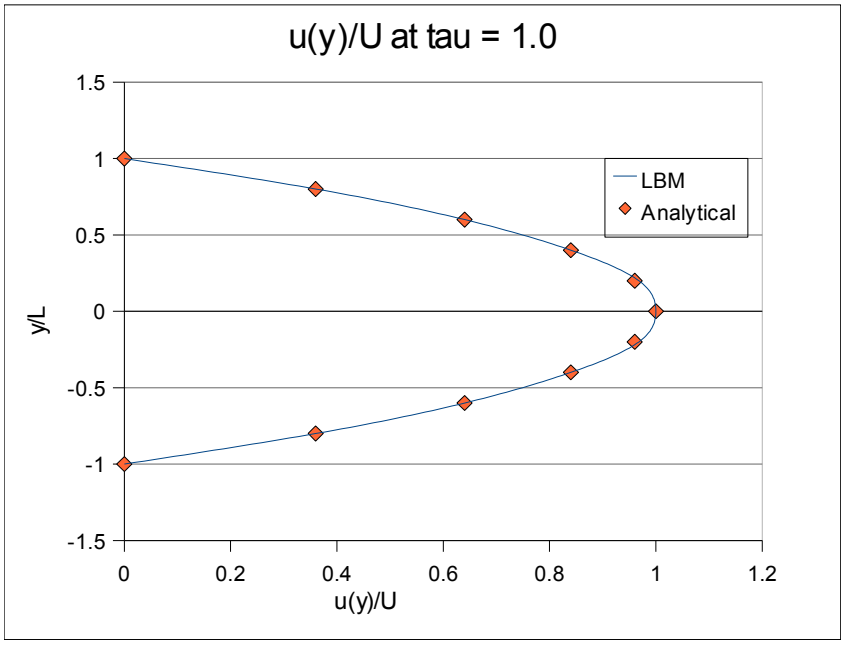

Figure 6.1: u(y)/U Analytical and lattice Boltzmann solution

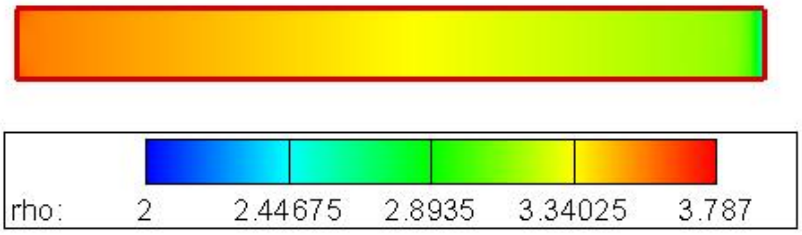

Figure 6.2: Pressure contours

It should be mentioned here the contours presented in the figure 6.2 are density. Pressure is related to density with the relation $p=\rho c_s^2=\rho(1/3)$ for d2q9 lattice. The vertical component of the velocity, u_y is also observed in this numerical experiments. In all cases it was found to be less than 10^{-6} .

All the numerical results agree with the analytical results of the Navier-Stokes equation within the machine accuracy. This is not a surprise since lattice Boltzmann model should work well under the condition that Mach number M<<1.

6.3 Cavity Flow

The problem considered is two-dimensional viscous flow in a cavity. An incompressible fluid is bounded by the a square enclosure and the flow driven by a uniform translation of the top wall. Schematic example is given in figure 6.3.

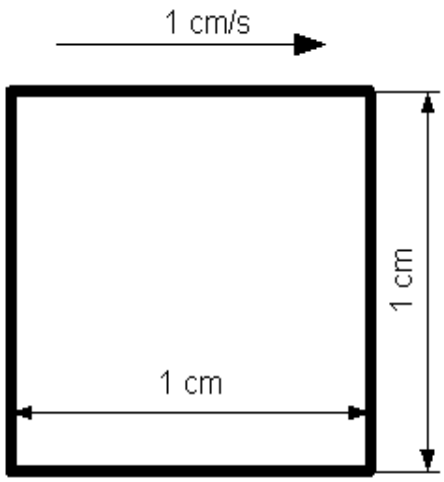

Figure 6.3: Cavity flow

The fluid motion generated in this cavity is an example of closed streamline problems that are of theoretical importance because they are part of the broader field of steady, separated flows. Cavity flow is a very standard benchmark problem for CFD. Most of the conventional numerical solution to this problem use vorticity -stream function formulation and discretize the incompressible Navier-Stokes equation using finite difference[16], multi grid[17] and finite element[3] method and there variations. Most of the validation done here is based on work presented in reference [4]. Results are validated against the results of Ghia et al. [17].

Present numerical study has been carried out for a wide range of Reynold's number. Reynold's number used are from 100, 400, 1000,2000, 5000, and 7500. Various properties such as stream function, pressure contours, velocity profiles etc. against the reference [17].

6.3.1. Boundary Conditions and configuration

Boundary condition for lid driven cavity flow are simple. On the top wall sliding wall or Dirichlet velocity boundary conditions were applied. On other walls Dirichlet velocity boundary

condition with zero normal velocity component were applied. Configuration for the boundary conditions are shown in figure 6.4

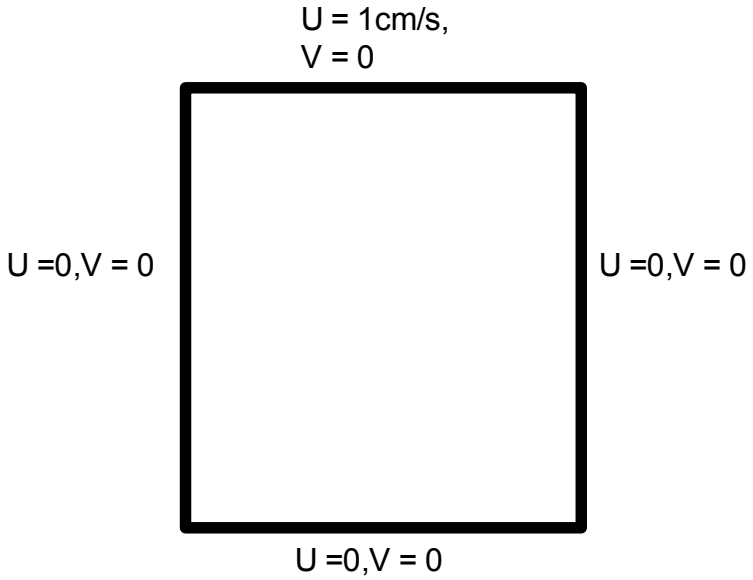

Figure 6.4 : Boundary condition for cavity flow

For the Re = 100, mesh size used was 60x60. For other Reynold's number different configuration were used. For Re = 400,1000 128x128 mesh size was used. For all other Reynold's number 257x257 mesh size was used. Initially the velocities at all nodes except the top nodes, are set to zero. The x-velocity of the top is U and y velocity is zero. Uniform fluid density $\rho = 2.7$ was imposed initially.

The Reynold's number used in the cavity simulation is $Re = U L_N / \nu$. Where *U* is the uniform velocity at the top wall. L_N Is the number of lattice units along any one side of the square cavity. In next section results are discussed.

6.3.2 Stream Function

Figure 6.5 a – f (on next page) show plot of the stream function for the Reynold's number considered. It is apparent that flow structure is in good agreement with the previous

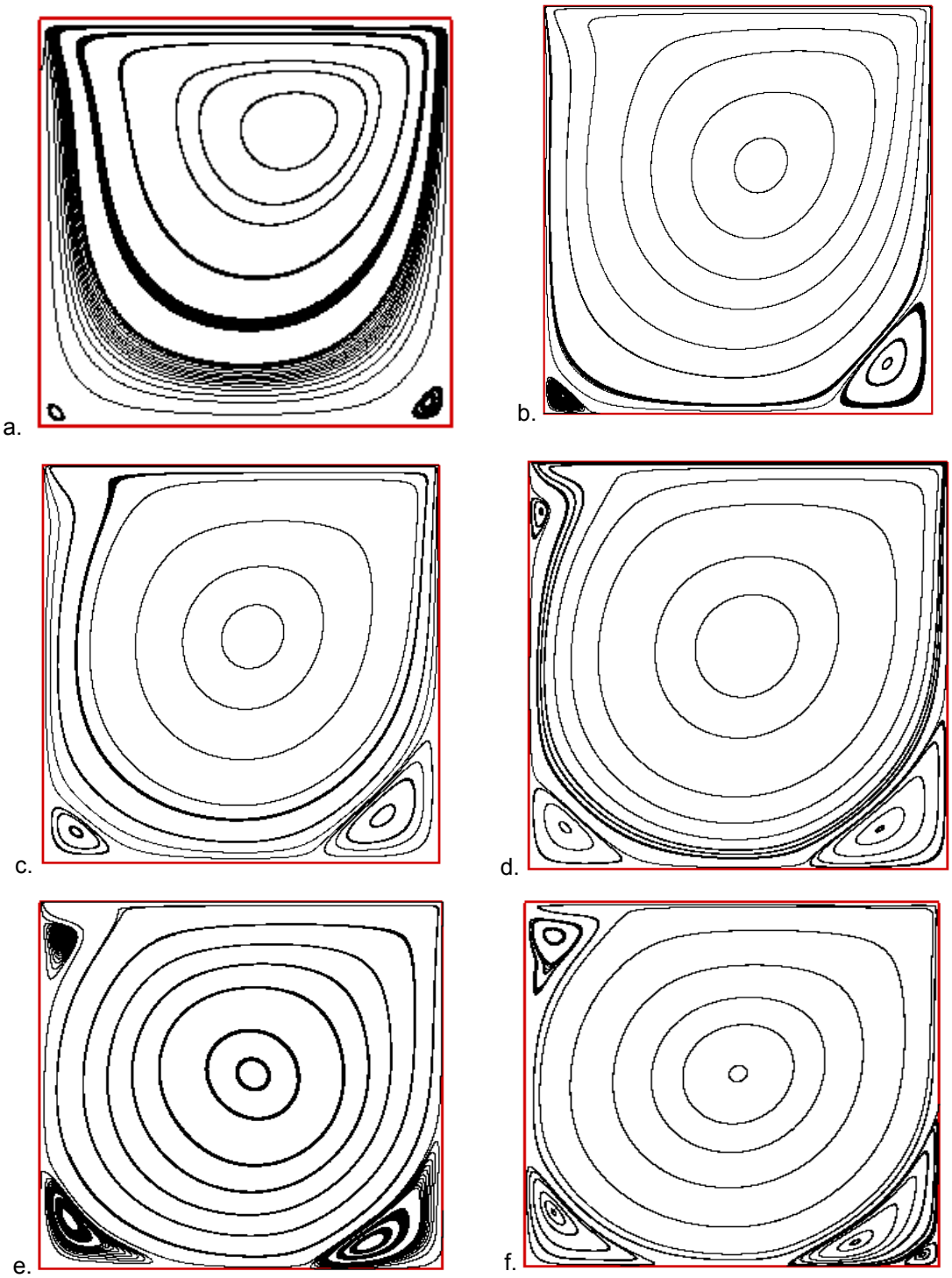

Figure 6.5:Streamlines for a). Re=100 b). Re=400 c). Re=1000 d). 2000 e).5000 f). Re =7500

work of Ghia et al. [17]. These plots give a clear picture of the over all flow pattern and the effect of the Reynold's number on the structure of the steady recirculating eddies in the cavity. In addition to primary center vortex, a pair of counter rotating eddies of much smaller strength develop in the lower corners of the cavity. At Re = 2000, a third secondary vortex is seen in the upper left corner. For Re>5000, a tertiary vortex in the lower right hand corner appears.

As Reynold's number increases, the primary vortex center moves towards the right and becomes increasing circular. Finally, this center moves down towards the geometric center of the cavity as the Re increases and becomes fixed in its x location. The movement of the vortex center location versus Re is shown in figure 6.6.

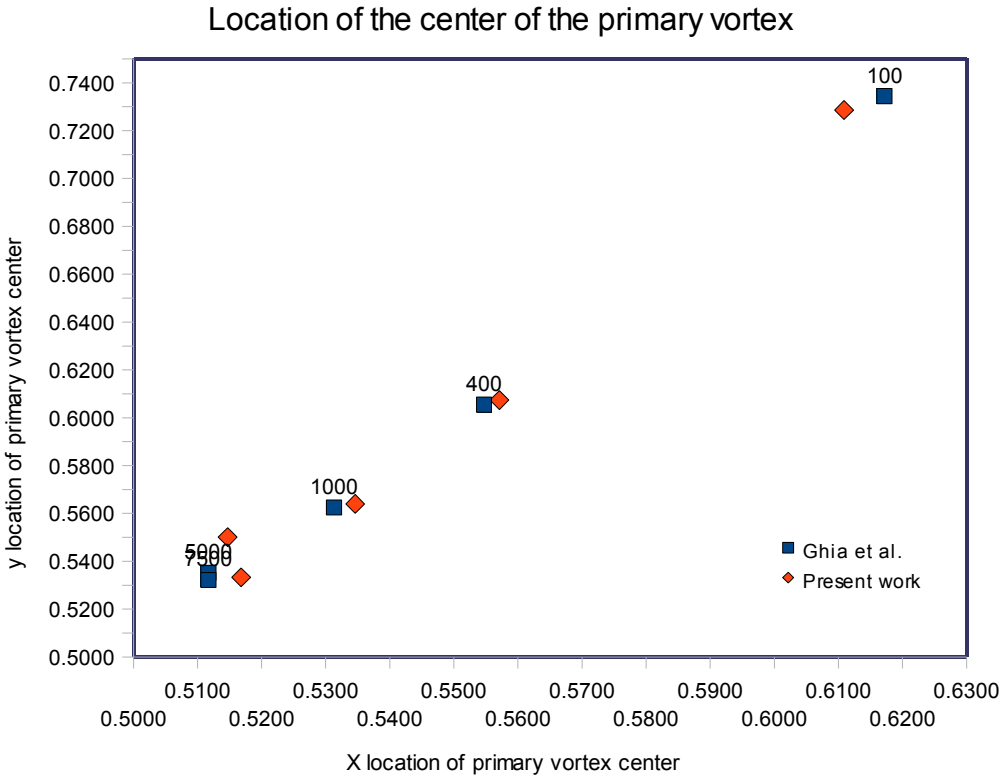

Figure 6.6: Location of the center of the primary vortex

For all the results presented here following steady state criteria was used. $\sum \frac{|u(x_i,t+1)-u(x_i,t)|}{|u(x_i,t)|} \ll 10^{-6}$. Considering the kinetic, unsteady and compressible nature of lattice Boltzmann method, the excellent agreement with other established methods [17], is quite encouraging.

6.3.3 Velocity Profiles

Velocity components along a vertical and horizontal center lines for different values of Reynold's number is given in figure 6.7-6.8. These results are compared against the results presented by Ghia et al [17].

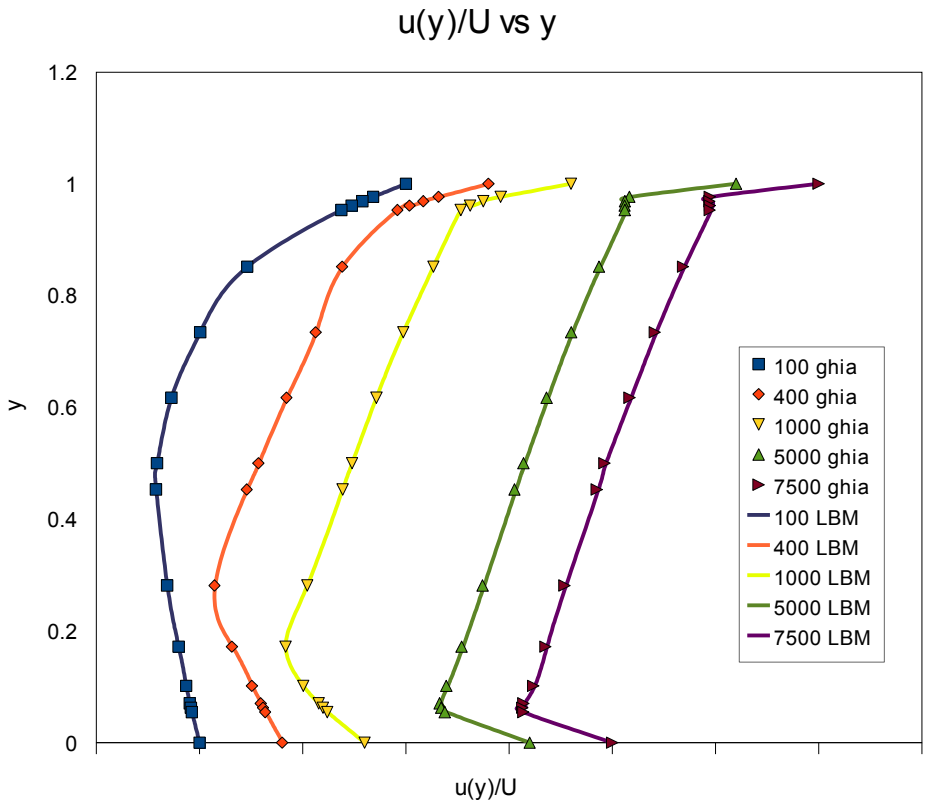

Figure 6.7: Velocity profile for x velocity through geometric center

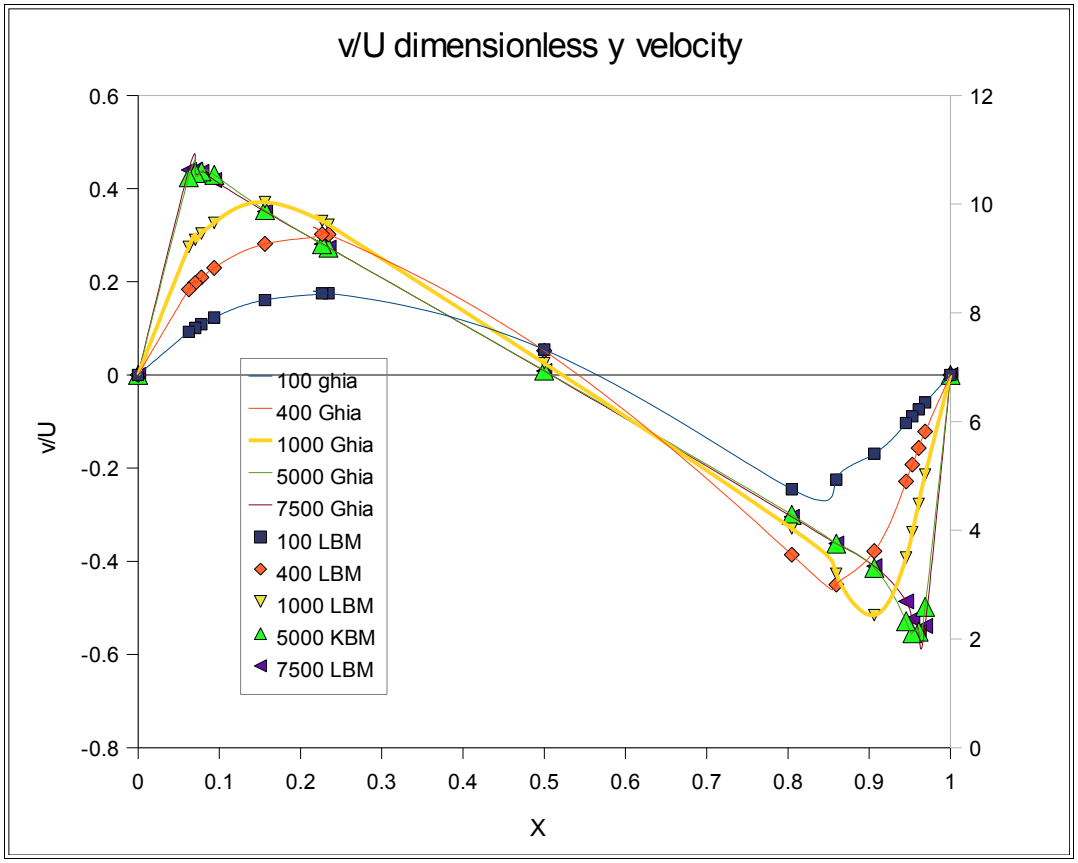

Figure 6.8: Profile of y velocity at the geometric center of cavity

Velocity profiles presented in figure 6.7-6.8 are normalized against top wall velocity U. Profiles matches really well with the presented in reference [17] for all given Reynold's numbers.

6.3.4 Vorticity

The plots are vorticity are given in figure 6.9(a-f). These plots confirms the fact that as Reynold's number increases viscous effects are decreasing. It is reflected by the decrease of shear layer. For 7500 Reynold's number, vorticity contours in center of cavity are nearly constant, while viscous effects are confined to thin shear layers near the wall. The thinning of the wall boundary layers with increasing Reynold's number is evident from these plots.

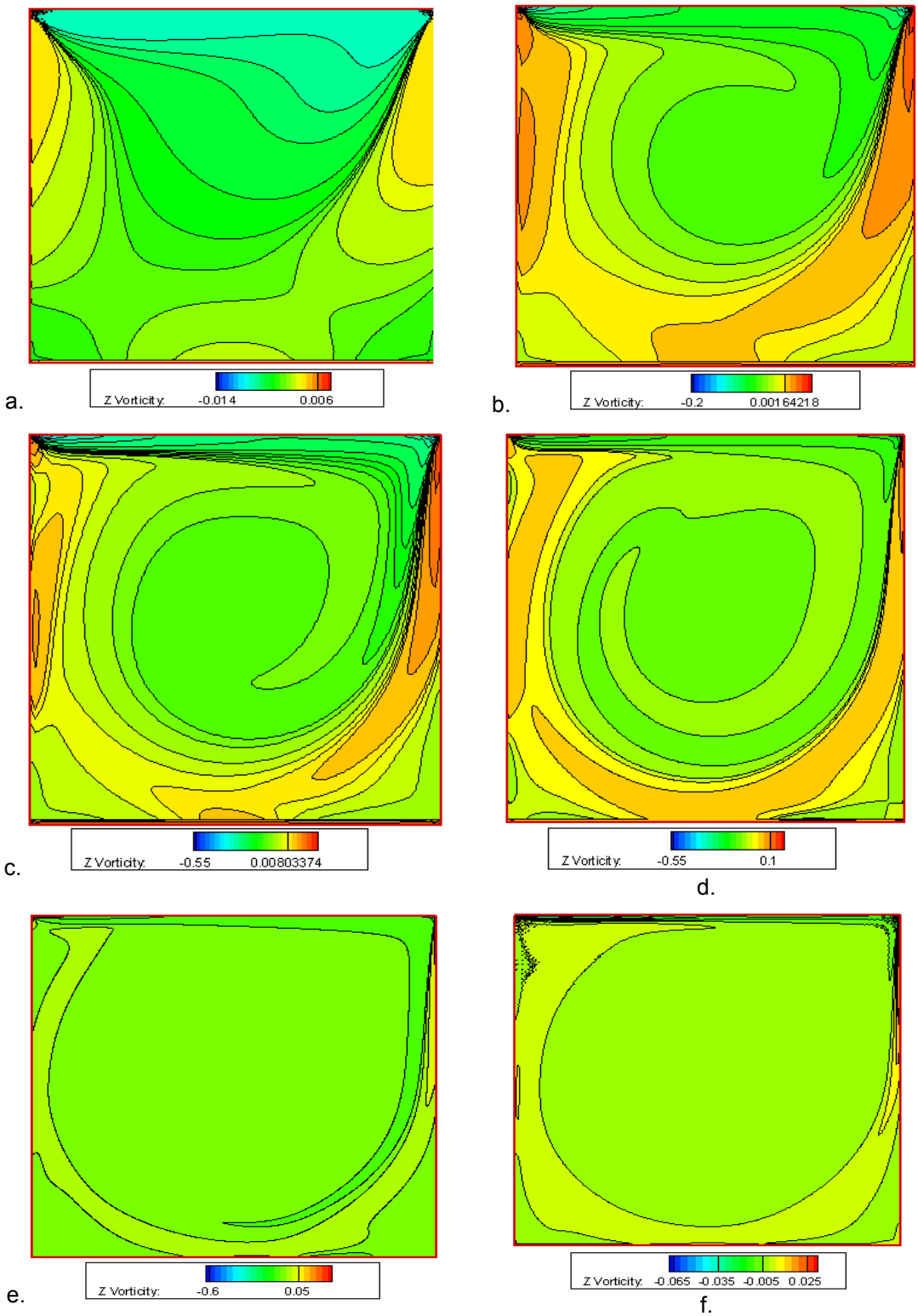

Figure 6.9 : Vorticity contours for Re a. 100,b. 400, c. 1000, d. 2000, e. 5000, f. 7500

6.3.5 Pressure

Pressure contours for different Reynold's numbers are given in the figure 6.10. It is worth noting here that pressure in the lattice Boltzmann method satisfies the equation of state of an isothermal gas, given by $p = c_s^2 \rho$ The observed agreement between different approaches demonstrate that the LBE is valid for simulating incompressible flows.

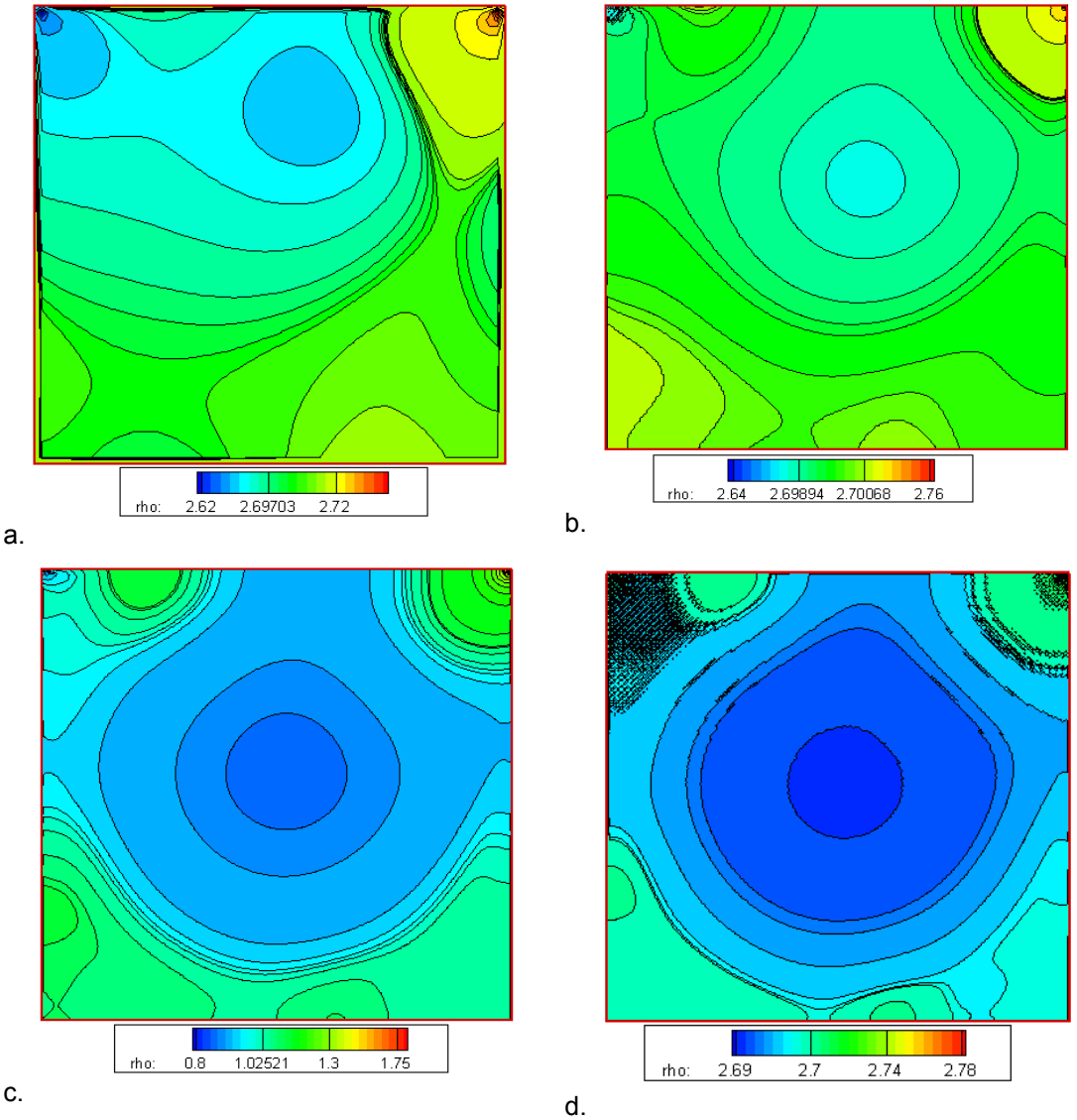

Figure 6.10: Pressure contours a. Re = 100, b. Re = 400, c. Re = 5000, d. Re = 7500

6.4 Sources of Errors

Results presented here matches well with the results given by Ghia et al [17]. At high Reynold's number there are some variation in the location of center of primary vortex and secondary vortex. For the secondary vortex one reason could be the singularities on the corners. In pressure contours results on the corners are fluctuating. Possible source of errors are listed below.

There is a small compressibility effect present in lattice Boltzmann method. It has been shown that lattice Boltzmann equation represents the Navier-Stokes equation in nearly incompressible limit for the small Mach number limit (M=u/Cs<<1). For incompressible fluid density is constant. But in lattice Boltzmann simulations density can not be constant, otherwise pressure changes can not be described. So in the steady case, the continuity equation lattice Boltzmann equation represent is $\nabla\cdot(\rho u)=0.$ Due to non constant density the velocity *u* does not satisfy the incompressible continuity equation given by $\nabla\cdot u=0.$ This could be the source of compressibility error in the method.

6.5 Summary

In this chapter numerical results for lattice Boltzmann method were presented. Two problems were considered. A two-dimensional flow in channel or Poiseuille flow and two-dimensional lid driven cavity flow. Results of lattice Boltzmann method closely matches with the results from traditional CFD methods. Despite of the lattice Boltzmann method being kinetic in nature this validation is quite encouraging and leads to a conclusion that lattice Boltzmann method can be used for sub sonic, incompressible CFD applications.

REFERENCES

[1]. McNamara G, Zanetti G. Use of the Boltzmann Equation to simulate lattice gas automata. Phys rev Lett 1988;61:2332-5

[2]. U. Frisch, B. Hasslacher and Y. Pomeau, “Lattice gas automata for the Navier Stokes Equations”, *Phys. Rev. Lett.* **56**, 1505, 1986.

[3]. G. McNamara and G. Zanetti, “Use of the Boltzmann equation to simulate lattice gas automata”, *Phys. Rev. Lett.* **61**, 2332, 1998.

[4]. Succi S., “The lattice boltzmann equation for fluid dynamics and beyond”, p. **34**, New York: *Oxford University press*, 2001.

[5]. X. He and L.-S. Luo, A priory derivation of the lattice Boltzmann equation, *Phys. Rev.* E **55** R6333, 1997.

[6]. X. He and L.-S. Luo, “Theory of the lattice Boltzmann method: From the Boltzmann equation to the lattice Boltzmann equation”, *Phys. Rev.* E **56**, 6811, 1977.

[7]. P. L. Bhatnagar, E. P. Gross, and M. Krook, “A Model for Collision Processes in Gases. I. Small Amplitude Processes in Charged and Neutral One Component Systems”, *Phys. Rev.* **94**, 511, 1954.

[8]. S. Harris, “An Introduction to the Theory of the Boltzmann Equation”, *Holt, Rinehart and Winston, New York*, 1971.

[9]. R. L. Liboff, “Kinetic Theory”, *Prentice Hall, Englewood Cliffs, N. J.*, 1990.

[10]. U. Frisch, D. d'Humières, B. Hasslacher, P. Lallemand, Y. Pomeau, and J.P. Rivet. “Lattice gas hydrodynamics in two and three dimensions”, *Complex Systems*, 1:649-707, 1987.

[11]. U. Frisch, D. d'Humieres, B. Hasslacher, P. Lallemand, Y. Pomeau, and J. -P. Rivet, “Lattice gas hydrodynamics in two and three dimensions,” Complex Syst, **1**, 649 (1987).

[12]. S. Chen, D. Martinez, and R. Mei, “On boundary conditions in lattice Boltzmann methods,” Phys. Fluids **8** (9), September 1996

[13]. Z. Guo, C. Zheng and B. Shi, “An extrapolation method for boundary conditions in lattice Boltzmann method,” Phys. Of Fluids, Brief Communications, Volume 14 ,**6**, June 2002

[14]. Q. Zou, and X. He, “On pressure and velocity boundary conditions for the lattice Boltzmann BGK model,” Phys. Fluids **9**, 1591 (1997)

[15]. J. D. Sterling and S. Chen, “Stability analysis of the lattice-Boltzmann methods,” J. Comput. Phys. **123**, 196 (1996).

[16] F. Pan and A. Acrivos, J. Fluid Mechanics. **28**, 643 (1967)

[17]. U. Ghia, K. N. Ghia, and C. Y. Shin, J. Comput. Phys. **48**, 387 (1982)

[18]. P. Demaret and M. O. Deville, J. Comput. Phys. **95**, 359 (1991)

[19]. S. Hou, Q. Zou, S. Chen, G. Doolen, “Simulation of Cavity Flow by the Lattice Boltzmann Method”, J. Comput. Phys. **118**, 329 (1995)

BIOGRAPHICAL INFORMATION

Chinmay H. Adhvaryu was born in Valsad, India in 1984. He took his elementary education in Seth R. J. J. Primary School, Valsad and High school education from Experimental High School, Surat, India. He received his Bachelor of Engineering degree in Mechanical engineering from South Gujarat University, Surat, India in 2001. After receiving his Bachelor of engineering he work as a Quality Inspection Engineer at Larsen and Toubro Ltd, Hazira, India. In 2006, he joined University of Texas at Arlington to pursue his graduate studies in Aerospace engineering. He has worked as an intern in Stillwater Supercomputing Inc., in the summer of 2007. His areas of interests include Computational fluid dynamics and scientific computing.