Lattice Boltzmann Flux Solver: An Efficient Approach for Numerical Simulation of Fluid Flows

Shu Chang[1*], *Wang Y*[1], *Yang L M*[2], *Wu J*[2]

1. Department of Mechanical Engineering, National University of Singapore, 10 Kent Ridge Crescent, Singapore, 119260;
2. College of Aerospace Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, 210016, P. R. China

(Received 20 January 2014; revisied 18 February 2014; accepted 22 February 2014)

Abstract: A lattice Boltzmann flux solver (LBFS) is presented for simulation of fluid flows. Like the conventional computational fluid dynamics (CFD) solvers, the new solver also applies the finite volume method to discretize the governing differential equations, but the numerical flux at the cell interface is not evaluated by the smooth function approximation or Riemann solvers. Instead, it is evaluated from local solution of lattice Boltzmann equation (LBE) at cell interface. Two versions of LBFS are presented in this paper. One is to locally apply one-dimensional compressible lattice Boltzmann (LB) model along the normal direction to the cell interface for simulation of compressible inviscid flows with shock waves. The other is to locally apply multi-dimensional LB model at cell interface for simulation of incompressible viscous and inviscid flows. The present solver removes the drawbacks of conventional lattice Boltzmann method (LBM) such as limitation to uniform mesh, tie-up of mesh spacing and time interval, limitation to viscous flows. Numerical examples show that the present solver can be well applied to simulate fluid flows with non-uniform mesh and curved boundary.

Key words: finite volume method; flux solver; compressible flow; incompressible flow; Navier-Stokes equation; lattice Boltzmann equation

CLC number: O35 **Document code**: A **Article ID**: 1005-1120(2014)01-0001-15

1 Introduction

Computational fluid dynamics (CFD) is to apply a numerical method to solve governing equations of fluid flows on the computer. Among various numerical methods available[1-8], the finite volume method (FVM) is the most popular approach in CFD. This is because numerical discretization by FVM is in line with application of physical conservation laws to a control cell. The discrete forms of governing equations by FVM usually involve the conservative variables at cell centers and numerical fluxes at cell interfaces. From numerical point of view, only the conservative variables at cell centers are defined as unknowns, which can be given from the solution of discrete governing equations. In the solution process, we need to use conservative variables at cell centers to evaluate numerical fluxes at cell interfaces. This process is often termed flux solver. Currently, there are three major flux solvers in CFD. One is based on the smooth function approximation. In this solver, a smooth function, which could be a polynomial[9] or a radial basis function[10], is applied to approximate the solution in the local region. The coefficients in the smooth function can be determined by collocation method. Once the smooth function is decided, its integral or derivative can be given in a straightforward way. It should be noted that this solver is a mathematical approach, which can be applied to general engineering problems. However, this solver cannot resolve discontinuity problems such as compressible flows with shock wave. To re-

Foundation item: Supported by the National Natural Science Foundation of China (11272153).

* **Corresponding author**: Shu Chang, Professor, E-mail: mpeshuc@nus.edu.sg.

solve shock wave problems in CFD, the Riemann solver or approximate Riemann solver is often used. The pioneer work in this category was made by Godunov[11], who simplified the compressible flow into a series of Riemann problems and then solved one-dimensional (1D) Euler equations to get local solution. After the work of Godunov[11], various approximate Riemann solvers were presented[12-16]. These solvers usually pursue approximate solution of 1D Euler equations along the normal direction to the cell interface. Thus, they can only be used to evaluate inviscid flux. For compressible viscous flows, the viscous flux is still evaluated by the smooth function approximation. In the literature, there is another type of flux solver called gas kinetic flux solver[17-19], which evaluates inviscid and viscous fluxes simultaneously from local solution of multi-dimensional Boltzmann equation. The solvers in this category can be well applied to simulate both incompressible and compressible flows. But they are usually more complicated and less efficient than the smooth function-based solvers and Riemann solvers. In this paper, we will present a new flux solver, which is based on local solution of lattice Boltzmann equation (LBE).

In recent years, lattice Boltzmann method (LBM)[20-32] has received more and more attention due to its simplicity, easy implementation and parallel nature. In LBM, the density distribution functions are taken as unknowns and LBE is an algebraic formulation. Once the density distribution functions are known at a physical location, the macroscopic flow variables such as density and velocity can be easily computed from local conservation laws of mass and momentum. No differential equation and solution of algebraic equations are involved in the LBE solver. On the other hand, it is indicated that LBE solvers also suffer from some drawbacks. Due to uniformity of the lattice, the standard LBE solver is limited to the simple geometry and uniform mesh. For complex geometry and application on the non-uniform mesh, additional efforts such as interpolation have to be incorporated. The process may increase the complexity of the solver, and requires additional computational effort and virtual storage. The second drawback is the tie-up of time interval with mesh spacing. This drawback makes the adaptive and multi-block computation of LBE solvers extremely complicated. In addition, LBE solvers need more memory to store density distribution functions than the Navier-Stokes (N-S) solvers. Another drawback is that LBE solvers can only be applied to simulate viscous flows. Furthermore, the physical boundary conditions such as given pressure cannot be implemented directly in the LBE solver. As will be shown in this paper, all the above drawbacks of LBE solvers are completely removed by the lattice Boltzmann flux solver (LBFS).

LBFS is based on Chapman-Enskog (C-E) expansion analysis, which is a bridge to link N-S equations and LBE. Usually, the C-E analysis is applied in the whole flow domain to verify that the macroscopic flow variables obtained by LBE solvers at any physical location and any time level can satisfy N-S equations. On the other hand, it was found that the C-E analysis can be applied at any location within a small streaming step. This idea has been well applied by Xu[17] in the development of gas kinetic scheme, where the flux at the cell interface is computed by local solution of BGK equation. In this work, the numerical fluxes at the cell interface are evaluated by local reconstruction of LBE solution. Two versions of LBFS are presented in this work. One is to locally apply 1D compressible LB model along the normal direction of cell interface for evaluation of inviscid flux. This version is only applicable for simulation of compressible inviscid flows. The other is to locally apply multi-dimensional LB model at the cell interface for evaluation of viscous and inviscid fluxes simultaneously. The performance of present LBFS will be investigated through some test examples. Numerical results demonstrate that LBFS can accurately and effectively simulate fluid flows with curved boundary and non-uniform mesh. It also removes the drawbacks of conventional LBM.

2 Lattice Boltzmann Flux Solver (LBFS) for Compressible Inviscid Flows

The integral form of Euler equations without source term can be written as

$$\frac{\partial}{\partial t}\int_{\Omega}\boldsymbol{W}_n\,\mathrm{d}\Omega+\oint_{\Gamma}\boldsymbol{F}_n\,\mathrm{d}S=0 \tag{1}$$

where the conservative flow variables $\boldsymbol{W}$ and inviscid flux $\boldsymbol{F}_n$ are given by

$$\boldsymbol{W}=\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ \rho w\\ \rho E\end{bmatrix},\quad \boldsymbol{F}_n=\begin{bmatrix}\rho U_n\\ \rho u U_n+n_x p\\ \rho v U_n+n_y p\\ \rho w U_n+n_z p\\ (\rho E+p)U_n\end{bmatrix} \tag{2}$$

where ρ and p are the density and pressure of the mean flow, respectively. $\boldsymbol{U}=(u,v,w)$ is the velocity vector in the Cartesian coordinate system and $\boldsymbol{n}=(n_x,n_y,n_z)$ denotes the unit normal vector on the control surface. U_n represents the normal velocity, which is defined as the scalar product of the velocity vector and the unit normal vector, i. e.

$$U_n=n_x u+n_y v+n_z w \tag{3}$$

E is the total energy of the mean flow, which is defined as

$$E=e+\frac{1}{2}(u^2+v^2+w^2) \tag{4}$$

Here $e=p/[(\gamma-1)\rho]$ is the potential energy of the mean flow, and γ is the specific heat ratio. On the control surface, the tangential velocity $\boldsymbol{U}_\tau=(U_{\tau x},U_{\tau y},U_{\tau z})$ can be computed by

$$\boldsymbol{U}_\tau=(U_{\tau x},U_{\tau y},U_{\tau z})=\boldsymbol{U}-U_n\boldsymbol{n} \tag{5}$$

Applying Eq. (1) to a control volume gives

$$\frac{\mathrm{d}\boldsymbol{W}_I}{\mathrm{d}t}=-\frac{1}{\Omega_I}\sum_{i=1}^{N_f}\boldsymbol{F}_{ni}\,\mathrm{d}S_i \tag{6}$$

where I is the index of a control volume, Ω_I and N_f represent the volume and the number of the faces of the control volume I. $\mathrm{d}S_i$ denotes the area of the ith face of the control volume. As indicated in the introduction, the flux solver needs to reconstruct numerical flux $\boldsymbol{F}_n$ at each cell interface from the conservative variables $\boldsymbol{W}_I$ at cell centers. In this section, $\boldsymbol{F}_n$ will be computed from the solution of 1D compressible LB model to a local Riemann problem. When 1D LB model is applied along the normal direction to the cell interface, only density, pressure and normal velocity are involved. Thus, before we address how to apply the 1D compressible LB model to reconstruct $\boldsymbol{F}_n$, it is better to rewrite expression of $\boldsymbol{F}_n$ in terms of density, pressure, normal velocity and tangential velocity. From Eq. (5), we can express the velocity components in the Cartesian coordinate system in terms of normal velocity and tangential velocity as

$$\begin{aligned}u&=U_n n_x+U_{\tau x}\\ v&=U_n n_y+U_{\tau y}\\ w&=U_n n_z+U_{\tau z}\end{aligned} \tag{7}$$

Using Eq. (7) and the expression of potential energy, $\boldsymbol{F}_n$ can be rewritten as

$$\boldsymbol{F}_n=\begin{bmatrix}\rho U_n\\ (\rho U_n U_n+p)n_x+\rho U_n U_{\tau x}\\ (\rho U_n U_n+p)n_y+\rho U_n U_{\tau y}\\ (\rho U_n U_n+p)n_z+\rho U_n U_{\tau z}\\ \left(\dfrac{\gamma}{\gamma-1}p+\dfrac{1}{2}\rho U_n^2\right)U_n+\dfrac{1}{2}\rho U_n|\boldsymbol{U}_\tau|^2\end{bmatrix} \tag{8}$$

It can be seen clearly from Eq. (8) that, to evaluate numerical flux $\boldsymbol{F}_n$, we need to know the density, pressure, normal velocity and tangential velocity at the cell interface. This task can be fulfilled by local application of 1D compressible LB model to the Riemann problem defined at cell interface. In this work, the non-free parameter D1Q4 LB model presented in Ref. [33-34] is adopted. This model is derived from conservation forms of moments, which can be used to simulate hypersonic flows with strong shock waves. The non-free parameter D1Q4 model is shown in Fig. 1. The equilibrium distribution functions and lattice velocities of this model are given below,

$$g_1=\frac{\rho(-d_1d_2^2-d_2^2u+d_1u^2+d_1c^2+u^3+3uc^2)}{2d_1(d_1^2-d_2^2)}$$

$$g_2=\frac{\rho(-d_1d_2^2+d_2^2u+d_1u^2+d_1c^2-u^3-3uc^2)}{2d_1(d_1^2-d_2^2)}$$

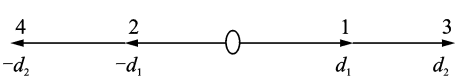

Fig. 1 Configuration of non-free parameter D1Q4 model

$$g_3=\frac{\rho(d_1^2d_2+d_1^2u-d_2u^2-d_2c^2-u^3-3uc^2)}{2d_2(d_1^2-d_2^2)}$$
$$g_4=\frac{\rho(d_1^2d_2-d_1^2u-d_2u^2-d_2c^2+u^3+3uc^2)}{2d_2(d_1^2-d_2^2)} \tag{9}$$

$$d_1=\sqrt{u^2+3c^2-\sqrt{4u^2c^2+6c^4}}$$
$$d_2=\sqrt{u^2+3c^2+\sqrt{4u^2c^2+6c^4}} \tag{10}$$

where g_i is the equilibrium distribution function in the ith direction of phase space, d_i is the lattice velocity in the ith direction, c is the peculiar velocity of particles defined as $c=\sqrt{D(\gamma-1)e}$ (D is the dimension of space). Note that when the above 1D model is applied along the normal direction to the cell interface, u has to be replaced by U_n.

Next, we will show how to apply the non-free parameter D1Q4 model to evaluate $\boldsymbol{F}_n$ at cell interface. As shown in Fig. 1, at any physical location, D1Q4 model has 4 moving particles. Now, we consider a local Riemann problem around a cell interface as shown in Fig. 2. To compute $\boldsymbol{F}_n$, we need to know distribution functions of 4 moving particles at the cell interface. In the framework of LBM, the moving particles are actually streamed from neighbouring points. As illustrated in Fig. 3, by giving a streaming step δ_t, particles 1 and 3 from left side of interface will stream to the cell interface while particles 2 and 4 from right side of interface will also stream to the cell interface. Mathematically, the streaming process provides the distribution functions of four moving particles at cell interface as

$$f_i^{\text{interface}}=\begin{cases}g_i^{\text{L}} & \text{if} \quad i=1,3\\ g_i^{\text{R}} & \text{if} \quad i=2,4\end{cases} \tag{11}$$

where g_i^{L} and g_i^{R} are the equilibrium distribution functions at the left and right sides of cell interface. For the Riemann problem, they are given from information at left and right cell centers.

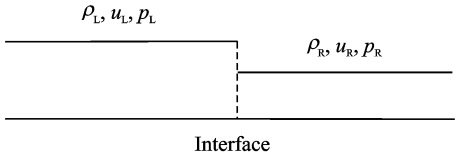

Fig. 2 Configuration of a Riemann problem

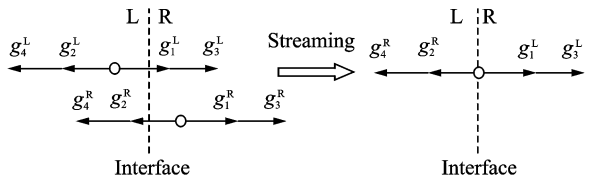

Fig. 3 Streaming process of D1Q4 model at the cell interface

With flow variables, they can be computed by using Eq. (9). With Eq. (11), there are two basic ways to evaluate the numerical flux $\boldsymbol{F}_n$ at the cell interface. The first way is to compute the flow variables (density, pressure and normal velocity) first, and then substitute them into Eq. (8) to compute $\boldsymbol{F}_n$. The density, normal velocity and pressure can be computed by

$$\rho=\sum_{i=1}^{4}f_i^{\text{interface}} \tag{12a}$$
$$\rho U_n=\sum_{i=1}^{4}f_i^{\text{interface}}e_i \tag{12b}$$
$$\left(\frac{1}{\gamma-1}p+\frac{1}{2}\rho U_n^2\right)=\sum_{i=1}^{4}f_i^{\text{interface}}\left(\frac{1}{2}e_ie_i+\lambda\right) \tag{12c}$$

where e_i is the lattice velocity, $e_1=d_1$, $e_2=-d_1$, $e_3=d_2$, $e_4=-d_2$, $\lambda=\left[1-\frac{D}{2}(\gamma-1)\right]e$ is the potential energy of particles (D is the dimension of space and takes 1 for the 1D model). The tangential density $\boldsymbol{U}_\tau$ at the cell interface can be given from mean value of $\boldsymbol{U}_\tau^{\text{L}}$ and $\boldsymbol{U}_\tau^{\text{R}}$, where $\boldsymbol{U}_\tau^{\text{L}}$ and $\boldsymbol{U}_\tau^{\text{R}}$ are the tangential velocity at the left and right side of cell interface, respectively. Alternatively, it can be approximated by

$$\rho\boldsymbol{U}_\tau=\sum_{i=1}^{4}f_i^{\text{interface}}\cdot\boldsymbol{U}_\tau=\sum_{i=1,3}g_i^{\text{L}}\cdot\boldsymbol{U}_\tau^{\text{L}}+\sum_{i=2,4}g_i^{\text{R}}\cdot\boldsymbol{U}_\tau^{\text{R}} \tag{13}$$

Once the density, pressure, normal velocity and tangential velocity at the cell interface are computed by Eqs. (12—13), they can be substituted into Eq. (8) to compute $\boldsymbol{F}_n$. This way is equivalent to use equilibrium distribution functions at the cell interface to compute $\boldsymbol{F}_n$. From CE analysis, this way has very little numerical dissipation, which may not be able to get stable solution for problems with strong shock waves. To compute

$\boldsymbol{F}_n$ with numerical dissipation, we can use distribution function given in Eq. (11) to compute $\boldsymbol{F}_n$ directly. In fact, ρU_n in $\boldsymbol{F}_n$ has been calculated by Eq. (12b). Other terms in $\boldsymbol{F}_n$ can be computed by the following formulations

$$\rho U_n U_n + p = \sum_{i=1}^{4} f_i^{\text{interface}} e_i e_i \tag{14}$$

$$\left(\frac{\gamma}{\gamma-1}p + \frac{1}{2}\rho U_n^2\right) U_n = \sum_{i=1}^{4} f_i^{\text{interface}} e_i \left(\frac{1}{2} e_i e_i + \lambda\right) \tag{15}$$

Similar to Eq. (13), $\rho U_n \boldsymbol{U}_\tau$ and $\rho U_n |\boldsymbol{U}_\tau|^2$ can be approximated by

$$\rho U_n \boldsymbol{U}_\tau = \sum_{i=1}^{4} f_i^{\text{interface}} e_i \cdot \boldsymbol{U}_\tau = \sum_{i=1,3} g_i^{\text{L}} e_i \cdot \boldsymbol{U}_\tau^{\text{L}} + \sum_{i=2,4} g_i^{\text{R}} e_i \cdot \boldsymbol{U}_\tau^{\text{R}} \tag{16}$$

$$\rho U_n |\boldsymbol{U}_\tau|^2 = \sum_{i=1}^{4} f_i^{\text{interface}} e_i \cdot |\boldsymbol{U}_\tau|^2 = \sum_{i=1,3} g_i^{\text{L}} e_i \cdot |\boldsymbol{U}_\tau^{\text{L}}|^2 + \sum_{i=2,4} g_i^{\text{R}} e_i \cdot |\boldsymbol{U}_\tau^{\text{R}}|^2 \tag{17}$$

Overall, the basic solution procedure of this LBFS can be summarized below:

(1) At first, we need to choose a 1D LB model such as non-free parameter D1Q4 model. The LB model provides expressions for equilibrium distribution functions and lattice velocities.

(2) For the considered cell interface with unit normal vector $\boldsymbol{n} = (n_x, n_y, n_z)$, obtain flow variables (density, pressure, velocity components) at the left and right sides of interface from two neighbouring cell centers (MUSCL interpolation with limiter may be used for high-order schemes). Then use Eqs. (3, 5) to calculate the normal and tangential velocities at the left and right sides of interface.

(3) Use Eq. (9) to calculate $g_1^{\text{L}}, g_3^{\text{L}}, g_2^{\text{R}}, g_4^{\text{R}}$ by using density, pressure and normal velocity.

(4) Compute the density, normal velocity, pressure and tangential velocity at the cell interface by using Eqs. (12—13), and then substitute them into Eq. (8) to calculate numerical flux $\boldsymbol{F}_n$. Alternatively, use Eqs. (12b), (14—17) to compute $\boldsymbol{F}_n$ directly (this way is recommended for hypersonic flows with strong shock waves).

(5) Once numerical fluxes at all cell interfaces are obtained, solve ordinary differential equations (6) by using 4-stage Runge-Kutta scheme.

For simulation of viscous flows, one also needs to use a smooth function to approximate the viscous flux.

3 Lattice Boltzmann Flux Solver (LBFS) for Incompressible Flows

From C-E expansion analysis[22, 32], the incompressible Navier-Stokes (N-S) equations

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{u}) = 0 \tag{18}$$

$$\frac{\partial \rho \boldsymbol{u}}{\partial t} = -\nabla \cdot (\rho \boldsymbol{u}\boldsymbol{u}) - \nabla p + \nu \nabla \cdot [\nabla \rho \boldsymbol{u} + (\nabla \rho \boldsymbol{u})^{\text{T}}] = -\nabla \cdot \boldsymbol{\Pi} \tag{19}$$

can be recovered by the following LBE

$$f_\alpha(\boldsymbol{r} + \boldsymbol{e}_\alpha \delta_t, t + \delta_t) = f_\alpha(\boldsymbol{r}, t) + \frac{f_\alpha^{\text{eq}}(\boldsymbol{r}, t) - f_\alpha(\boldsymbol{r}, t)}{\tau}, \alpha = 0, 1, \cdots, N \tag{20}$$

where ρ is the fluid density, $\boldsymbol{u}$ the flow velocity and p the pressure. $\boldsymbol{r}$ represents a physical location, τ is the single relaxation parameter; f_α is the density distribution function along the α direction; f_α^{eq} is its corresponding equilibrium state; δ_t is the streaming time step and $\boldsymbol{e}_\alpha$ is the particle velocity in the α direction; N is the number of discrete particle velocities. The relationships between the density distribution functions and flow variables as well as fluxes in the N-S equations are

$$\rho = \sum_{\alpha=0}^{N} f_\alpha^{\text{eq}} \tag{21}$$

$$\rho u_\gamma = \sum_{\alpha=0}^{N} e_{\alpha\gamma} f_\alpha^{\text{eq}} \tag{22}$$

$$\Pi_{\beta\gamma} = \sum_{\alpha=0}^{N} e_{\alpha\beta} e_{\alpha\gamma} \left[f_\alpha^{\text{eq}} + \left(1 - \frac{1}{2\tau}\right) \varepsilon f_\alpha^{(1)} \right] \tag{23}$$

where β and γ represent the space coordinate directions, and $e_{\alpha\beta}$ is the component of the lattice velocity vector $\boldsymbol{e}_\alpha$ in the β-coordinate direction. As shown in Ref. [22, 32], to recover N-S equations by Eq. (20), $\varepsilon f_\alpha^{(1)}$ can be approximated by

$$f_\alpha^{neq} \approx \varepsilon f_\alpha^{(1)} = -\tau \delta_t \left(\frac{\partial}{\partial t} + \boldsymbol{e}_\alpha \cdot \nabla \right) f_\alpha^{\text{eq}} \tag{24}$$

Substituting Eq. (24) into Eq. (23) gives

$$\Pi_{\beta\gamma}=\sum_{\alpha=0}^{N}e_{\alpha\beta}e_{\alpha\gamma}\left[f_{\alpha}^{\mathrm{eq}}+\left(1-\frac{1}{2\tau}\right)f_{\alpha}^{n\mathrm{eq}}\right] \tag{25}$$

The equilibrium distribution function f_{α}^{eq} depends on the lattice velocity model used. For example, when the following two-dimensional D2Q9 lattice velocity model

$$\boldsymbol{e}_{\alpha}=\begin{cases}0 & \alpha=0\\ \left(\cos\left[(\alpha-1)\dfrac{\pi}{2}\right],\sin\left[(\alpha-1)\dfrac{\pi}{2}\right]\right)c & \alpha=1,2,3,4\\ \sqrt{2}\left(\cos\left[(\alpha-5)\dfrac{\pi}{2}+\dfrac{\pi}{4}\right],\sin\left[(\alpha-5)\dfrac{\pi}{2}+\dfrac{\pi}{4}\right]\right)c & \alpha=5,6,7,8\end{cases} \tag{26}$$

is used, f_{α}^{eq} can be given by

$$f_{\alpha}^{\mathrm{eq}}(\boldsymbol{r},t)=\rho w_{\alpha}\left[1+\frac{\boldsymbol{e}_{\alpha}\cdot\boldsymbol{u}}{c_{\mathrm{s}}^{2}}+\frac{(\boldsymbol{e}_{\alpha}\cdot\boldsymbol{u})^{2}-(c_{\mathrm{s}}|\boldsymbol{u}|)^{2}}{2c_{\mathrm{s}}^{4}}\right] \tag{27}$$

where $c=\delta_x/\delta_t$, δ_x is the lattice spacing. For the case of $\delta_x=\delta_t$, which is often used in the literature and also adopted in this work, c is taken as 1. The coefficients w_{α} and the sound speed c_{s} are given as: $w_0=4/$, $w_1=w_2=w_3=w_4=1/9$ and $w_5=w_6=w_7=w_8=1/36$. $c_{\mathrm{s}}=c/\sqrt{3}$. The relaxation parameter τ is linked to the kinematic viscosity of fluid through C-E expansion analysis by the following relationship

$$\upsilon=\left(\tau-\frac{1}{2}\right)c_{\mathrm{s}}^{2}\delta_t \tag{28}$$

The pressure can be calculated from the equation of state by

$$p=\rho c_{\mathrm{s}}^{2} \tag{29}$$

Using Eqs. (22) and (23), for the two-dimensional case, Eqs. (18) and (19) can be rewritten as

$$\frac{\partial\boldsymbol{W}}{\partial t}+\frac{\partial\boldsymbol{E}}{\partial x}+\frac{\partial\boldsymbol{F}}{\partial y}=0 \tag{30}$$

where

$$\boldsymbol{W}=\begin{Bmatrix}\rho\\ \rho u\\ \rho v\end{Bmatrix},\boldsymbol{E}=\begin{Bmatrix}P_x\\ \Pi_{xx}\\ \Pi_{yx}\end{Bmatrix},\ \boldsymbol{F}=\begin{Bmatrix}P_y\\ \Pi_{xy}\\ \Pi_{yy}\end{Bmatrix} \tag{31}$$

$$P_x=\sum_{\alpha=0}^{N}e_{\alpha x}f_{\alpha}^{\mathrm{eq}} \tag{32a}$$

$$P_y=\sum_{\alpha=0}^{N}e_{\alpha y}f_{\alpha}^{\mathrm{eq}} \tag{32b}$$

$$\Pi_{xx}=\sum_{\alpha=0}^{N}e_{\alpha x}e_{\alpha x}\left[f_{\alpha}^{\mathrm{eq}}+\left(1-\frac{1}{2\tau}\right)f_{\alpha}^{n\mathrm{eq}}\right] \tag{33a}$$

$$\Pi_{xy}=\sum_{\alpha=0}^{N}e_{\alpha x}e_{\alpha y}\left[f_{\alpha}^{\mathrm{eq}}+\left(1-\frac{1}{2\tau}\right)f_{\alpha}^{n\mathrm{eq}}\right] \tag{33b}$$

$$\Pi_{yx}=\sum_{\alpha=0}^{N}e_{\alpha y}e_{\alpha x}\left[f_{\alpha}^{\mathrm{eq}}+\left(1-\frac{1}{2\tau}\right)f_{\alpha}^{n\mathrm{eq}}\right] \tag{33c}$$

$$\Pi_{yy}=\sum_{\alpha=0}^{N}e_{\alpha y}e_{\alpha y}\left[f_{\alpha}^{\mathrm{eq}}+\left(1-\frac{1}{2\tau}\right)f_{\alpha}^{n\mathrm{eq}}\right] \tag{33d}$$

When a cell-centered FVM is applied to solve Eq. (30), the flow properties ρ and $\rho\boldsymbol{u}$ at the cell center can be obtained by marching in time. The fluxes at the cell interface can be evaluated by local reconstruction of LBM solution. By integrating Eq. (30) over a control volume Ω_i, we have

$$\frac{\mathrm{d}\boldsymbol{W}_i}{\mathrm{d}t}=-\frac{1}{\Delta V_i}\sum_{k}\boldsymbol{R}_k\Delta S_k \tag{34}$$

$$\boldsymbol{R}_k=(n_x\boldsymbol{E}+n_y\boldsymbol{F})_k$$

where ΔV_i is the volume of Ω_i, and ΔS_k is the area of the kth control surface enclosing Ω_i. n_x and n_y are the x and y components of the unit outward normal vector on the kth control surface. Obviously, once the fluxes at all cell interfaces are known, Eq. (34) can be solved by well established numerical schemes such as the 4-stage Runge-Kutta method. Thus, the evaluation of flux R_k at the cell interface is the key in the solution process. The detailed expression of R_k depends on the lattice velocity model. By defining f_{α}^{*} as

$$f_{\alpha}^{*}=f_{\alpha}^{\mathrm{eq}}+\left(1-\frac{1}{2\tau}\right)f_{\alpha}^{n\mathrm{eq}} \tag{35}$$

When the D2Q9 lattice velocity model is used, $\boldsymbol{R}_k$ can be written in detail as follows

$$\boldsymbol{R}_k=\begin{pmatrix}n_x(f_1^{\mathrm{eq}}-f_3^{\mathrm{eq}}+f_5^{\mathrm{eq}}-f_6^{\mathrm{eq}}-f_7^{\mathrm{eq}}+f_8^{\mathrm{eq}})+n_y(f_2^{\mathrm{eq}}-f_4^{\mathrm{eq}}+f_5^{\mathrm{eq}}+f_6^{\mathrm{eq}}-f_7^{\mathrm{eq}}-f_8^{\mathrm{eq}})\\ n_x(f_1^{*}+f_3^{*}+f_5^{*}+f_6^{*}+f_7^{*}+f_8^{*})+n_y(f_5^{*}-f_6^{*}+f_7^{*}-f_8^{*})\\ n_x(f_5^{*}-f_6^{*}+f_7^{*}-f_8^{*})+n_y(f_2^{*}+f_4^{*}+f_5^{*}+f_6^{*}+f_7^{*}+f_8^{*})\end{pmatrix}_k \tag{36}$$

Obviously, the key issue in the evaluation of the flux $\boldsymbol{R}_k$ is to perform an accurate evaluation of f_{α}^{eq} and f_{α}^{*} at the cell interface. In the following, we will show the detailed calculation of f_{α}^{eq} and f_{α}^{*} at the cell interface.

Consider a cell interface between two control

cells Ω_i and Ω_{i+1} as shown in Fig. 4. It is assumed that the physical location for the two cell centers and their interface is $\boldsymbol{r}_i$, $\boldsymbol{r}_{i+1}$ and $\boldsymbol{r}$ respectively.

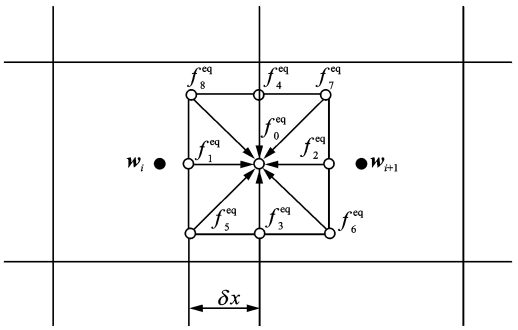

Fig. 4 Local reconstruction of LBM solution at a cell interface

Using Taylor series expansion, we have

$$f_\alpha^{eq}(\boldsymbol{r},t)-f_\alpha^{eq}(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t,t-\delta_t)=\delta_t\left(\frac{\partial}{\partial t}+\boldsymbol{e}_\alpha\cdot\nabla\right)f_\alpha^{eq}+O(\delta_t^2) \tag{37}$$

From Eqs. (37) and (24), we can get the following form

$$f_\alpha^{neq}(\boldsymbol{r},t)=f_\alpha^{neq}(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t,t-\delta_t)=-\tau[f_\alpha^{eq}(\boldsymbol{r},t)-f_\alpha^{eq}(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t,t-\delta_t)]+O(\delta_t^2) \tag{38}$$

Eq. (38) shows that once we have the equilibrium distribution functions $f_\alpha^{eq}(\boldsymbol{r},t)$, $f_\alpha^{eq}(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t,t-\delta_t)$ at the cell interface and its surrounding points, we can have the full information of distribution function at the interface. Note that the approximation for Eq. (38) is the second order of accuracy in δ_t. Using Eq. (27), the equilibrium distribution function f_α^{eq} can be computed from the fluid density ρ and flow velocity $\boldsymbol{u}$. With the given density and velocity at the cell center, the respective density and velocity at location $(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t)$ can be easily obtained by interpolation. One of interpolation forms can be written as

$$\rho(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t)=\begin{cases}\rho(\boldsymbol{r}_i)+(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t-\boldsymbol{r}_i)\cdot\nabla\rho(\boldsymbol{r}_i),\\ \qquad\text{when } \boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t \text{ is in the cell } \Omega_i\\ \rho(\boldsymbol{r}_{i+1})+(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t-\boldsymbol{r}_{i+1})\cdot\nabla\rho(\boldsymbol{r}_{i+1}),\\ \qquad\text{when } \boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t \text{ is in the cell } \Omega_{i+1}\end{cases} \tag{39}$$

$$\boldsymbol{u}(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t)=\begin{cases}\boldsymbol{u}(\boldsymbol{r}_i)+(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t-\boldsymbol{r}_i)\cdot\nabla\boldsymbol{u}(\boldsymbol{r}_i),\\ \qquad\text{when } \boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t \text{ is in the cell } \Omega_i\\ \boldsymbol{u}(\boldsymbol{r}_{i+1})+(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t-\boldsymbol{r}_{i+1})\cdot\nabla\boldsymbol{u}(\boldsymbol{r}_{i+1}),\\ \qquad\text{when } \boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t \text{ is in the cell } \Omega_{i+1}\end{cases} \tag{40}$$

With computed $\rho(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t)$ and $\boldsymbol{u}(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t)$ by Eqs. (39, 40), $f_\alpha^{eq}(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t,t-\delta_t)$ can be calculated by Eq. (27). Now, we are only left to determine $f_\alpha^{eq}(\boldsymbol{r},t)$ as shown in Eq. (38). Again, with Eq. (27), the calculation of $f_\alpha^{eq}(\boldsymbol{r},t)$ is equivalent to computing $\rho(\boldsymbol{r},t)$ and $\boldsymbol{u}(\boldsymbol{r},t)$. Using Eqs. (21, 22), the conservative variables ρ and $\rho\boldsymbol{u}$ can be computed by

$$\rho(\boldsymbol{r},t)=\sum_{\alpha=0}^{N}f_\alpha(\boldsymbol{r},t) \tag{41}$$

$$\rho(\boldsymbol{r},t)\boldsymbol{u}(\boldsymbol{r},t)=\sum_{\alpha=0}^{N}f_\alpha(\boldsymbol{r},t)\boldsymbol{e}_\alpha \tag{42}$$

Since f_α can be written as $f_\alpha^{eq}+f_\alpha^{neq}$, application of Eq. (20) at the cell interface leads to

$$f_\alpha(\boldsymbol{r},t)=f_\alpha^{eq}(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t,t-\delta_t)+\left(1-\frac{1}{\tau}\right)f_\alpha^{neq}(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t,t-\delta_t) \tag{43}$$

Furthermore, by substituting Eq. (38) into Eq. (43), we obtain

$$f_\alpha(\boldsymbol{r},t)=(1-\tau)f_\alpha^{eq}(\boldsymbol{r},t)+\tau f_\alpha^{eq}(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t,t-\delta_t) \tag{44}$$

Equation (44) is actually equivalent to $f_\alpha(\boldsymbol{r},t)=f_\alpha^{eq}(\boldsymbol{r},t)+f_\alpha^{neq}(\boldsymbol{r},t)$. Finally, Summation of Eq. (44) over α and applying the compatibility condition gives

$$\rho(\boldsymbol{r},t)=\sum_{\alpha=0}^{N}f_\alpha^{eq}(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t,t-\delta_t) \tag{45}$$

$$\rho(\boldsymbol{r},t)\boldsymbol{u}(\boldsymbol{r},t)=\sum_{\alpha=0}^{N}f_\alpha^{eq}(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t,t-\delta_t)\boldsymbol{e}_\alpha \tag{46}$$

where $f_\alpha^{eq}(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t,t-\delta_t)$ has already been calculated previously. Once $f_\alpha^{eq}(\boldsymbol{r}-\boldsymbol{e}_\alpha\delta_t,t-\delta_t)$ and $f_\alpha^{eq}(\boldsymbol{r},t)$ are obtained, f_α^{neq} can be approximated using Eq. (38), and f_α^{*} can be easily computed from Eq. (35).

Eqs. (45, 46) show that the conservative flow variables at the cell interface are fully determined from the equilibrium distribution functions at the surrounding points. As equilibrium distribution functions only depend on the macroscopic flow variables, there is no need to store the densi-

ty distribution functions for all the time levels. In fact, at any time step, we locally reconstruct a LBM solution at each cell interface independently. The reconstruction process is applied locally and repeated from one time level to another time level. Overall, the basic solution procedure of LBFS can be summarized below:

(1) At beginning, we need to choose a lattice velocity model such as D2Q9 model. Then we need to specify a streaming time step δ_t. The choice of δ_t should satisfy the constraint that the location of $(\boldsymbol{r}-\boldsymbol{e}_a\delta_t)$ must be within either the cell Ω_i or the cell Ω_{i+1}. Note that as local LBM solution is reconstructed at each cell interface, different interfaces could use different δ_t. This provides a great flexibility for application if we use non-uniform mesh or solve problems with a curved boundary. Once δ_t is chosen, the single relaxation parameter τ in LBFS is calculated by Eq. (28).

(2) For the considered interface position $\boldsymbol{r}$, identify its surrounding positions $(\boldsymbol{r}-\boldsymbol{e}_a\delta_t)$, and then use Eqs. (39, 40) to compute the macroscopic flow variables at those positions.

(3) Use Eq. (27) to calculate the equilibrium density distribution function $f_a^{\mathrm{eq}}(\boldsymbol{r}-\boldsymbol{e}_a\delta_t, t-\delta_t)$.

(4) Compute the macroscopic flow variables at the cell interface by using Eqs. (45) and (46), and further calculate $f_a^{\mathrm{eq}}(\boldsymbol{r},t)$ by Eq. (27).

(5) Calculate $f_a^{n\mathrm{eq}}(\boldsymbol{r},t)$ by using Eq. (38).

(6) Compute the fluxes at the cell interface by Eq. (36).

(7) Once fluxes at all cell interfaces are obtained, solve ordinary differential Eq. (34) by using 4-stage Runge-Kutta scheme.

It is indicated that the present LBFS can be used to simulate both incompressible viscous flows and incompressible inviscid flows. For the inviscid flow, we just simply set $\tau=0.5$. Another point to note is that the time marching step used in solving Eq. (34) and the streaming time step δ_t used in LBFS are independent. δ_t can be selected differently at different interface and different time level. Numerical experiments show that δ_t has no effect on the solution accuracy.

4 Numerical Examples and Discussion

In this section, the developed LBFS is validated by its application to solve some test problems. In all following simulations, the non-free parameter D1Q4 model[33-34] is used for simulation of compressible inviscid flows, and the D2Q9 lattice velocity model is applied for simulation of two-dimensional incompressible viscous flows.

4.1 Simulation of two-dimensional compressible inviscid flows

At first, the LBFS developed in Section 2 will be applied to simulate three two-dimensional compressible inviscid flows. They are the flow around a NACA0012 airfoil, the flow around a forward facing step, and the flow around a circular cylinder. For the flow around the NACA0012 airfoil, the free-stream Mach number is taken as 0.8 and the angle of attack is chosen as 1.25°. Unstructured grid with 10 382 cells is used for numerical computation. Both LBFS and Roe scheme are applied to solve this problem on the same computational mesh. It was found that the pressure coefficient distributions obtained by LBFS and Roe scheme are close to each other. The lift and drag coefficients (C_l and C_d) obtained by LBFS are respectively 0.304 1 and 0.023 7, which agree well with the results given from Roe scheme ($C_l=0.283\ 6$, $C_d=0.021\ 5$) and those of Stolcis and Johnston[35] ($C_l=0.339\ 7$, $C_d=0.022\ 8$). Fig. 5 shows the pressure contours around the airfoil. As can be seen clearly, the shock wave on the upper surface is well captured by present solver. The second test example in this part is a stationary flow (Mach number equals 3) hitting a rectangular step. This problem has been well studied by Woodward and Colella[36], and is often used to investigate performance of new numerical methods for capturing the

shock waves. In our computation, a uniform mesh size of 300×100 is used. Fig. 6 shows the density contours computed by present solver. Our results are in good agreement with those in Ref. [36]. It is noted that no special treatment around step corner is made in the present computation, which is often needed by conventional schemes. To further explore the capability of present solver for simulation of hypersonic flows with strong shock waves, the flow around a circular cylinder is simulated. For this case, a uniform mesh size of 160×40 in the cylindrical coordinate system is used. It is well known that for this problem, conventional numerical schemes such as Roe scheme may encounter the "carbuncle phenomenon" in front of cylinder when the free stream Mach number is high. The "carbuncle phenomenon" may be due to unsatisfying of entropy condition and negative value of density in the local region. We have used different free-stream Mach numbers to test simulation of this problem by LBFS. For all the cases tested (free-stream Mach number up to 100), no "carbuncle phenomenon" was found in the present results. This can be seen clearly from Fig. 7, which shows pressure contours of $Ma=3$ and 100. Both results show regular pressure distribution around the cylinder.

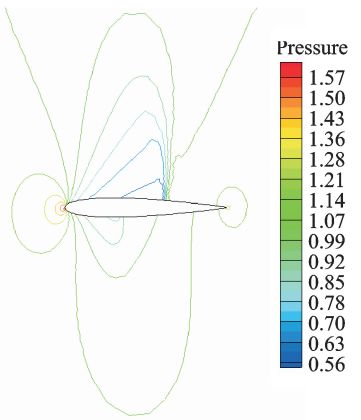

Fig. 5 Pressure contours around NACA0012 airfoil

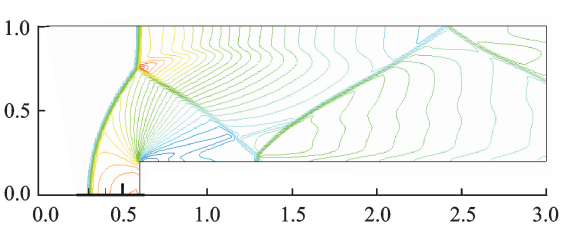

Fig. 6 Density contours for flow around a forward facing step

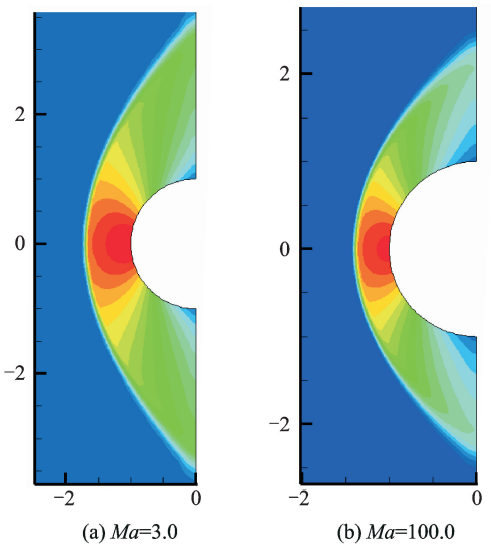

Fig. 7 Pressure contours for flow around a circular cylinder

4.2 Simulation of compressible inviscid flows around ONERA M6 wing

To investigate the capability of present LBFS for solving practical flow problems, the three-dimensional (3D) transonic flow around the ONERA M6 wing is simulated. This is also a standard test case for 3D computations. For numerical simulation, the free-stream Mach number is taken as 0.839 5 and the angle of attack is chosen as 3.06°. The part of computational mesh is shown in Fig. 8, which has 294 912 cells. The pressure contours obtained by present solver are

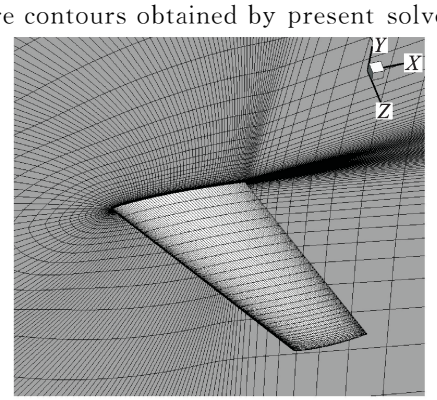

Fig. 8 Partial view of computational mesh for flow around ONERA M6 wing

displayed in Fig. 9. The "λ" shape shock wave on the upper surface of the wing can be seen clearly in Fig. 9, which is in line with the result in Ref. [37]. The pressure coefficient distribution at a section of $z/b=0.65$ is shown in Fig. 10. Also included in Fig. 10 are the experimental data given in Ref. [38]. As can been clearly, the present results quantitatively compare very well with the experimental data.

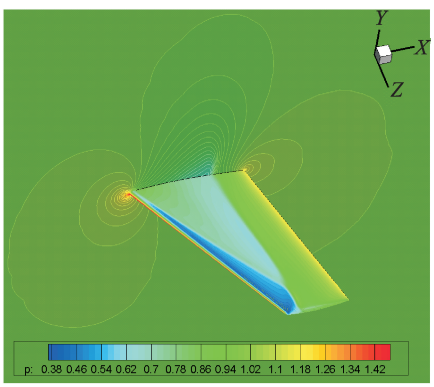

Fig. 9 Pressure contours around ONERA M6 wing

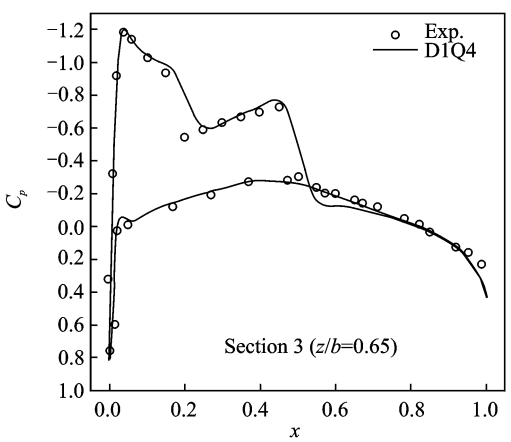

Fig. 10 Pressure coefficient distribution at section of $z/b=0.65$ on M6 wing

4.3 Simulation of incompressible lid-driven flow in a square cavity

The lid-driven flow in a square cavity is a standard test case for validating new numerical methods in simulation of incompressible viscous flows. The flow pattern of this problem is governed by the Reynolds number defined by $Re=UL/\nu$, where U is the lid speed, L is the length of the cavity, and ν is the kinematic viscosity of fluid. Two cases of this problem at moderate and high Reynolds numbers of 3 200 and 7 500 are considered in this work. LBFS introduced in Section 3 will be applied to solve this problem and the following problems.

To conduct numerical simulations, the non-uniform grid is generated according to the following formulation

$$x_i=\frac{1}{2}\left[1-\cos\left(\frac{i-1}{N-1}\pi\right)\right],\quad i=1,2,\cdots,N \tag{47a}$$

$$y_j=\frac{1}{2}\left[1-\cos\left(\frac{j-1}{M-1}\pi\right)\right],\quad j=1,2,\cdots,M \tag{47b}$$

where N and M are the total number of mesh points in the x and y directions respectively. With Eq. (47), the non-uniform grids of 101×101 for $Re=3\ 200$ and 121×121 for $Re=7\ 500$ are used respectively. In the present study, we set $U=0.1$ and $L=1$. The initial flow field is at rest.

Table 1 compares the locations of the primary vortex centers at $Re=3\ 200$ and 7 500 obtained by LBFS with those given by Ghia et al[39]. As can be seen, the maximum relative error between present results and those of Ghia et al[39] is less than 1.1%. Fig. 11 displays u-velocity profile along the horizontal centerline and v-velocity profile along the vertical centerline of the considered two cases. As can be seen from this figure, the present results agree very well with those of Ghia et al[39]. Fig. 12 shows the streamlines of $Re=3\ 200$, 7 500. The most striking aspect of this figure is that the Reynolds number apparently has unique effect on flow patterns. Secondary and tertiary vortices appear and evolve into larger ones as Re becomes large. These results and observations are in good agreement with those of Ghia et al[39].

Table 1 Locations of primary vortex centers at different Reynolds numbers

Re	Vortex center (x, y)	
	Ref. [39]	Present
3 200	(0.516 5, 0.546 9)	(0.518 6, 0.5412)
7 500	(0.511 7, 0.532 2)	(0.514 6, 0.533 6)

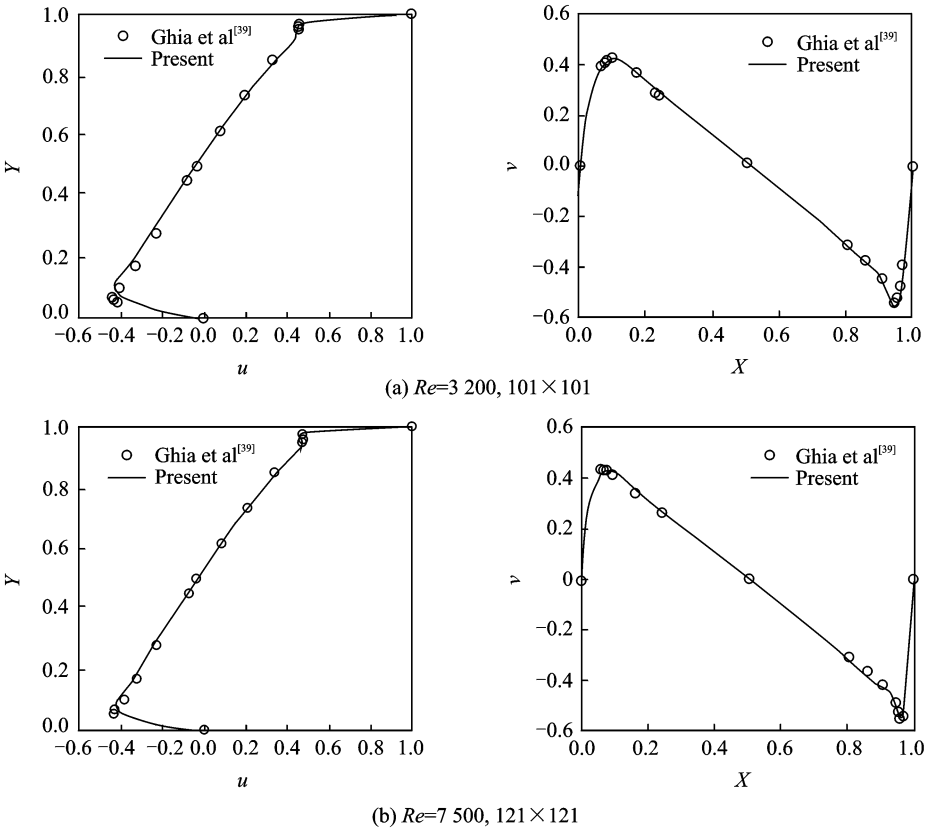

Fig. 11 u and v velocity profiles along horizontal and vertical centerlines for a lid-driven cavity flow at Re=3 200, 7 500

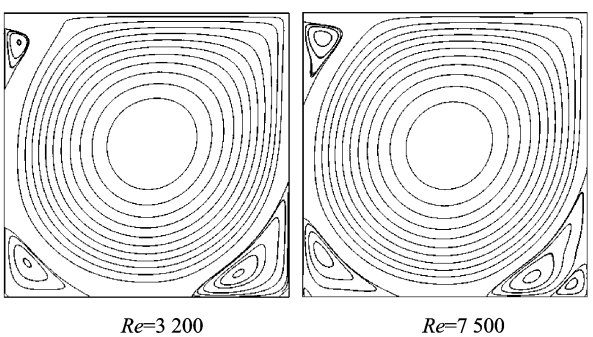

Fig. 12 Streamlines of a lid-driven cavity flow at Re= 3 200, 7 500

Note that for this test example, we have also studied the effect of streaming distance in local reconstruction of LBM solution. It was found that when the streaming distance is less than half of mesh spacing in the two neighboring cells (this constraint guarantees that only interpolation is performed in each cell), any value of streaming distance will have no effect on the accuracy of solution. This is an appealing feature, which ensures that LBFS can be easily applied on non-uniform mesh.

4.4 Simulation of incompressible polar cavity flow

Although the complex lid-driven cavity flows have been successfully simulated to validate the present solver, the geometry of the cavity which only involves straight boundaries is nevertheless simple. To further illustrate the capability of LBFS for problems with curved boundary, a polar cavity flow is simulated on body-fitted meshes. The schematic diagram and the typical non-uniform mesh for this problem are depicted in Fig. 13. As shown in Fig. 13, a sector with an angle of θ=1 is bounded by two straight walls and two curved walls with radii of R_i and R_o. The inner curved wall rotates with an azimuthal velocity of U_θ. The flow pattern of this problem is governed by the Reynolds number defined as $Re = U_\theta R_i/\nu$. In this study, two cases of Re=60 and 1 000 are considered, and the following parameters are applied: R_i=1.0, R_o=2.0, ρ_0=1.0 and U_θ = 0.1. Initially, the flow field is at rest. Fig. 14 shows the radial (u_r) and azimuthal (u_θ)

velocity profiles along the horizontal line of $\theta=0.5$ at $Re=60$ and 1 000. The experimental results of Fuchs and Tillmark[40] and the numerical solutions of Shu et al[41] obtained by applying Taylor series expansion- and least-square-based LBM (TLLBM) are also included for comparison. Note that the present results and TLLBM results are both obtained on the same non-uniform grids, i.e., 61×61 for $Re=60$ and 81×81 for $Re=1\,000$. It can be seen that good agreements have been achieved between the present results and those of Fuchs and Tillmark[40] and Shu et al[41], which validate the reliability of the present solver for problems with curved boundary and use of non-uniform grid. The streamlines are shown in Fig. 15. As can be seen, with increase of the Reynolds number, the primary vortex moves upward and reduces its size. At the same time, the two secondary vortices at the upper-right and lower-right corners enlarge their size. These observations agree well with those of Fuchs and Tillmark[40].

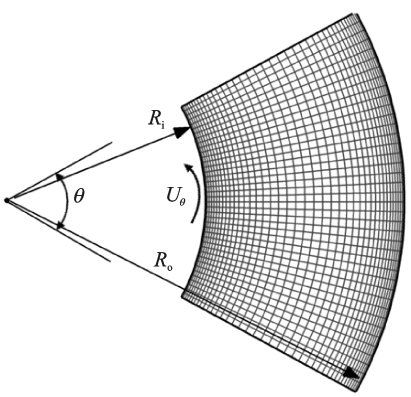

Fig. 13 Schematic diagram and a typical body-fitted mesh for flow in a polar lid-driven cavity

4.5 Simulation of flow induced by an impulsively started cylinder

In this part, LBFS is applied to simulate the unsteady flow induced by an impulsively started circular cylinder. The Reynolds number of this flow is defined as $Re=UD/\nu$, where U is the free-stream velocity and D is the diameter of cylinder. A wide range of Reynolds numbers from 10^2 to 10^4 are considered in this study to further demonstrate the capability of LBFS for effective simulation of unsteady flows at high Reynolds numbers.

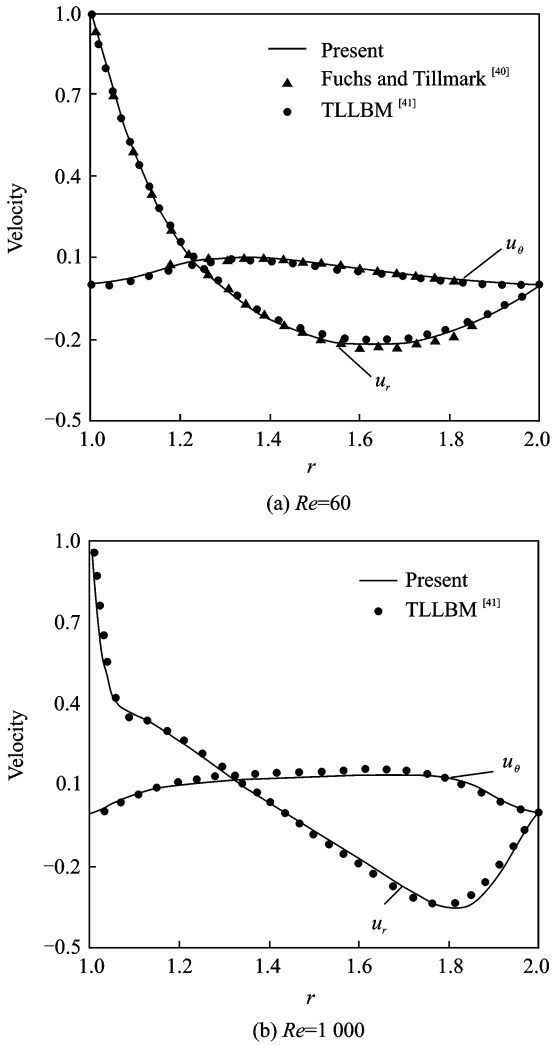

Fig. 14 Comparison of radial (u_r) and azimuthal (u_θ) velocity profiles along the horizontal line of $\theta=0.5$ for the polar cavity flow

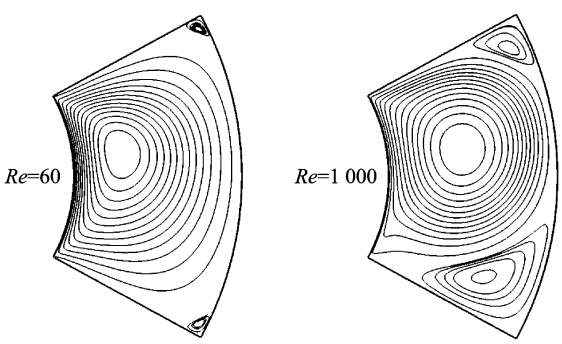

Fig. 15 Streamlines for the polar cavity flow at $Re=60$ and 1 000

In the present simulation, for flows at $Re=550$ and 3 000, a mesh size of 301×201 is used and the outer boundary is placed at 15 diam-

eters away from the cylinder center. For the flow at $Re=9\ 950$, the computational mesh is set as 301×351 and the outer boundary is set as 4 diameters away from the cylinder center. The flow parameters are set as: $\rho=1.0$, $U=0.1$ and $a=0.5$. Initially, the flow field is at rest.

For incompressible flows around the circular cylinder at high Reynolds number, a pair of primary symmetric vortices will be developed at the rear of cylinder initially. With increase of the Reynolds number, the size of the two vortices is decreased. As time increases, the primary vortices will move away and detach from the rear of cylinder. In the meantime, a pair of secondary symmetric vortices appears and becomes larger and stronger. The vortex structures exhibit the so-called "α" and "β" patterns. All these features have been well captured in present simulation. To save the space, these results are not displayed in this paper. Fig. 16 shows a quantitative comparison of the time evolution of the vortex length with experimental data of Bouard and Coutanceau[42]. Obviously, good agreement has been achieved. For $Re=550$, the vortex length almost grows linearly with respect to time. For high Reynolds numbers ($Re=3\ 000$ and $9\ 500$), a slow increase in vortex length, which corresponds to the "fore-wake" region, can be observed when $t<3.0$ s. When $t>3$ s, a fast growth of the vortex length can be seen due to destruction of the "fore-wake". Fig. 17 further compares the radial velocity along the symmetric axis at $Re=3\ 000$ with experimental data of Bouard and Coutanceau[42] and numerical results of Niu et al[43]. Once again, good agreement is achieved.

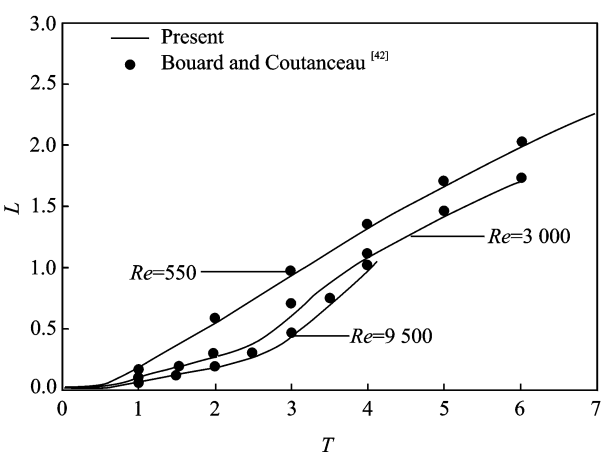

Fig. 16 Comparison of the vortex length for flow induced by impulsively started cylinder at different Reynolds numbers

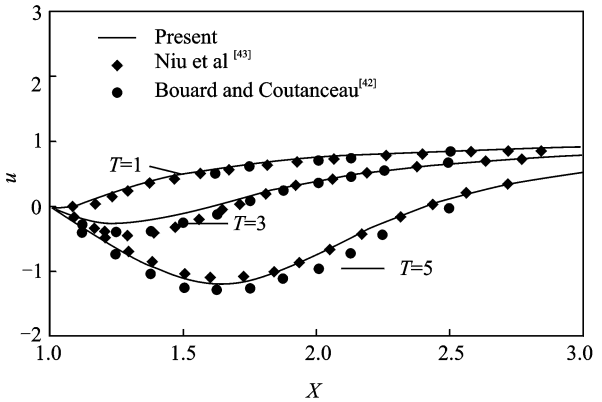

Fig. 17 Comparison of the radial velocity along symmetric axis for flow induced by impulsively started cylinder at $Re=3\ 000$

5 Conclusions

In this paper, the LBFS is presented for simulation of compressible and incompressible flows. The solver is based on numerical discretization of FVM to the governing differential equations (Navier-Stokes equations or Euler equations). Specifically, the conservative flow variables at cell centers are given from the solution of discrete governing equations but numerical fluxes at cell interfaces are evaluated by local reconstruction of LBE solution from flow variables at cell centers. Two versions of LBFS are presented in this paper. One is to locally apply 1D compressible LB model along the normal direction to the cell interface for simulation of compressible inviscid flows. The other is to locally apply incompressible LB model at the cell interface for simulation of incompressible viscous flows.

The present LBFS is well validated by its application to simulate some two- and three-dimensional compressible inviscid flows, and two-dimensional incompressible viscous flows. Numerical results show that the compressible version of LBFS can well simulate compressible inviscid flows with strong shock waves, and its in-

compressible version can accurately simulate incompressible viscous flows with curved boundary and non-uniform mesh. It removes the drawbacks of conventional LBM such as limitation to the uniform mesh, tie-up of mesh spacing and time interval. It is believed that LBFS has a great potential for solving various flow problems in practice.

References:

[1] Roach P J. Computational fluid dynamics[M]. Hermosa Beach, USA: Hermosa Press, 1972.

[2] Anderson D A, Tannehill J C, Pletcher R H. Computational fluid mechanics and heat transfer[M]. New York, USA: McGraw-Hill, 1984.

[3] Hirsch C. Numerical computation of internal and external flows[M]. Hoboken, USA: John Wiley & Sons, 1988.

[4] Fletcher C A J. Computational techniques for fluid dynamics: fundamental and general techniques[M]. Berlin, Germany: Springer-Verlag, 1991.

[5] Anderson J D. Computational fluid dynamics: the basics with applications[M]. New York, USA: McGraw-Hill, 1995.

[6] Versteeg H K, Malalasekera W. An introduction to computational fluid dynamics: the finite volume method[M]. Harlow, England: Longman Scientific & Technical, 1995.

[7] Donea J, Huerta A. Finite element methods for flow problems[M]. Hoboken, USA: John Wiley, 2003.

[8] Wendt J F. Computational fluid dynamics[M]. Berlin, Germany: Springer Berlin Heidelberg, 2009.

[9] Funaro D. Polynomial approximation of differential equations[M]. Berlin, Germany: Springer-Verlag, 1992.

[10] Buhmann M D. Radial basis functions: theory and implementations[M]. Cambridge University Press, 2003.

[11] Godunov S K. A difference method for numerical calculation of discontinuous solutions of the equations of hydrodynamics[J]. Matematicheskii Sbornik, 1959, 47: 271-306.

[12] Roe P L. Approximate Riemann solvers, parameter vectors, and difference schemes[J]. Journal of Computational Physics, 1981, 43: 357-372.

[13] Steger J, Warming R. Flux vector splitting of the inviscid gas dynamic equations with applications to finite-difference methods[J]. Journal of Computational Physics, 1981, 40: 263-293.

[14] Shu C W, Osher S. Efficient implementation of essentially non-oscillatory shock-capturing scheme[J]. Journal of Computational Physics, 1988, 77: 439-471.

[15] Shu C W. High order weighted essentially non-oscillatory schemes for convection dominated problems [J]. SIAM Review, 2009,51:82-126.

[16] B van Leer, Lo M. A discontinuous Galerkin method for diffusion based on recovery[J]. Journal of Scientific Computation, 2011,46:314-328.

[17] Xu K. A gas-kinetic BGK scheme for the Navier-Stocks equations and its connection with artificial dissipation and Godunov method[J]. J Comput Phys, 2001,171:289-335.

[18] Chen S Z, Xu K, Lee C B, et al. A unified gas kinetic scheme with moving mesh and velocity space adaptation[J]. Journal of Computational Physics, 2012, 231:6643-6664.

[19] Yang L M, Shu C, Wu J, et al. Circular function-based gas-kinetic scheme for simulation of inviscid compressible flows[J]. J Comput Phy, 2013, 255: 540-557.

[20] Chen S, Chen H, Martínez D, et al. Lattice Boltzmann model for simulation of magnetohydrodynamics [J]. Phys Rev Let, 1991,67(27):3776-3779.

[21] Qian Y H, D′ Humières D, Lallemand P. Lattice BGK models for Navier-Stokes equation[J]. Europhys Lett, 1992,17:479-484.

[22] Chen S, Doolen G. Lattice Boltzmann method for fluid flows[J]. Ann Rev Fluid Mech, 1998,30:329-64.

[23] Mei R, Luo L S, Shyy W. An accurate curved boundary treatment in the lattice Boltzmann method [J]. J Comput Phys, 1999,155:307-330.

[24] Guo Z L, Shi B C, Wang N C. Lattice BGK model for incompressible Navier-Stokes equation[J]. J Comput Phys, 2000,165:288-306.

[25] Shu C, Chew Y T, Niu X D. Least square-based LBM: a meshless approach for simulation of flows with complex geometry[J]. Phys Rev E, 2001:64, 045701.

[26] Shu C, Niu X D, Chew Y T. Taylor series expansion- and least square-based lattice Boltzmann method: two-dimensional formulation and its applications [J]. Phys Rev E, 2002,65:036708 .

[27] Succi S, Mesoscopic modeling of slip motion at fluid-solid interfaces with heterogeneous catalysis[J]. Phys Rev Lett, 2002,89:064502.

[28] Shan X, Yuan X F, Chen H. Kinetic theory representation of hydrodynamics: a way beyond the Navi-

er-Stokes equation[J]. J Fluid Mech, 2006,550:413-441.

[29] Guo Z L, Asinari P, Zheng C G. Lattice Boltzmann equation for microscale gas flows of binary mixtures [J]. Phys Rev E, 2009,79:026702.

[30] Aidun C K, Clausen J R. Lattice-Boltzmann method for complex flows[J]. Ann Rev Fluid Mech, 2010, 42:439-72.

[31] Wu J, Shu C. A solution-adaptive lattice Boltzmann method for two-dimensional incompressible viscous flows[J]. J Comput Phys, 2011,230:2246-2269.

[32] Guo Z, Shu C. Lattice Boltzmann method and its applications in engineering[J]. World Scientific Publishing, 2013.

[33] Yang L M, Shu C, Wu J. Development and comparative studies of three non-free parameter lattice Boltzmann models for simulation of compressible flows [J]. Adv Appl Math Mech, 2012,4:454-472.

[34] Yang L M, Shu C, Wu J. A moment conservation-based non-free parameter compressible lattice Boltzmann model and its application for flux evaluation at cell interface[J]. Comput Fluids, 2013,79:190-199.

[35] Stolcis L, Johnston L J. Solution of the Euler equations on unstructured grids for two-dimensional compressible flow[J]. Aeronautical Journal, 1990, 94: 181-195.

[36] Woodward P, Colella P. The numerical simulation of two-dimensional fluid flow with strong shocks[J]. Journal of Computational Physics, 1984,54:115-173.

[37] Batina J T. Accuracy of an unstructured-grid upwind-Euler algorithm for the ONERA M6 wing[J]. J Aircraft, 1991,28:397-402.

[38] Schmitt V, Charpin F. Pressure distributions on the ONERA-M6-wing at transonic Mach numbers, experimental data base for computer program assessment[J]. Report of the Fluid Dynamics Panel Working Group 04, 1979, AGARD AR:138.

[39] Ghia U, Chia K N, Shin C T. High-resolutions for incompressible flow using the Navier-Stokes equations: a multigrid method [J]. J Comput Phys, 1982,48:387-411.

[40] Fuchs L, Tillmark N. Numerical and experimental study of driven flow in a polar cavity[J]. Int J Num Methods in Fluids, 1985,5:311-329.

[41] Shu C, Niu X D, Chew Y T. Taylor series expansion- and least square-based lattice Boltzmann method: two-dimensional formulation and its applications [J]. Phys Rev E, 2002,65:036708.

[42] Bouard R, Coutanceau M. The early stage of development of the wake behind an impulsively started cylinder for $40<Re<10^4$ [J]. J Fluid Mech, 1980, 101:583-607.

[43] Niu X D, Chew Y T, Shu C. Simulation of flows around an impulsively started circular by Taylor series expansion- and least squares-based lattice Boltzmann method[J]. J Comput Phys, 2003,188:176-193.

(Executive editor: Zhang Tong)