Advances in Applied Mathematics and Mechanics
Adv. Appl. Math. Mech., Vol. **11**, No. 1, pp. 72-90

DOI: 10.4208/aamm.OA-2018-0167
February 2019

An Implicit Block LU-SGS Algorithm-Based Lattice Boltzmann Flux Solver for Simulation of Hypersonic Flows

Z. X. Meng[1], S. Z. Li[2,*], K. Peng[1] and W. H. Zhang[1]

[1] *College of Aerospace Science and Engineering, National University of Defense Technology, Changsha 410073, China*
[2] *National Innovation Institute of Defense Technology, Academy of Military Science, Beijing 100091, China*

Received 17 July 2018; Accepted (in revised version) 3 September 2018

Abstract. This paper proposes a stable and efficient implicit block Lower-Upper Symmetric-Gauss-Seidel (LU-SGS) algorithm-based lattice Boltzmann flux solver (LBFS) for simulation of hypersonic flows. In this method, the finite volume method (FVM) is applied to discretize the Navier-Stokes equations, and the LBFS is utilized to evaluate the numerical flux at the cell interface. In LBFS, the local solution of discrete velocity Boltzmann equation (DVBE) with the non-free parameter D1Q4 lattice Boltzmann model is adopted to reconstruct the inviscid flux across the cell interface, and the viscous flux is approximated by conventional smooth function approach. In order to improve the robustness and convergence rate of the simulation for hypersonic flows, especially for problems with complex geometry, the implicit block LU-SGS algorithm is introduced to solve resultant discrete governing equations. A double cone model at Mach number of $Ma=9.86$ is firstly simulated to validate the proposed scheme, and a hypersonic flight vehicle with wings and rudders at Mach number of $Ma=5.56$ is then calculated to extend the application in practical engineering problems. Numerical results show that the proposed scheme could offer a more accurate and effective prediction for hypersonic flows.

AMS subject classifications: 76M12

Key words: Hypersonic flows, lattice Boltzmann flux solver, implicit block LU-SGS, finite volume method.

1 Introduction

The computational fluid dynamics (CFD) plays an important role in solving flow field problems due to the prosperous development in numerical algorithms and computa-

[*] Corresponding author.
Emails: lishengze12@gmail.com (S. Z. Li), mengzhuxuan08@nudt.edu.cn (Z. X. Meng)

tional facility. CFD provides lower cost than wind tunnel experiment and higher accuracy than engineering approximation method, which brings it widely industrial application. The finite volume method (FVM) is one of the most popular numerical methods in CFD due to its good numerical conservation properties and suitability in solving flow problems with complex geometry [1–3]. The key of FVM is to construct a flux solver to evaluate the numerical flux at the cell interface. Godunov [4] simplified the Euler equations to a Riemann problem firstly in 1959 and proposed an accurate Riemann solver for calculating the inviscid numerical flux. Subsequently, varieties of approximate Riemann solvers [5–9] were developed to improve the efficiency of the original one. Among them, the upwind schemes such as Roe scheme [5], AUSM (Advection Upstream Splitting Method) scheme [7] and van Leer scheme [8] have been widely used in evaluating the numerical flux for simulating the compressible flows. However, some of them often exhibit a carbuncle phenomenon and induce numerical instability during the hypersonic simulation [10].

Boltzmann equation-based flux solver is another popular method for calculation of the numerical flux. Different from the conventional CFD approaches, which only compute the flux by numerical approximation, the Boltzmann equation-based schemes evaluate the flux through a local reconstruction with the solution for Boltzmann equation at the cell interface. This feature makes the computation of Boltzmann equation-based schemes be robustness and effectively prevent unphysical solutions. One of the representative Boltzmann equation-based flux solvers is the gas-kinetic scheme (GKS) [11–13], which calculates the numerical flux by the local solution of continuous Boltzmann equation. In the works of Xu and his coworkers [14–17], the local integral solution of Boltzmann equation with Maxwellian distribution function is utilized to reconstruct the numerical flux. To simplify the Maxwellian function-based GKS [14–17], the circular function-based GKS (CGKS) for two-dimensional cases [18–20] and the sphere function-based GKS (SGKS) for three-dimensional cases [21–23] have been developed recently by Shu and his coworkers. Another representative Boltzmann equation-based flux solver is the lattice Boltzmann flux solver (LBFS), which adopts the local solution of discrete velocity Boltzmann equation (DVBE) with lattice Boltzmann model to reconstruct the numerical flux at the cell interface [24–29]. By using the discrete model, the LBFS is more convenient for both mathematical derivation and coding. It has been proven that the LBFS can provide an accurate and efficient prediction for both incompressible and compressible flows [28,29]. Nowadays, more applications such as turbomachinery flows [30] and multi-component flows [31] are implemented by LBFS, which make it becoming a popular solver. However, the application of LBFS in hypersonic flows, especially for complex geometry, is still rarely studied. Therefore, a further endeavor is needed to extend the application of LBFS to hypersonic flows.

In the simulation of hypersonic flows, the extreme low-pressure area usually occurrs behind the object [32,33]. This phenomenon may slow down the convergence rate significantly or even produce a divergent result with negative density and pressure, especially for complex geometry. To overcome this defect, during the hypersonic simulation, the implicit scheme is often used for temporal discretization due to its high stability and efficiency. Tlke et al. [34] proposed a fully implicit lattice Boltzmann scheme which is based on finite difference method (FDM) to improve the computational efficiency in high Reynolds number. Lee and Lin [35] extended this scheme into the Taylor-Galerkin finite element method (FEM). Huang et al. [36] further applied it into a general FDM. Recently, based on the conventional CFD schemes, Chen and Wang [37] developed an implicit block Lower-Upper Symmetric-Gauss-Seidel (LU-SGS) algorithm for solving flow problems with arbitrary grids. Subsequently, Li and Luo [38] introduced the implicit block LU-SGS scheme into the lattice Boltzmann method. It was found that the implicit block LU-SGS scheme can significantly speed up the efficiency and improve the robustness of computation, especially for hypersonic problems.

Due to nice features of the implicit block LU-SGS algorithm, it is introduced into the current LBFS for solving complex hypersonic flow problems. In the method, the Navier-Stokes equations are discretized by FVM in space and by implicit block LU-SGS scheme in time, and the numerical flux at the cell interface is computed by LBFS. In addition, in order to simulate turbulence flows, the Spalart-Allmaras one-equation turbulence model is adopted to evaluate the turbulent viscosity [39]. To validate the present method, a two-dimensional double cone model is first computed and compared with the available data. Subsequently, a hypersonic flight vehicle with wings and rudders in two-dimensional space is calculated to show the potential of the developed method for simulating practical engineering problems. Results show that the present scheme can predict the flow properties with high accuracy and efficiency in simulating the hypersonic flows. Overall, the developed scheme well combines the fine convergence and high efficiency properties of implicit block LU-SGS algorithm with the high computational accuracy of LBFS.

2 Methodology

In this section, an implicit block LU-SGS algorithm-based LBFS scheme is revealed. Here, the Navier-Stokes equations are solved on a macroscopic scale, where the governing equations are discretized by FVM and the conservative variables are defined at cell centers. To simulate hypersonic flows accurately and stably, the LBFS is adopted to evaluate the numerical flux at the cell interface. In LBFS, the flux is reconstructed by the local solution of DVBE with the non-free parameter D1Q4 lattice Boltzmann model. At the same time, the numerical dissipation is introduced by the collision term of DVBE and controlled by a switch function which is defined by the difference of pressure at the left and the right sides of cell interface. To further improve the robustness and convergent speed for solving hypersonic flow problems, the implicit block LU-SGS is introduced to solve resultant governing equations.

2.1 FVM discretization and D1Q4 lattice Boltzmann model

In general, the discretized form of the Navier-Stokes equations given by FVM can be written as

$$\frac{dW_I}{dt} = -\frac{1}{\Omega_I}\sum_{i=1}^{N_f}(F_{ci}-F_{vi})S_i, \tag{2.1}$$

where $W$is the vector of conservative variables, $F_c$is inviscid flux and F_v is viscous flux. The subscript I denotes the index of the control volume, Ω_I is the volume and N_f is the number of faces. It can be seen that the key to evolve the conservative variables at cell center is to calculate the numerical fluxes at the cell interface. In this work, the viscous flux F_v is calculated by the central difference scheme [1], and the inviscid flux F_c is evaluated by LBFS [28,29]. The details will be discussed in Subsection 2.2.

In this work, we focus on solving 2-D hypersonic problems. For the 2-D case, the conservative variables W at cell center and inviscid flux vector F_c at the cell interface can be expressed as

$$W = \begin{bmatrix} \rho \\ \rho(U_n n_x + u_{\tau x}) \\ \rho(U_n n_y + u_{\tau y}) \\ \rho(U_n^2/2+e)+\rho|u_\tau|^2/2 \end{bmatrix}, \tag{2.2a}$$

$$F_c = \begin{bmatrix} \rho U_n \\ (\rho U_n^2+p)n_x+\rho U_n u_{\tau x} \\ (\rho U_n^2+p)n_y+\rho U_n u_{\tau y} \\ (\rho(U_n^2/2+e)+p)U_n+\rho|u_\tau|^2/2 \end{bmatrix}, \tag{2.2b}$$

where ρ is the density, p is the pressure, e is the internal energy, U_n is the normal velocity, and $\mathbf{u}_\tau = (u_{\tau x}, u_{\tau y})$ is the tangential velocity. $\mathbf{n} = (n_x, n_y)$ denotes the unit normal vector on the control surface in the Cartesian coordinate system.

To use the LBFS for evaluating the numerical flux, the lattice Boltzmann model is required. Most of the existing lattice Boltzmann models [40–43] involve a number of user-specified parameters, which could significantly affect the performance of LBFS. To remove this drawback, Yang et al. [25,26] proposed a non-free parameter D1Q4 model. In this model, there are 4 equilibrium distribution functions f_1^{eq}, f_2^{eq}, f_3^{eq}, f_4^{eq} and 2 lattice velocities d_1, d_2, which are derived from conservation forms of moments and given by

$$\begin{cases} f_1^{eq} = \dfrac{\rho(-d_1d_2^2 - d_2^2u + d_1u^2 + d_1c^2 + u^3 + 3uc^2)}{2d_1(d_1^2-d_2^2)}, \\ f_2^{eq} = \dfrac{\rho(-d_1d_2^2 + d_2^2u + d_1u^2 + d_1c^2 - u^3 - 3uc^2)}{2d_1(d_1^2-d_2^2)}, \\ f_3^{eq} = \dfrac{\rho(d_1^2d_2 + d_1^2u - d_2u^2 - d_1c^2 - u^3 - 3uc^2)}{2d_2(d_1^2-d_2^2)}, \\ f_4^{eq} = \dfrac{\rho(d_1^2d_2 - d_1^2u - d_2u^2 - d_2c^2 + u^3 + 3uc^2)}{2d_2(d_1^2-d_2^2)}, \end{cases} \tag{2.3a}$$

$$\left\{\begin{array}{l} d_1=\sqrt{u^2+3c^2-\sqrt{4u^2c^2+6c^4}}, \\ d_2=\sqrt{u^2+3c^2+\sqrt{4u^2c^2+6c^4}}. \end{array}\right. \tag{2.3b}$$

Here, u is the velocity, $c=\sqrt{Dp/\rho}$ represents the particular velocity of particles, and D is the spatial dimension. $D=1$ is used in the D1Q4 model. The distribution of discrete lattice velocities for D1Q4 model is shown in Fig. 1.

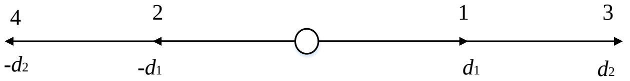

Figure 1: Lattice velocities distribution for non-free parameter D1Q4 model.

As shown above, the D1Q4 model is actually a one-dimensional model, only the flux in the normal direction is evaluated from the LBFS. When the multi-dimensional problems are considered, the D1Q4 model needs to be applied along the normal direction of cell interface, as shown in Fig. 2. Accordingly, the velocity u in Eq. (2.3) should be replaced by the normal velocity U_n. Then by defining the unit normal vector at the cell interface as (n_x,n_y), the relations between different directions of velocity in Fig. 2 can be written as

$$u=U_n n_x+u_{\tau x}, \quad v=U_n n_y+u_{\tau y}. \tag{2.4}$$

By using the D1Q4 model to the 2-D case, the physical conservation laws can be given by

$$\left\{\begin{array}{l} \rho=\sum\limits_{i=1}^{4} f_i^{eq}, \\ \rho U_n=\sum\limits_{i=1}^{4} f_i^{eq}\xi_i, \\ \rho U_n^2+p=\sum\limits_{i=1}^{4} f_i^{eq}\xi_i\xi_i, \\ \rho(U_n^2/2+e)=\sum\limits_{i=1}^{4} f_i^{eq}\left(\dfrac{1}{2}\xi_i\xi_i+e_p\right), \\ \left[\rho(U_n^2/2+e)+p\right]U_n=\sum\limits_{i=1}^{4} f_i^{eq}\left(\dfrac{1}{2}\xi_i\xi_i+e_p\right)\xi_i, \end{array}\right. \tag{2.5}$$

where ξ_i means particle velocity in i-direction and $e_p=\left[1-\frac{D}{2}(\gamma-1)\right]e$ is the potential energy of particles. γ is the specific heat ratio. Following the description in Fig. 1, we have $\xi_1=d_1$, $\xi_2=-d_1$, $\xi_3=d_2$ and $\xi_4=-d_2$.

2.2 LBFS for evaluation of numerical flux

Suppose that the cell interface is located at $x=0$, then the distribution function at the cell interface can be written as

$$f_i(0,t)=f_i^{eq}(0,t)-f_i^{neq}(0,t), \tag{2.6}$$

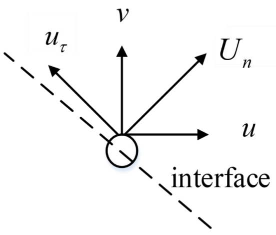

Figure 2: Application of D1Q4 model in 2-D case.

where $f_i^{neq}(0,t)$ is the non-equilibrium part of the distribution function. According to Chapman-Enskog analysis, the Boltzmann equation can recover the Navier-Stokes equations [44,45], and the non-equilibrium part can be expressed as

$$f_i^{neq}(0,t)=-\tau\left(\frac{\partial f_i^{eq}}{\partial t}+\xi_i\frac{\partial f_i^{eq}}{\partial x}\right)\bigg|_{(0,t)}. \tag{2.7}$$

By applying Taylor series expansion along the characteristic direction of Boltzmann equation, the non-equilibrium part could be further approximated by

$$f_i^{neq}(0,t)=-\frac{\tau}{\delta t}\left[f_i^{eq}(0,t)-f_i^{eq}(-\xi_i\delta t,t-\delta t)\right]+\mathcal{O}({\xi_i}^2\delta t^2,\tau\delta t). \tag{2.8}$$

Here, $f_i^{eq}(0,t)$ is equal to equilibrium distribution function at the cell interface and $f_i^{eq}(-\xi_i\delta t,t-\delta t)$ is the equilibrium distribution function at the surrounding point of the cell interface. Substituting Eq. (2.8) into Eq. (2.6), we have

$$f_i(0,t)=f_i^{eq}(0,t)-\tau_0\left[f_i^{eq}(0,t)-f_i^{eq}(-\xi_i\delta t,t-\delta t)\right]+\mathcal{O}({\xi_i}^2\delta t^2,\tau\delta t), \tag{2.9}$$

where $\tau_0=\tau/\delta t$ is the dimensionless collision time, which will be determined subsequently.

In the present scheme, the LBFS is only used to evaluate inviscid flux. Thus, the non-equilibrium part can be viewed as numerical dissipation, and τ_0 can be treated as the weight of numerical dissipation. According to relationships (2.5), at the cell interface, the inviscid flux can be computed by

$$\begin{aligned}\mathbf{F}_c=&\sum_{i=1}^{4}\xi_i\varphi_a f_i^{eq}(0,t)+\tau_0\left[\sum_{i=1}^{4}\xi_i\varphi_a f_i^{eq}(-\xi_i\delta t,t-\delta t)-\sum_{i=1}^{4}\xi_i\varphi_a f_i^{eq}(0,t)\right]\\ =&F_c+\tau_0(F_c-F_c).\end{aligned} \tag{2.10}$$

It should be noticed that the moments φ_a is consists of three terms when applying the D1Q4 model. But if a global 2-D coordinate system is considered, the moments φ_a will include four terms, because the normal velocity in 1-D case is decomposed into two parts

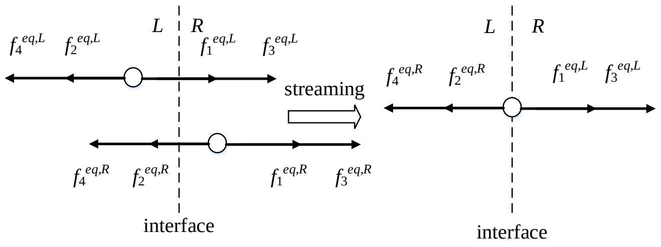

Figure 3: Streaming process of non-free parameter D1Q4 model at the cell interface.

of the x and y directions. By applying the expressions in Eq. (2.4), the moments φ_a are given by

$$\varphi_a = \left[1, \xi_i n_x + u_{\tau x}, \xi_i n_y + u_{\tau y}, \frac{1}{2}\left(\xi_i^2 + |u_\tau|^2\right) + e_p\right]^T. \tag{2.11}$$

It can be seen from Eq. (2.10) that the inviscid flux consists of two parts: F_c is produced by the equilibrium distribution function at the cell interface $f_i^{eq}(0,t)$, and F_c is attributed to the equilibrium distribution function at the surrounding points of the interface $f_i^{eq}(-\xi_i\delta t, t-\delta t)$.

Like the conventional upwind schemes, it is assumed that a Riemann problem with the pricewise constant distribution of the distribution functions is formed at the cell interface, as shown in the left side of Fig. 3. Thus, the equilibrium distribution function $f_i^{eq}(-\xi_i\delta t, t-\delta t)$ can be given according to the location of $-\xi_i\delta t$ as follows:

$$f_i^{eq}(-\xi_i\delta t, t-\delta t) = \begin{cases} f_i^{eq,L} & -\xi_i\delta t \le 0, \\ f_i^{eq,R} & -\xi_i\delta t > 0. \end{cases} \tag{2.12}$$

More specifically, for D1Q4 model, Eq. (2.12) can be rewritten as

$$f_i^{eq}(-\xi_i\delta t, t-\delta t) = \begin{cases} f_i^{eq,L}, & i=1,3, \\ f_i^{eq,R}, & i=2,4. \end{cases} \tag{2.13}$$

This process is depicted in Fig. 3. In fact, it is equivalent to solve the collisionless Boltzmann equation at the cell interface.

To calculate the equilibrium distribution function $f_i^{eq}(0,t)$, the conservative variables at the cell interface should be computed in advance.

According to the compatibility condition [28], the non-equilibrium part of distribution function does not contribute to the conservative variables. As a result, the conservative variables at the cell interface can be computed by

$$\mathbf{W}_{j+1/2} = \sum_{i=1}^{4} \varphi_a f_i^{eq}(0,t) = \sum_{i=1}^{4} \varphi_a f_i^{eq}(-\xi_i\delta t, t-\delta t). \tag{2.14}$$

By substituting Eq. (2.13) into Eq. (2.14), we have

$$\mathbf{W}_{j+1/2}=\sum_{i=1,3}\varphi_a f_i^{eq,L}+\sum_{i=2,4}\varphi_a f_i^{eq,R}. \tag{2.15}$$

Once $\mathbf{W}_{j+1/2}$ is obtained, the equilibrium distribution function at the cell interface $f_i^{eq}(0,t)$ can be calculated by substituting the flow variables into Eq. (2.3) and the flux F_c can be computed by substituting $f_i^{eq}(0,t)$ into Eq. (2.10).

Finally, the two parts of inviscid flux at the cell interface can be computed by

$$F_c=\begin{bmatrix}\rho U_n\\ (\rho U_n^2+p)n_x+\rho U_n u_{\tau x}\\ (\rho U_n^2+p)n_y+\rho U_n u_{\tau y}\\ (\rho({U_n}^2/2+e)+p)U_n+\rho U_n|u_\tau|^2/2\end{bmatrix}_{j+1/2}, \tag{2.16a}$$

$$F_c=\sum_{i=1,3}\xi_i\varphi_a f_i^{eq,L}+\sum_{i=2,4}\xi_i\varphi_a f_i^{eq,R}, \tag{2.16b}$$

where the flow variables with subscript "$j+1/2$" are determined by $\mathbf{W}_{j+1/2}$. In addition, the viscous flux at the cell interface is calculated by the central difference method. The details can be referred to [28,46].

In Eq. (2.10), an undetermined variable is the dimensionless collision time τ_0. The influence of the value of τ_0 has been discussed by Yang et al. in [28]. For the hypersonic simulation here, τ_0 can be treated as a switch function defined by:

$$\tau_0=\max\left\{\tau^L,\tau^R\right\}, \tag{2.17a}$$

$$\tau^L=\max_{j=1,N_{fL}}\left\{\tau_j\right\},\quad \tau^R=\max_{j=1,N_{fR}}\left\{\tau_j\right\}, \tag{2.17b}$$

$$\tau_j=\tanh\left(C\frac{|p^L-p^R|}{p^L+p^R}\right), \tag{2.17c}$$

where C is the amplification factor, p^L and p^R are the pressure at left and right sides of the cell interface. N_{fL} and N_{fR} are the number of faces of control volume on left and right sides of the cell interface respectively.

2.3 Implicit block LU-SGS algorithm for temporal discretization

The simulation of hypersonic flow field is always time consuming due to slow convergent rate, especially for some complex geometries. Complex conditions such as strong shock waves, boundary layer interactions and large viscosity gradient may occur in such problems, which will lead to computation unstable or even divergent. In addition, the huge quantities of mesh points used in the simulation of hypersonic flows request a much stricter convergence condition. Therefore, an implicit block LU-SGS algorithm is introduced to the above mentioned LBFS to improve its convergence and stability in this work.

An implicit scheme-based LBFS for steady flows can be written as

$$\Omega\frac{\Delta W^n}{\Delta t}+\oint_\Gamma (F_c^{n+1}-F_v^{n+1})dS=0, \tag{2.18}$$

where $\Delta W^n=W^{n+1}-W^n$ and the superscript $''n''$ denotes the n-th time step. For simplicity, we introduce the following definition of the residual

$$R(W)=\oint_\Gamma (F_c-F_v)dS. \tag{2.19}$$

Since the viscous and inviscid fluxes at the $(n+1)$-th time step are unknown, Eq. (2.18) cannot be solved directly. However, the residual at $(n+1)$-th time step can be linearized to that at n-th time step as

$$R(W^{n+1})=R(W^n)+\frac{\partial R(W^n)}{\partial W}\Delta W^n=R(W^n)+A\Delta W^n, \tag{2.20}$$

where A is the flux Jacobian. By substituting Eq. (2.20) into Eq. (2.18) and according to the LU-SGS scheme, we have

$$(L+D+U)\Delta W^n=-R(W^n). \tag{2.21}$$

Here, L, D and U are strictly the lower, diagonal and upper matrices. For structured grid, these matrices have the expressions of

$$\begin{cases} L=-(A_{i-1}^{+}+A_{j-1}^{+}),\\ D=\dfrac{\Omega}{\Delta t}\bar{\bar{I}}+A_i^{+}-A_i^{-}+A_j^{+}-A_j^{-},\\ U=A_{i+1}^{-}+A_{j+1}^{-}, \end{cases} \tag{2.22}$$

where

$$A^{\pm}=\frac{1}{2}(A\pm\omega\sigma_A\bar{\bar{I}}). \tag{2.23}$$

ω is the overrelaxation parameter, which would increase stability but may result in slow convergence with a higher value. In this work, ω is taken as 1. σ_A is the spectral radius of the flux Jacobian. Furthermore, to construct a matrix-free method, Eq. (2.21) can be approximated by

$$(D+L)D^{-1}(D+U)\Delta W^n=-R(W^n). \tag{2.24}$$

Note that, in Eq. (2.24), the residual at n-th time step $R(W^n)$ is computed by the LBFS as illustrated in Subsection 2.2.

Different from the conventional LU-SGS scheme, in the implicit block LU-SGS scheme, the forward and backward sweep processes are executed more than once in each time step to well approximate the original block diagonal matrix. Specifically, the sweep process of the implicit block LU-SGS scheme can be written as

$$\begin{cases} D\Delta W^* = -R(W^n) - L\Delta W^* - U\Delta W^{n,k-1}, & \text{forward sweep,} \\ D\Delta W^{n,k} = -R(W^n) - L\Delta W^* - U\Delta W^{n,k}, & \text{backward sweep.} \end{cases} \tag{2.25}$$

where k is the iteration number in a single calculation loop, which is generally called the inner iteration. This whole process is to replace the unknowns in the upper diagonal part by the values obtained in the last iteration step in the forward sweep, and the unknowns in lower diagonal part of the backward sweep are replaced by the values calculated in the forward step. Once the inner iteration is completed, the macroscopic variables can be updated by $W^{n+1} = W^n + \Delta W^n$. The full implementation of the implicit block LU-SGS algorithm-based LBFS is shown in Algorithm 1.

3 Numerical examples

The biconics model is firstly used to validate the present scheme and a hypersonic flight vehicle outline with wings and rudders is then calculated to extend its applications in complex hypersonic problems. In this work, all the computations were done on a personal computer (PC) with a processor of 3.3GHz.

3.1 Case 1: Biconics model

Experimental data in the wind tunnel test of biconics model is used to validate the present method. The total length of this test case is 122.24mm, and the detailed geometric information is shown in reference [47]. The free stream Mach number is Ma=9.86, the pressure is $P = 59.92$Pa and the temperature is $T = 48.88$K. To capture shock waves and describe boundary layer accurately in hypersonic flows, structured, as shown in Fig. 4, is applied for numerical simulation.

To demonstrate the computational efficiency of the proposed scheme, we depict the convergence histories of residuals with different values of $k = 1,2,3,5,12$ in Fig. 5. Fig. 6 shows the CPU time of different k correspondingly. The explicit scheme cannot offer a convergent result and it is not shown in the figure.

It can be seen from Fig. 5 that the convergent speed is enhanced obviously when the inner iteration is applied in a single loop. As the value of k increases, the convergent speed is improved gradually. For CPU time cost as shown in Fig. 6, $k = 3$ could have 25% of time-saving and the iteration steps could decrease by a half. Therefore, $k = 3$ is recommended in this case to balance both convergent speed and computational efficiency.

Fig. 7 and Fig. 8 show the pressure and Mach number distributions obtained by present solver with inner iteration number of $k = 3$. These pictures show that the present solver can well capture strong shock waves and describe flow details such as low pressure after the biconics tail. Fig. 9 shows the comparison of pressure coefficient computed

Algorithm 1 Implicit block LU-SGS.

Input: W^n

Output: W^{n+1}

```
$R(W) \leftarrow \oint_\Gamma (F_c - F_v) dS$ /$*$ $n_x$, $n_y$: the number of grid points on $x$, $y$-direction $*$/
for $k \leftarrow 1$ to $n_k$ do
    for $const \leftarrow 2$ to $n_x + n_y - 2$ do                                 /* Forward sweep */
        for $j \leftarrow 1$ to $n_y - 1$ do
            $i \leftarrow const - j$
            $D_{ij} \leftarrow \frac{\Omega}{\Delta t}\bar{\bar{I}} + A_i^+ - A_i^- + A_j^+ - A_j^-$
            if  $k \leftarrow 1$ then
                $\Delta W_{ij}^{n,0} \leftarrow 0$
            end
            $\Delta W_{ij}^* \leftarrow D_{ij}^{-1}\left[-R_{ij}(W^n) - L_{ij}\Delta W_{ij}^* - U_{ij}\Delta W_{ij}^{n,k-1}\right]$
        end
    end
for $const \leftarrow n_x + n_y - 2$ to 2 do                                     /* Backward sweep */
        For $j \leftarrow n_y - 1$ to 1 do
            $i \leftarrow const - j$
            $D_{ij} \leftarrow \frac{\Omega}{\Delta t}\bar{\bar{I}} + A_i^+ - A_i^- + A_j^+ - A_j^-$
            $\Delta W_{ij}^{n,k} \leftarrow D_{ij}^{-1}\left[-R_{ij}(W^n) - L_{ij}\Delta W_{ij}^* - U_{ij}\Delta W_{ij}^{n,k}\right]$
        end
    end
end
$W^{n+1} \leftarrow W^n + \Delta W^{n,k}$
```

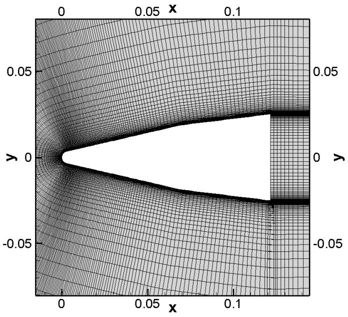

Figure 4: Partial view of structured grid around biconics model.

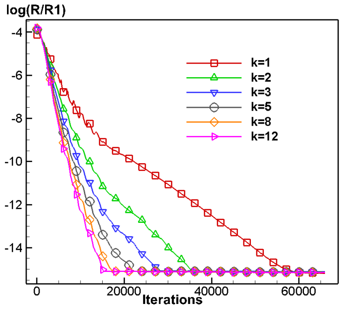

Figure 5: Comparison of convergence history by different number of inner iterations for biconics model.

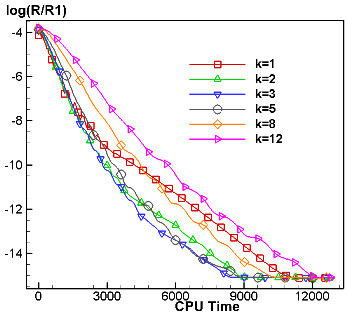

Figure 6: Comparison of CPU time cost by different number of inner iterations for biconics model.

by Roe scheme, van Leer scheme and the present solver. Fig. 10 depicts the comparison of heat flux between the above mentioned three schemes and the experimental data [47]. It can be observed that the results of present solver are in line with those of the classical schemes and the experimental data. Overall, the present solver shows good computational accuracy and better efficiency of reaching a convergent state for simulating hypersonic flows.

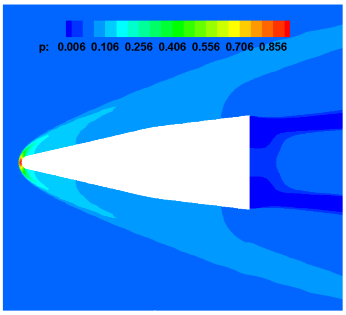

Figure 7: Pressure contours by the present solver for biconics model.

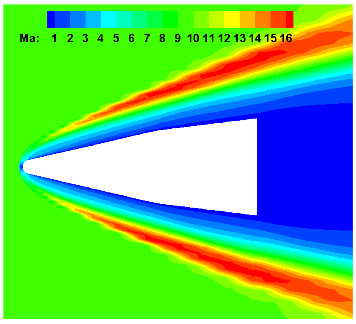

Figure 8: Mach number contours by the present solver for biconics model.

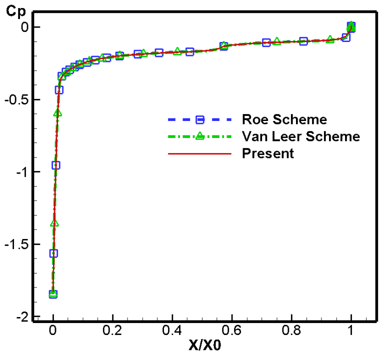

Figure 9: Comparison of pressure coefficient on the surface of biconics model by different schemes.

3.2 Case 2: Hypersonic flight vehicle

A hypersonic flight vehicle configuration with wings and rudders is used to demonstrate the potential of the developed method in industrial application. The geometry has 5665mm in total length and 1480mm in width (including rudder). The free stream with Mach number of $Ma=5.56$, pressure of $P=105$Pa and temperature of $T=218.6$K is

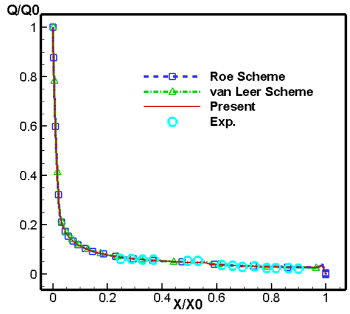

Figure 10: Comparison of heat flux on the surface of biconics model by different schemes.

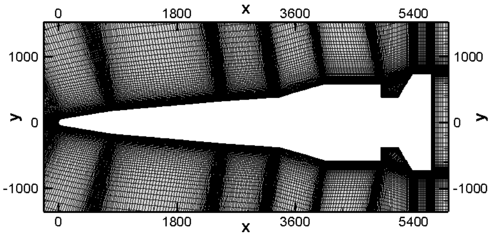

Figure 11: Partial view of structured grid around the hypersonic flight vehicle.

considered in this case. The structured grid is used, which is shown in Fig. 11.

Firstly, we present the comparison of convergence history and CPU time cost between different values of k in Figs. 12 and 13. It can be seen from Fig. 12 that increasing inner iterations in a single loop can speed up the convergent rate. As for CPU time shown in Fig. 13, $k=3$ could save 34% of the computational time as compared to $k=1$ and it will be used in this case.

Fig. 14 shows the pressure contours around the hypersonic flight vehicle. It can be seen clearly that the largest pressure occurs near head area, which matches the situation in Fig. 16. The leading edge on rudder has larger pressure than that on wing due to a larger angle, which would result in fierce gas squeezing. Fig. 15 shows the Mach number distribution. Together with Fig. 17, we can see that the largest value of Q/Q_0 is not exactly on the leading edge, but near it. The reason is that the turbulent flows would result in a larger heat flux after the stagnation point. Similarly, the larger angle of rudder results in larger heat flux than wing. Comparison of Fig. 16 and Fig. 17 shows that the results of the present solver have the same tendency as those of the classical schemes. However, the comparison of heat flux in Fig. 17 shows larger difference between the compared schemes than that of pressure coefficient in Fig. 16. It can be seen that near the area of shock waves such as head and rudders, the present solver according with the Roe scheme better, which has a high resolution of shock waves [48]. Therefore, it can be concluded that the LBFS could catch and describe more details, such as the strong

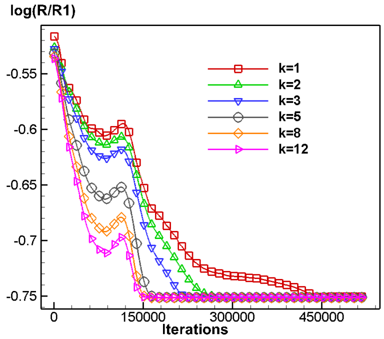

Figure 12: Comparison of convergence history by different number of inner iterations for the flight vehicle.

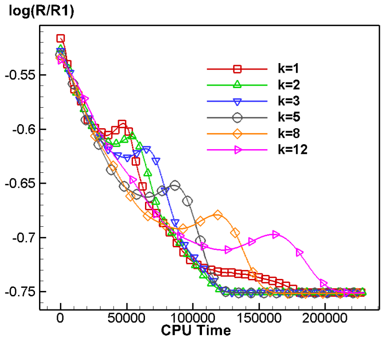

Figure 13: Comparison CPU time cost by different number of inner iterations for the flight vehicle.

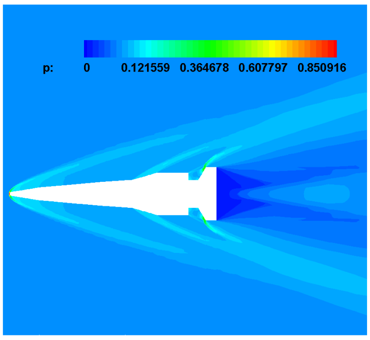

Figure 14: Pressure contours by the present solver for the flight vehicle.

shock waves and serious aeroheating produced by air drag and friction. From this test example, it is well demonstrated that the present solver has a great potential for flight vehicle design and industrial applications.

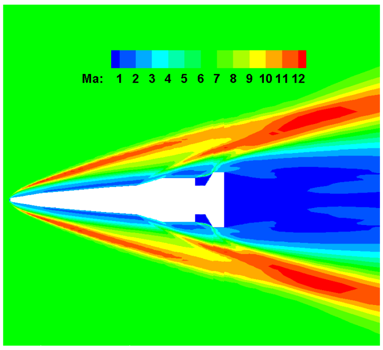

Figure 15: Mach number contours by the present solver for the flight vehicle.

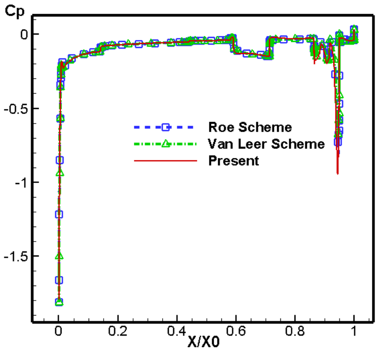

Figure 16: Comparison of pressure coefficient on the surface of the flight vehicle by different schemes.

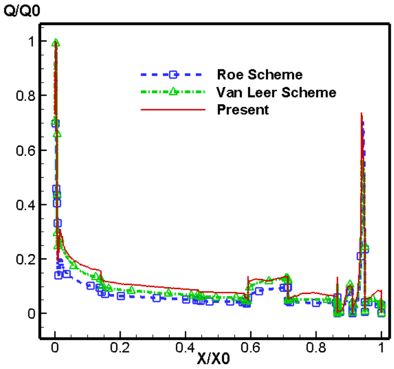

Figure 17: Comparison of heat flux on the surface of the flight vehicle by different schemes.

4 Conclusions

This paper presents a stable and efficient LBFS scheme which introduces the implicit block LU-SGS algorithm for simulating 2-D complex hypersonic flows. The Navier-

Stokes equations are discretized by FVM in spatial dimension and resultant governing equations are solved by implicit block LU-SGS algorithm. During the FVM discretization, the inviscid flux is evaluated by the LBFS with non-free parameter D1Q4 model, and a switch function is applied to control numerical dissipation. Meanwhile, the viscous flux is computed by conventional central difference method. During the iteration, the original LU-SGS computation is replaced by a k times repetitional looping, which is named as an implicit block approach. This repetition will approximately consider the original block diagonal matrix by introducing the lower and upper elements from the previous computation which obviously enhances the convergent property with little extra expense in memory request.

A biconics model at $Ma=9.86$ is firstly studied to validate the present solver. Then a hypersonic flight vehicle with wings and rudders is used to prove the potential of the present method in solving practical engineering problem. The numerical results show that the developed solver not only inherits the ability of describing the flow field with high accuracy from LBFS, but also combines the properties of robustness and good efficiency of reaching convergent from implicit block LU-SGS method. Therefore, it can be concluded that the present method is able to solve complex hypersonic problems and gets convergent results with both high accuracy and efficiency. This provides a stepping stone towards the future industrial application of LBFS method.

References

[1] J. Blazek, Computation Fluid Dynamics Principle and Application, Elsevier, 2001.

[2] X. L. Li, D. X. Fu and Y. W. Ma, *Optimized group velocity control scheme and DNS of decaying compressible turbulence of relative high turbulent Mach number,* Int. J. Numer. Meth. Fluids, 48 (2005), pp. 835–852.

[3] X. L. Li, D. X. Fu and Y. W. Ma, *Direct numerical simulation of hypersonic boundary layer transition over a blunt cone with a small angle of attack,* Phys. Fluids, 22 (2010), pp. 1–18.

[4] S. K. Godunov, *A difference method for numerical calculation of discontinuous solutions of the equations of hydrodynamics,* Matematicheskii Sbornik, 47 (1959), pp. 271–306.

[5] P. L. Roe, *Approximate Riemann solvers, parameter vectors, and difference schemes,* J. Comput. Phys., 43 (1981), pp. 357–372.

[6] J. Steger and R. Warming, *Flux vector splitting of the inviscid gasdynamic equations with applications to finite-difference methods,* J. Comput. Phys., 40 (1981), pp. 263–293.

[7] B. Van Leer, *Flux vector splitting for the Euler equations,* Lecture Notes Phys., 170 (1982), pp. 507–512.

[8] M. S. Liou and C. J. Steffen, *A new flux splitting scheme,* J. Comput. Phys., 107 (1993), pp. 23–39.

[9] C. W. Shu, *High order weighted essentially non-oscillatory schemes for convection dominated problems,* SIAM Rev., 51 (2009), pp. 82–126.

[10] K. Kitamura, E. Shima and P. L. Roe, *Evaluation of Euler fluxes for hypersonic heating computations,* AIAA J., 48 (2010), pp. 763–776.

[11] D. Chae, C. Kim and O. H. Rho, *Development of an improved gas-kinetic BGK scheme for inviscid and viscous flows,* J. Comput. Phys., 158 (2000), pp. 1–27.

[12] J. Jiang and Y. H. Qian, *Implicit gas-kinetic BGK scheme with multigrid for 3D stationary transonic high-Reynolds number flows,* Comput. Fluids, 66 (2012), pp. 21–28

[13] L. M. Yang, C. Shu, J. Wu, Y. Wang and Y. Sun, *Comparative study of 1D, 2D and 3D simplified gas kinetic schemes for simulation of inviscid compressible flows,* Appl. Math. Model., 43 (2017), pp. 85–109.

[14] K. Xu, *Gas-kinetic schemes for unsteady compressible flow simulations,* VKI for Fluid Dynamics Lecture Series, 1998-03 (1998).

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

[16] K. Xu, M. L. Mao and L. Tang, *A multidimensional gas-kinetic BGK scheme for hypersonic viscous flow,* J. Comput. Phys., 203 (2005), pp. 405–421.

[17] C. T. Tian, K. Xu, K. L. Chan and L. C. Deng, *A three-dimensional multidimensional gas-kinetic scheme for the Navier-Stokes equations under gravitational fields,* J. Comput. Phys., 226 (2007), pp. 2003–2027.

[18] L. M. Yang, C. Shu, J. Wu, N. Zhao and Z. L. Lu, *Circular function-based gas-kinetic scheme for simulation of inviscid compressible flows,* J. Comput. Phys., 255 (2013), pp. 540–557.

[19] L. M. Yang, C. Shu and J. Wu, *A simple distribution function-based gas-kinetic scheme for simulation of viscous incompressible and compressible flows,* J. Comput. Phys., 274 (2014), pp. 611–632.

[20] L. M. Yang, C. Shu, W. M. Yang and J. Wu, *Development of an efficient gas kinetic scheme for simulation of two-dimensional incompressible thermal flows,* Phys. Rev. E, 97 (2018), 013305.

[21] L. M. Yang, C. Shu and J. Wu, *A three-dimensional explicit sphere function-based gas-kinetic flux solver for simulation of inviscid compressible flows,* J. Comput. Phys., 295 (2015), pp. 322–339.

[22] L. M. Yang, C. Shu, Y. Wang and Y. Sun, *Development of discrete gas kinetic scheme for simulation of 3D viscous incompressible and compressible flows,* J. Comput. Phys., 319 (2016), pp. 129–144.

[23] L. M. Yang, C. Shu, W. M. Yang and J. Wu, *A simple gas kinetic scheme for simulation of 3D incompressible thermal flows,* Numer. Heat Tr. B-Fund, 72 (2017) pp. 450–468.

[24] C. Z. Ji, C. Shu and N. Zhao, *A lattice Boltzmann method-based flux solver and its application to solve shock tube problem,* Mod. Phys. Lett. B., 23 (2009), pp. 313–316.

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

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

[27] C. Shu, Y. Wang, L. M. Yang and J. Wu, *Lattice Boltzmann flux solver: an efficient approach for numerical simulation of fluid flows,* Trans. Nanjing Univ. Aeronaut. Astronaut., 31 (2014), pp. 1–15.

[28] L. M. Yang, C. Shu and J. Wu, *A hybrid lattice Boltzmann flux solver for simulation of viscous compressible flows,* Adv. Appl. Math. Mech., 8 (2016), pp. 887–910.

[29] L. M. Yang, C. Shu and J. Wu, *Extension of lattice Boltzmann flux solver for simulation of 3D viscous compressible flows,* Comput. Math. Appl., 71 (2016), pp. 2069–2081.

[30] D. Zhou, Z. L. Lu and T. Q. Guo, *A rotating reference frame-based lattice Boltzmann flux solver for simulation of turbomachinery flows,* Int. J. Numer. Meth. Fl., 83 (2017), pp. 561–582.

[31] L. M. Yang, C. Shu, W. M. Yang and Y. Wanf, *Extension of lattice Boltzmann flux solver*

for simulation of compressible multi-component flows, Mod. Phys. Lett. B, 32 (2018), 1840001.

[32] C. B. LEE, AND S. WANG, *Study of the shock motion in a hypersonic shock system/turbulent boundary layer interaction,* Exp. Fluids, 19 (1995), pp. 143–149.

[33] C. H. ZHANG, Q. TANG AND C. B. LEE, *Hypersonic boundary-layer transition on a flared cone,* Acta Mech. Sin., 29 (2013), pp. 48–54.

[34] J. TÖLKE, M. KRAFCZYK, M. SCHULZ, E. RANK AND R. BERRIOS, *Implicit discretization and nonuniform mesh refinement approach for FD discretization of LBGK models,* Int. J. Mod. Phys. C., 9 (1998), pp. 1143–1157.

[35] T. LEE AND C. L. LIN, *An Eulerian description of the streaming process in the lattice Boltzmann equation,* J. Comput. Phys., 185 (2003), pp. 445–471.

[36] J. HUANG, C. YANG AND X. C. CAI, *A fully implicit method for lattice Boltzmann equations,* SIAM J. Sci. Comput., 37 (2015), pp. 291–313.

[37] R. F. CHEN AND Z. J. WANG, *Fast, block lower-upper symmetric gauss-seidel scheme for arbitrary grids,* AIAA J., 38 (2000), pp. 2238–2245.

[38] W. D. LI AND L. S. LUO, *An implicit block LU-SGS finite-volume lattice-Boltzmann Scheme for steady flows of arbitrary unstructured meshes,* J. Comput. Phys., 327 (2016), pp. 503–518.

[39] P. R. SPALART AND S. A. ALLMARAS, *A one-equation turbulence model for aerodynamic flows,* AIAA Paper, 92-0439, 1992.

[40] T. KATAOKA AND M. TSUTAHARA, *Lattice Boltzmann method for the compressible Navier-Stokes equations with flexible specific-heat ratio,* Phys. Rev. E., 69 (2004), 035701.

[41] K. QU, C. SHU AND Y. T. CHEW, *Alternative method to construct equilibrium distribution functions in lattice-Boltzmann method simulation of inviscid compressible flows at high Mach number,* Phys. Rev. E., 75 (2007), 036706.

[42] K. QU, C. SHU AND Y. T. CHEW, *Simulation of shock-wave propagation with finite volume lattice Boltzmann method,* Int. J. Mod. Phys. C., 18 (2007), pp. 447–454.

[43] Q. LI, Y. L. HE, Y. WANG AND G. H. TANG, *Three-dimensional non-free-parameter lattice-Boltzmann model and its application to inviscid compressible flows,* Phys. Lett. A., 373 (2009), pp. 2101–2108.

[44] Z. L. GUO AND C. SHU, Lattice Boltzmann Method and Its Applications in Engineering, World Scientific Publishing, 2013.

[45] K. XU AND X. Y. HE, *Lattice Boltzmann method and gas-kinetic BGK scheme in the low-Mach number viscous flow simulations,* J. Comput. Phys., 190 (2003), pp. 100–117.

[46] R. C. SWANSON AND R. RADESPIEL, *Cell centered and cell vertex multigrid schemes for the Navier-Stokes equations,* AIAA J., 29 (1991), pp. 697–703.

[47] C. G. MILLER, *Experimental and predicted heating distributions for biconics at incidence in air at Mach 10,* NASA-TP-2334, 1984.

[48] W. L. WANG, H. LI AND S. PAN, *Performance comparison and analysis of different schemes and limiters,* Int. J. Phys. Math. Sci., 5 (2011), pp. 974–979.