Initial and boundary conditions for the lattice Boltzmann method

P. A. Skordos
Center for Nonlinear Studies, Los Alamos National Lab, B258, Los Alamos, New Mexico 87545
and Massachusetts Institute of Technology, 545 Technology Square, NE43-432, Cambridge, Massachusetts 02139
(Received 3 December 1992; revised manuscript received 27 August 1993)

An alternative approach of implementing initial and boundary conditions for the lattice Boltzmann method is presented. The basic idea is to calculate the lattice Boltzmann populations at a boundary node from the fluid variables that are specified at this node using the gradients of the fluid velocity. The numerical performance of the lattice Boltzmann method is tested on several problems with exact solutions and is also compared to an explicit finite-difference projection method. The discretization error of the lattice Boltzmann method decreases quadratically with finer resolution both in space and in time. The roundoff error of the lattice Boltzmann method creates problems unless double-precision arithmetic is used.

PACS number(s): 02.70.−c, 05.20.Dd, 47.15.−x, 51.90.+r

I. INTRODUCTION

The lattice Boltzmann method is a numerical scheme for simulating viscous fluids that is motivated by kinetic theory [1–7]. A short description of the method is as follows. The method represents the state of the fluid at a computational node using *a set of real numbers* which are called *populations* and are analogous to the microscopic density function of the Boltzmann equation. The populations are convected from one lattice site to another in discrete time steps, and are relaxed towards local equilibrium between every convection. The relaxation step or collision operator conserves mass and momentum (and energy for thermal models) just like a particle collision in kinetic theory.

The mapping from the populations F_i to the fluid variables ρ, V_x, V_y is a straightforward summation [Eq. (3) of Sec. II] and is performed in every step of the computation. By contrast, the inverse mapping from the fluid variables ρ, V_x, V_y to the populations F_i is somewhat tricky to compute, and has not received much attention for computational purposes. However, it is easy to see that if a good algorithm for calculating the inverse mapping were known, it could be used to implement initial and boundary conditions for the lattice Boltzmann method. This paper presents a method for calculating the inverse mapping that is accurate and easy to use, and demonstrates that this method provides accurate initial and boundary conditions.

In theory the inverse mapping can be obtained from the Chapman-Enskog expansion, but in practice additional manipulations are needed to employ the truncated Chapman-Enskog expansion successfully. These manipulations and the resulting equations are discussed in Sec. IV B. Aside from the Chapman-Enskog expansion, there is another way of calculating the inverse mapping that uses an extended collision operator (Sec. IV A). This approach is very intuitive and very general, and it is how the author originally discovered the recommended method for calculating the inverse mapping. The two approaches however, the Chapman-Enskog expansion and the extended collision operator produce identical results in the special case that is most useful in practice and is referred to here as the hybrid method [8].

The numerical tests of this paper include both initial and boundary value problems of incompressible viscous fluid flow. The accurate implementation of boundary conditions (nonperiodic boundaries and/or internal obstacle boundaries) is very important for practical scientific and engineering computations. The accurate initialization of the fluid flow is mostly important for numerical testing, and also for time-dependent flows when it is desired to calculate how an initial velocity field (which is chosen arbitrarily or perhaps is obtained from wind-tunnel measurements) evolves in time immediately after start up. The initial value problem is not very important for studying long-term behavior and steady-state flows except for avoiding initial errors such as large density waves that may delay the approach to the correct solution.

Traditionally the use of an accurate inverse mapping for the lattice Boltzmann populations has been avoided both for initial value and for boundary value problems. In the case of initial conditions, when the fluid density and velocity ρ, V_x, V_y are specified at time zero and the goal is to calculate ρ, V_x, V_y at later times, the populations F_i can be initialized equal to the equilibrium values F_i^{eq} which are known in terms of ρ, V_x, V_y. The error that results from this approximation can be overcome by discarding the first few steps and measuring the parameters of the flow afterwards (recalibrating the solution). This is often done in the literature without further discussion. A problem with this approach however is that it ends up solving a slightly different problem than the original problem defined by the original ρ, V_x, V_y. By contrast, traditional methods such as finite differences do not need any recalibration. Thus, to put the lattice Boltzmann method on equal footing with other methods (for numerical testing in particular) it is desirable to have an accurate means of calculating the populations F_i from the ini-

tial values of ρ, V_x, V_y.

In the case of boundary conditions there are techniques that avoid the inverse mapping as in the case of initial conditions. In particular, for solid wall boundaries the velocity of the fluid can be forced to zero by imposing a no-slip bounce back of the populations F_i (see [9] for a discussion of the actual location of the wall as a function of the simulation parameters). In the case of boundary conditions with nonzero velocity, such as the driven cavity problem ([10], p. 199), the velocity at the boundary can be "controlled" by inserting momentum (forcing) in every step as is done in lattice-gas automata. This type of forcing is somewhat *ad hoc* however, and is often inaccurate, and requires recalibration of the simulation parameters. In the case of an arbitrary velocity specification at the boundary, such as the fluid flows of Sec. VII, the forcing techniques and the recalibration become very difficult. Thus it is desirable to have an accurate means of calculating the populations F_i at a boundary node from the fluid variables ρ, V_x, V_y that are specified at this node.

In this paper we show how to calculate accurately the populations F_i at any node from the fluid variables ρ, V_x, V_y that are specified at this node. We have tested our method in the case of initialization and in the case of boundary conditions with arbitrary velocity and with zero velocity, and we have obtained good results in all cases. For exposition we use the hexagonal seven-speed model with the $1/\tau$ collision operator (Refs. [1–4]), but it is straightforward to apply our results to other lattice Boltzmann schemes. In the Appendix we apply our results to the orthogonal nine-speed model. We present computer simulations using both the hexagonal seven-speed model and the orthogonal nine-speed model, and we compare the two models in terms of numerical accuracy. In the next two sections we review the hexagonal model. Then we describe several ways of calculating the populations F_i from ρ, V_x, V_y, including our recommended method. In Sec. V we discuss the numerical roundoff error of the lattice Boltzmann method, and show that roundoff error in the equilibrium population formulas causes problems unless double-precision arithmetic is used. In Secs. VI and VII we discuss our simulations of initial-value problems and boundary-value problems. We compare the lattice Boltzmann method against an explicit finite-difference projection method, and we also demonstrate that the lattice Boltzmann method has second-order convergence both in space and in time.

II. HEXAGONAL SEVEN-SPEED $1/\tau$ MODEL

We consider a hexagonal lattice with six moving populations denoted by F_i, $i=1,\ldots,6$ and one rest-particle population denoted by F_0. We suppose that the fluid variables ρ, V_x, V_y are known at every node at time zero, and the goal is to calculate ρ, V_x, V_y at later times. At startup the populations F_i are initialized from the given fluid variables ρ, V_x, V_y (see Sec. IV). After initialization, successive steps of relaxation and convection are performed to calculate the fluid variables ρ, V_x, V_y at later times. The relaxation and convection steps are described by the following formulas:

$$\begin{aligned} F_i(\mathbf{x}+\mathbf{e}_i\Delta t, t+\Delta t) &= F_i(\mathbf{x},t)+(-1/\tau) \\ &\quad \times[F_i(\mathbf{x},t)-F_i^{\mathrm{eq}}(\mathbf{x},t)] , \\ F_0(\mathbf{x},t+\Delta t) &= F_0(\mathbf{x},t)+(-1/\tau)[F_0(\mathbf{x},t)-F_0^{\mathrm{eq}}(\mathbf{x},t)] , \\ & \qquad i=1,\ldots,6 , \qquad (1) \\ \tau &= \tfrac{1}{2}+\frac{4\Delta t\nu}{\Delta x^2} . \end{aligned}$$

The relaxation parameter τ is chosen to achieve the desired kinematic viscosity ν given the space and time discretization parameters Δx, Δt. The vector $\mathbf{e}_i$ stands for the six velocity directions of the hexagonal lattice

$$\mathbf{e}_i=\frac{\Delta x}{\Delta t}\left[\cos\frac{2\pi(i-1)}{6},\ \sin\frac{2\pi(i-1)}{6}\right] . \qquad (2)$$

The velocity $\mathbf{V}(\mathbf{x},t)$ and density $\rho(\mathbf{x},t)$ are computed from the populations $F_i(\mathbf{x},t)$ using the relations

$$\begin{aligned} \rho(\mathbf{x},t) &= \sum_{i=0}^{6} F_i(\mathbf{x},t) , \\ \rho(\mathbf{x},t)\mathbf{V}(\mathbf{x},t) &= \sum_{i=1}^{6} F_i(\mathbf{x},t)\mathbf{e}_i . \end{aligned} \qquad (3)$$

The variations of density around its mean value (spatial mean which is constant in time) provide an estimate of the fluid pressure $P(\mathbf{x},t)$, according to the following equation:

$$P(\mathbf{x},t)=c_s^2[\rho(\mathbf{x},t)-\langle\rho\rangle] . \qquad (4)$$

The speed of sound is

$$c_s=\sqrt{3w_0}(\Delta x/\Delta t) , \qquad (5)$$

where the coefficient w_0 is discussed below. The equilibrium populations $F_i^{\mathrm{eq}}(x,t)$ are given by the following equations:

$$\begin{aligned} F_i^{\mathrm{eq}}(\mathbf{x},t) &= \rho(\mathbf{x},t)[w_0+w_1(\mathbf{e}_i\cdot\mathbf{V})+w_{20}(\mathbf{e}_i\cdot\mathbf{V})(\mathbf{e}_i\cdot\mathbf{V}) \\ &\quad +w_{21}(\mathbf{V}\cdot\mathbf{V})] , \\ F_0^{\mathrm{eq}}(\mathbf{x},t) &= \rho(\mathbf{x},t)[z_0+z_{21}(\mathbf{V}\cdot\mathbf{V})] , \qquad (6) \\ 6w_0+z_0 &= 1 , \\ w_1 &= 1/(3c^2) , \quad w_{20}=2/(3c^4) , \quad w_{21}=-1/(6c^2) , \\ z_{21} &= -1/c^2 , \quad c=\Delta x/\Delta t . \end{aligned}$$

The above coefficients are chosen so that the Chapman-Enskog expansion (see Sec. III) of the evolution equation (1) matches the Navier-Stokes equations. In particular, the coefficient w_1 is determined from momentum conservation, the coefficient w_{20} is determined from Galilean invariance (i.e., the convection term $(V_x\partial V_x/\partial x+V_y\partial V_x/\partial y)$ must appear in the Chapman-Enskog expansion with a constant factor equal to one), the coefficient w_{21} is chosen to eliminate the $(\mathbf{V}\cdot\mathbf{V})$ dependence of the pressure, and the coefficient z_{21} is chosen to eliminate the $(\mathbf{V}\cdot\mathbf{V})$ term in the mass conservation equation. There is some freedom in choosing the remaining

coefficients w_0 and z_0, but they must satisfy $6w_0+z_0=1$ to conserve mass. In our simulations we use the balanced choice $w_0=z_0=\frac{1}{7}$ unless indicated otherwise.

The computational cycle of the lattice Boltzmann method is organized as follows: The current lattice populations $F_i(\mathbf{x},t)$ are used to calculate the velocity field $\mathbf{V}(\mathbf{x},t)$ and density field $\rho(\mathbf{x},t)$ according to Eq. (3). These fields are the numerical solution at time t, and they are also used to compute the equilibrium populations $F_i^{\rm eq}(\mathbf{x},t)$ which are needed to advance the solution. The equilibrium populations $F_i^{\rm eq}(\mathbf{x},t)$ are used to relax the $F_i(\mathbf{x},t)$ into "relaxed" populations which are then convected according to Eq. (1) to produce the lattice populations at the next time step. Then the cycle repeats. An implementation issue regarding roundoff error is discussed in Sec. V. Boundary conditions are discussed in Secs. IV and VII.

III. CHAPMAN-ENSKOG EXPANSION

The Chapman-Enskog expansion is outlined here. The goal of the Chapman-Enskog expansion is to derive a set of partial differential equations in terms of ρ and $\rho\mathbf{V}$ that approximate the behavior of the lattice Boltzmann fluid in the limit of Δx, Δt going to zero, with the ratio $\Delta x/\Delta t=c$ constant, and the ratio (V/c) small where V denotes the macroscopic speed of the fluid. In particular, we want to derive the mass continuity equation and the Navier-Stokes momentum equations. The first step is to Taylor expand the population variable $F_i(\mathbf{x}+\mathbf{e}_i\Delta t, t+\Delta t)$ in the evolution equation (1) around the point $(\mathbf{x},t)$. This produces an equation whose left-hand side is a Taylor series and whose right-hand side is equal to $(-1/\tau)(F_i-F_i^{\rm eq})$. This equation has the following form to first order in the derivatives:

$$\Delta t\left[\frac{\partial F_i}{\partial t}+\mathbf{e}_i\cdot\nabla F_i\right]+\cdots=(-1/\tau)(F_i-F_i^{\rm eq})\ . \tag{7}$$

The second step is to combine the Taylor series equation (7) with the mass and momentum conservation relations [Eq. (3)] to derive three equations corresponding to mass and momentum conservation. The left-hand side of these equations is a Taylor series, and the right-hand side is zero because the equilibrium populations $F_i^{\rm eq}$ are chosen to satisfy mass and momentum conservation (for example, $\sum_{i=0}^{6}F_i=\sum_{i=0}^{6}F_i^{\rm eq}$). The three Taylor series that are derived in this way contain partial derivatives of quantities that are sums and tensors of the populations F_i. These equations have the following form to first order:

$$\begin{aligned}&\partial\left[\sum_{i=0}^{6}F_i\right]\Big/\partial t+\nabla\cdot\left[\sum_{i=1}^{6}\mathbf{e}_iF_i\right]+\cdots=0\ ,\\ &\partial\left[\sum_{i=1}^{6}\mathbf{e}_iF_i\right]\Big/\partial t+\nabla\cdot\left[\sum_{i=1}^{6}\mathbf{e}_i\mathbf{e}_iF_i\right]+\cdots=0\ .\end{aligned} \tag{8}$$

If we truncate the mass equation to first-order terms in the derivatives, the resulting equation contains only sums of F_i and no tensors. The sums of F_i can be converted easily to ρ and $\rho\mathbf{V}$, and this produces the mass continuity equation. The momentum equation must be truncated to second-order terms in the derivatives to produce the Navier-Stokes equations. This is necessary because second-order spatial derivatives contribute to the viscosity of the fluid. Second-order terms are not shown in Eq. (8) but they are easy to write [11].

A complication arises with the pressure tensor $(\sum_i\mathbf{e}_i\mathbf{e}_iF_i)$ which appears in the momentum equation (8). The pressure tensor cannot be expressed in terms of ρ and $\rho\mathbf{V}$ without introducing an approximation of the F_i in terms of ρ and $\rho\mathbf{V}$. This approximation is necessary in the mass equation also if we include higher-order terms in the mass equation. The approximation of the populations F_i is the third step of the Chapman-Enskog expansion.

The Chapman-Enskog expansion approximates the populations $F_i(\mathbf{x},t)$ with the equilibrium populations $F_i^{\rm eq}(\mathbf{x},t)$ to zero order, and with the sum $F_i^{\rm eq}(\mathbf{x},t)+F_i^{(1)}(\mathbf{x},t)$ to first order. The correction term $F_i^{(1)}(\mathbf{x},t)$ is discussed below. The approximation of F_i can be viewed as another series expansion that is used in parallel with the Taylor series expansion. The accuracy of the F_i approximation improves as (V/c) becomes smaller. To retrieve the Navier-Stokes equations, it is sufficient to calculate up to the first-order approximation $F_i^{\rm eq}(\mathbf{x},t)+F_i^{(1)}(\mathbf{x},t)$. However, we must calculate up to second-order terms in the Taylor series, as stated previously, in order to retrieve all the viscosity terms.

The correction term $F_i^{(1)}$ is computed from $F_i^{\rm eq}$ using the evolution equation (1) Taylor expanded to first order with the F_i replaced by the zero order estimate $F_i^{\rm eq}$ as follows:

$$F_i^{(1)}=-\tau\Delta t\left[\frac{\partial F_i^{\rm eq}}{\partial t}+\mathbf{e}_i\cdot\nabla F_i^{\rm eq}\right]\ . \tag{9}$$

Now we can use the estimate $F_i^{\rm eq}(\mathbf{x},t)+F_i^{(1)}(\mathbf{x},t)$ for the populations F_i in the momentum Eq. (8), and we can also express $F_i^{\rm eq}$ in terms of ρ and $\rho\mathbf{V}$ to derive two partial differential equations in terms of ρ and $\rho\mathbf{V}$ corresponding to momentum conservation. By choosing the parameters of the equilibrium population formulas appropriately, we can make the momentum equations match the Navier-Stokes equations. For example, the parameters of Eq. (6) produce the following x-momentum equation (to second-order terms),

$$\begin{aligned}&\frac{\partial(\rho V_x)}{\partial t}+\frac{\partial(\rho V_xV_x)}{\partial x}+\frac{\partial(\rho V_xV_y)}{\partial y}\\ &\quad=-\frac{\partial(3c^2w_0\rho)}{\partial x}+\nu\nabla^2(\rho V_x)+\mu\frac{\partial(\nabla\cdot(\rho\mathbf{V}))}{\partial x}\ ,\\ &\nu=\frac{c^2\Delta t}{8}(2\tau-1)\ ,\quad \mu=2z_0\nu\ .\end{aligned} \tag{10}$$

Higher-order terms (which consist of higher than second-order derivatives of velocity and density) represent deviations from the Navier-Stokes equations, and they are discussed at some length in references [11,12]. This concludes our summary of the Chapman-Enskog procedure.

IV. CALCULATING THE F_i FROM ρ, V_x, V_y

We now address the problem of calculating the populations F_i at any node from given values of the fluid variables ρ, V_x, V_y at that node.

First we consider the initial value problem where the fluid variables ρ, V_x, V_y are specified on every node at time zero, and the goal is to calculate ρ, V_x, V_y at later times. If ρ is not specified according to compressible fluid flow equations—for example ρ is typically assumed to be constant in incompressible fluid flow—then ρ must be computed from the pressure as follows:

$$\rho=\langle\rho\rangle+\left[\frac{1}{c_s^2}\right]P\ , \tag{11}$$

where c_s is the speed of sound, $\langle\rho\rangle$ is the constant average density, and P is the pressure with the constant average pressure subtracted so that $\langle P\rangle=0$. The pressure can be calculated from the velocity by solving a Poisson equation if necessary (for example, see Ref. [10], p. 153). It is very important *not* to initialize the density to be constant [13]. The density must follow the pressure gradients of the lattice Boltzmann fluid according to Eq. (11); otherwise large density waves and error transients may result.

Once the fluid variables ρ, V_x, V_y are specified correctly, there are several ways of calculating the populations F_i using the fluid variables ρ, V_x, V_y. In this section we describe several methods and in Secs. VI and VII we compare these methods experimentally. We refer to each method by keyword names that are inspired by [2]. For example, the keyword d2q7 denotes 2 space dimensions and 7 populations per node. First we describe the hybrid method denoted by d2q7H, which is the recommended way for calculating the inverse mapping, and then we discuss the truncated Chapman-Enskog expansion which also produces the hybrid method after some manipulations.

A. Hybrid method and extended collision operator

The hybrid method is a lattice Boltzmann formulation that uses both the standard collision operator [Eqs. (1), (3), and (6)] and a new extended collision operator described here and denoted by d2q7X.

The idea behind the new collision operator is to include additional terms in the collision operator that are based on the gradients of the fluid velocity, so that the viscosity can be controlled independent of the relaxation parameter τ. The terms used here are motivated by equation 2.5.1 of Ref. [11].

The extended collision operator is used in the same way as the standard collision operator; namely the evolution equation and the conservation relations [Eqs. (1) and (3)] remain unchanged. The equilibrium population formulas of the extended collision operator, denoted $F_i^{*\mathrm{eq}}$, are as follows:

$$\begin{aligned}F_i^{*\mathrm{eq}}(\mathbf{x},t)=&\rho(\mathbf{x},t)[w_0+w_1(\mathbf{e}_i\cdot\mathbf{V})+w_{20}(\mathbf{e}_i\cdot\mathbf{V})(\mathbf{e}_i\cdot\mathbf{V})\\&+w_{21}(\mathbf{V}\cdot\mathbf{V})]\\&+w_{31}(\mathbf{e}_i\cdot\nabla(\mathbf{e}_i\cdot\rho\mathbf{V}))+w_{32}(\nabla\cdot\rho\mathbf{V})\ ,\\&i=1,\ldots,6\ ,\end{aligned} \tag{12}$$

$$F_0^{*\mathrm{eq}}(\mathbf{x},t)=\rho(\mathbf{x},)[z_0+z_{21}(\mathbf{V}\cdot\mathbf{V})]+z_{32}(\nabla\cdot\rho\mathbf{V})\ ,$$

$$3c^2w_{31}+6w_{32}+z_{32}=0\ .$$

The velocity gradients in the above equation (the terms with coefficients w_{31},w_{32},z_{32}) are computed using finite differences unless they are known by other means; for example some of the velocity gradients may be known at the boundary nodes (see Sec. VII). The coefficients $w_0,w_1,w_{20},w_{21},z_0,z_{21}$ have the same values as in the standard collision operator d2q7 [Eq. (6)]. It is worth noting that the velocity gradient terms of Eq. (12) can be viewed as a correction to the equilibrium population formulas as follows:

$$\begin{aligned}F_i^{*\mathrm{eq}}&=F_i^{\mathrm{eq}}+F_i^{(1X)}\ ,\\F_i^{(1X)}&=w_{31}(\mathbf{e}_i\cdot\nabla(\mathbf{e}_i\cdot\rho\mathbf{V}))+w_{32}(\nabla\cdot\rho\mathbf{V})\ .\end{aligned} \tag{13}$$

This view is useful in the next section to relate the extended collision operator to a truncated Chapman-Enskog expansion.

The shear and bulk viscosities of the extended collision operator are given by the following formulas (calculated using the Chapman-Enskog procedure):

$$\begin{aligned}\nu^*&=\frac{c^2\Delta t}{8}(2\tau^*-1)-\frac{3c^4w_{31}}{4}\ ,\\\mu^*&=\frac{c^2\Delta t}{4}(2\tau^*-1)z_0-\frac{3c^4w_{31}}{2}-3c^2w_{32}\ .\end{aligned} \tag{14}$$

We denote by τ^* the relaxation parameter of the extended collision operator to distinguish it from the relaxation parameter τ of the standard collision operator (and accordingly for other parameters). When τ^* is set equal to one, the coefficient w_{31} is chosen to achieve the desired shear viscosity given the discretization parameters $\Delta x,\Delta t$. The coefficient w_{32} is chosen to achieve the desired bulk viscosity, and the coefficient z_{32} is chosen to enforce the relation $(3c^2w_{31}+6w_{32}+z_{32})=0$ which corresponds to mass conservation.

In the case of the hybrid method (when the standard and extended collision operators are used in the same computation), the bulk viscosity of Eq. (14) is chosen equal to the bulk viscosity of the standard collision operator given by Eq. (10) (similarly for the shear viscosity). Also the relaxation parameter τ^* of the extended collision operator is set equal to 1.0. In this special case the coefficients w_{31},w_{32},z_{32} simplify as follows:

$$\begin{aligned}w_{31}&=\frac{(1-\tau)\Delta t}{3c^2}\ ,\\w_{32}&=-w_0(1-\tau)\Delta t\ ,\\z_{32}&=-z_0(1-\tau)\Delta t\ .\end{aligned} \tag{15}$$

The extended collision operator (d2q7X) controls the viscosity independent of the relaxation parameter τ^* so that τ^* can be set equal to one. In the special case of $\tau^*=1.0$ the extended collision operator replaces the F_i with $F_i^{*\mathrm{eq}}$ in each step, and thus the F_i can be set exactly equal to $F_i^{*\mathrm{eq}}$ at startup and at the boundary nodes.

The extended collision operator is accurate when implementing initial and boundary conditions, but it is not accurate when iterated many times. The reason is that the extended collision operator controls the viscosity using the gradients of the fluid velocity, and the gradients are computed using finite differences. The inexactness of finite differences produces an error in viscosity which means that the computed solution decays at a slightly different rate than desired. The error accumulates with successive iterations, and the method does not approximate the solution as Δt goes to zero (see Fig. 3 in Sec. VI B).

The best of both collision operators can be achieved, however (extended and standard), by combining the two operators. We observe that two different collision operators having the same transport coefficients (shear and bulk viscosity in our case) can be used interchangeably in the same computation. We have verified experimentally that switching between the different but equivalent collision operators incurs only a small error. By combining the two collision operators we get a hybrid method, denoted by d2q7H, where the extended collision operator is used at the boundary nodes all the time, and at every node during the first step of the computation. After the first step, the standard collision operator is used at the inner (nonboundary) nodes.

B. Truncated Chapman-Enskog expansion

An alternative way of deriving the hybrid method is to employ a truncated Chapman-Enskog expansion and to perform additional manipulations. We start by describing the zero order and first order expansions, and then we show how to modify and simplify the first-order expansion in order to derive the hybrid method.

The zero-order expansion, denoted by d2q7F0, approximates the populations F_i with the equilibrium value F_i^{eq}. As stated earlier this approximation is used very often in the literature, and it is accompanied by recalibration of the solution after the first few steps are discarded (initial transients). In our simulation tests of Sec. VI we do not perform any recalibration however, because our goal is to compare the accuracy of calculating the populations F_i from the fluid variables ρ, V_x, V_y.

The first-order expansion, denoted by d2q7F1, approximates the populations F_i with the Chapman-Enskog expansion truncated to first order,

$$F_i=F_i^{\mathrm{eq}}+F_i^{(1)} ,$$
$$F_i^{(1)}=-\tau\Delta t\left[\frac{\partial F_i^{\mathrm{eq}}}{\partial t}+\mathbf{e}_i\cdot\nabla F_i^{\mathrm{eq}}\right] . \tag{16}$$

By differentiating the equilibrium population formulas [Eq. (6)], we can get formulas for the derivatives of F_i^{eq} in terms of the derivatives of the fluid variables ρ, V_x, V_y. The derivatives of ρ, V_x, V_y are known in some cases (for example, in our exactly solvable fluid flow problems), but in general they must be estimated using finite differences. In our initialization tests of Sec. VI we use finite differences. In particular, the time derivatives of ρ, V_x, V_y are estimated using the Navier-Stokes momentum and continuity equations, and the spatial derivatives of ρ, V_x, V_y are estimated using spatial finite differences. We have also tested the initialization methods of this section using the exact values of the derivatives, and the results are qualitatively the same as those reported in Sec. VI.

In Sec. VI we will see that both d2q7F0 and d2q7F1 produce significant errors in initialization. It is a little surprising that the first-order Chapman-Enskog correction does not perform well, but there is an easy explanation. We observe that the correction term $F_i^{(1)}$ of Eq. (16) does not conserve momentum. This means that the velocity field that results from Eq. (16) is different from the original velocity field. The conservation relations that correspond to Eq. (16) are as follows:

$$\sum_i F_i^{(1)}=(\nabla\cdot\rho\mathbf{V})+\frac{\partial\rho}{\partial t} , \tag{17}$$

$$\sum_i F_i^{(1)}e_{ix}=\frac{\partial(\rho V_x)}{\partial t}+c_s^2\frac{\partial\rho}{\partial x}+\frac{\partial(\rho V_x V_x)}{\partial x}+\frac{\partial(\rho V_x V_y)}{\partial y} ,$$

and a similar equation for $\sum_i F_i^{(1)}e_{iy}$. Therefore mass is conserved via the macroscopic continuity equation, but momentum is not conserved. On the other hand, the above equations suggest an easy way to fix the problem: We simply add a viscosity Laplacian term so that momentum will be conserved via the Navier-Stokes momentum equation. The new (modified) Chapman-Enskog correction term, denoted by $F_i^{(1\mathrm{M})}$, is as follows [14]:

$$F_i^{(1\mathrm{M})}=-\tau\Delta t\left[\frac{\partial F_i^{\mathrm{eq}}}{\partial t}+\mathbf{e}_i\cdot\nabla F_i^{\mathrm{eq}}+[-\nu/(3c^2)]\nabla^2(\mathbf{e}_i\cdot\mathbf{V})\right] . \tag{18}$$

In our numerical tests of Sec. VI we refer to the above equation as d2q7F1M. Our numerical tests show that d2q7F1M is very accurate for initialization purposes. In practice, however, the d2q7F1M method is rather cumbersome to apply because it requires the calculation of many derivatives, including a time derivative and a Laplacian term.

Fortunately, Eq. (18) can be simplified greatly by neglecting second-order terms in the Mach number. This means that only terms up to first order in (V/c) are kept in the Chapman-Enskog expansion, and terms proportional to $(V/c)^2$ are discarded because they are small. In addition, the time derivatives are replaced by space derivatives using the macroscopic mass and momentum equations. Examples of this kind of expansion can be found in Refs. [15] and [16]. Thus, Eq. (18) simplifies to

$$F_i^{(1\mathrm{S})}=-\tau\Delta t\left[\frac{1}{3c^2}\mathbf{e}_i\cdot\nabla(\mathbf{e}_i\cdot\rho\mathbf{V})-w_0(\nabla\cdot\rho\mathbf{V})\right] \tag{19}$$

and similarly for the rest particle population

$$F_0^{(1S)} = -\tau\Delta t[-z_0(\nabla\cdot\rho\mathbf{V})] \ . \tag{20}$$

In Sec. VI we shall see that the simplified Eq. (19) is as accurate as the original Eq. (18) for initialization purposes. We also note that the above formulas look suspiciously similar to the hybrid method that was described in the last section. In fact, it is easy to verify that Eqs. (19) and (20) produce identical results with the hybrid method.

If Eq. (19) is used to initialize the populations F_i as $F_i = F_i^{\rm eq} + F_i^{(1S)}$, and the first relaxation step is performed, then the resulting populations which are convected (denoted as $\bar{F}_i$) are as follows:

$$\bar{F}_i = (F_i^{\rm eq} + F_i^{(1S)}) + (-1/\tau)F_i^{(1S)} \ , \tag{21}$$

$$\bar{F}_i = F_i^{\rm eq} + (1-\tau)\Delta t\left[\frac{1}{3c^2}(e_i\cdot\nabla(\mathbf{e}_i\cdot\rho\mathbf{V})) - w_0(\nabla\cdot\rho\mathbf{V})\right] \ . \tag{22}$$

The above populations are identical to the populations that are convected after a relaxation step using the extended collision operator [Eq. (13)] when the simplified values of w_{31}, w_{32}, z_{32} for the hybrid method are used [Eq. (15)]. In Secs. VI and VII we will test experimentally the methods discussed in this section.

V. NUMERICAL ROUNDOFF

This section discusses an implementation issue of the lattice Boltzmann method that can cause problems if one is unaware of it. If the lattice Boltzmann method is implemented exactly as described by Eqs. (1), (3), and (6), then the method suffers from roundoff error that grows as the ratio (V/c) becomes small; that is as $(\Delta x/\Delta t)$ becomes large. This is undesirable because large values of $(\Delta x/\Delta t)$ are useful to improve the accuracy of the lattice Boltzmann method.

The roundoff error (numerical loss of precision) arises in the computation of the equilibrium populations using Eq. (6). This formula is a sum of four terms. If we factor out the density $\rho(\mathbf{x},t)$, the first term is a constant coefficient w_0 and the remaining terms are proportional to V/c, $(V/c)^2$, and $V(V/c)^2$, respectively (see Table I). Consequently, when V/c is small, for example $V/c \simeq 10^{-3}$, the terms to be added have very disparate sizes and their sum suffers a significant loss of accuracy when the computer aligns the numbers to be added (about 5 or 6 decimal places when $V/c \simeq 10^{-3}$). If single precision arithmetic is used (about eight decimal places), then the loss of five digits is a serious problem.

In Sec. VI C, we will see that the computational error of the lattice Boltzmann method decreases at first when the speed $\Delta x/\Delta t$ increases, but after some point the error starts to increase with higher speeds. For example, in the Taylor vortex when the maximum fluid speed is 1.0, the error starts to increase at the rate of $(\Delta x/\Delta t)^{1.4}$ when the microscopic speed $(\Delta x/\Delta t)$ is larger than 300. Fortunately the error growth disappears when double-precision arithmetic is used, and this confirms that the breakdown of the method is caused by numerical roundoff. Furthermore we can estimate that each additional decimal place of computer arithmetic delays the roundoff problem in the equilibrium populations by a factor of 10 in the speed $\Delta x/\Delta t$. This means that double-precision arithmetic eliminates the roundoff problem for most practical purposes.

TABLE I. The terms of the equilibrium population formula have different sizes. When they are added together, numerical roundoff error can be significant.

Term	w_0	w_1	w_{20}	w_{21}
Size	1	V/c	$(V/c)^2$	$(V/c)^2$

We also note that apart from using double-precision arithmetic, there is an algebraic transformation that reduces the roundoff error in the equilibrium populations, and it can be used in all cases because it does not involve any additional cost. The algebraic transformation does not eliminate the roundoff error completely however; in general double-precision arithmetic remains necessary. The idea is to modify the populations F_i defined by Eqs. (1), (3), and (6) as follows:

$$\begin{aligned} F_i &= F_i - w_0\langle\rho\rangle \ , \\ \hat{F}_i^{\rm eq} &= F_i^{\rm eq} - w_0\langle\rho\rangle \ , \end{aligned} \tag{23}$$

where the spatial average density $\langle\rho\rangle$ is constant in time and typically equal to one. The non-moving population becomes $\hat{F}_0 = F_0 - z_0\langle\rho\rangle$. The conservation relations are modified accordingly,

$$\begin{aligned} \rho(\mathbf{x},t) &= \sum_{i=0}^{6}\hat{F}_i(\mathbf{x},t) + \langle\rho\rangle \ , \\ \rho(\mathbf{x},t)\mathbf{V}(\mathbf{x},t) &= \sum_{i=1}^{6}\hat{F}_i(\mathbf{x},t)\mathbf{e}_i \ . \end{aligned} \tag{24}$$

The new equilibrium population formulas are as follows:

$$\begin{aligned} \hat{F}_i^{\rm eq}(\mathbf{x},t) &= w_0[\rho(\mathbf{x},t) - \langle\rho\rangle] \\ &\quad + \rho(\mathbf{x},t)[w_1(\mathbf{e}_i\cdot\mathbf{V}) + w_{20}(\mathbf{e}_i\cdot\mathbf{V})(\mathbf{e}_i\cdot\mathbf{V}) \\ &\quad + w_{21}(\mathbf{V}\cdot\mathbf{V})] \ , \\ \hat{F}_0^{\rm eq}(\mathbf{x},t) &= z_0[\rho(\mathbf{x},t) - \langle\rho\rangle] + \rho(\mathbf{x},t)z_{21}(\mathbf{V}\cdot\mathbf{V}) \ . \end{aligned} \tag{25}$$

The new equilibrium population formulas are numerically better than the original ones because the term that used to be $w_0\rho$ is now $w_0(\rho - \langle\rho\rangle)$. The new quantity $(\rho - \langle\rho\rangle)$ is of the order $P/(3c^2w_0)$ and the pressure P is of the order ρV^2 as can be seen from the Navier-Stokes equations. Hence, the expression $w_0(\rho - \langle\rho\rangle)$ is of the order $\rho(V/c)^2$. The new formulas compute the same quantities as the original formulas, and they incur a smaller loss of precision. Loss of precision still occurs when the terms proportional to V/c and $(V/c)^2$ are combined.

VI. RESULTS—INITIAL VALUE

First we test initial value problems. For this purpose we use the analytic solutions of a decaying Taylor vortex

and a decaying shear flow in two dimensions with periodic boundary conditions. Figure 1 shows the velocity vector fields of these flows.

The decaying Taylor vortex [17] has the following analytic solution:

$$\begin{aligned} V_x(x,y,t) &= (-1/A)\cos(Ax)\sin(By)\exp(-2\alpha\nu t)\ , \\ V_y(x,y,t) &= (1/B)\sin(Ax)\cos(By)\exp(-2\alpha\nu t)\ , \\ P(x,y,t) &= -(1/4)[\cos(2Ax)/A^2 \\ &\quad +\cos(2By)/B^2]\exp(-4\alpha\nu t)\ , \end{aligned} \tag{26}$$

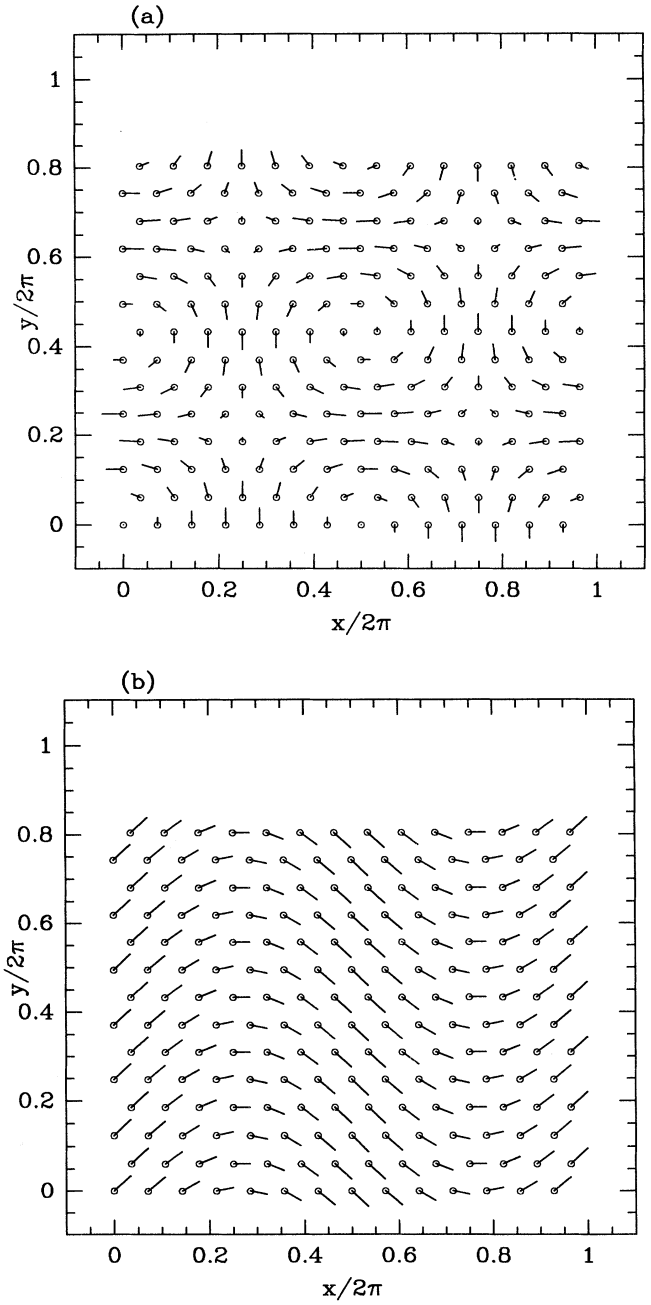

FIG. 1. The velocity field of the hexagonal Taylor vortex and the hexagonal shear flow are shown in (a) and (b), respectively. Both flows have periodic boundary conditions.

where the constant α is equal to $(A^2+B^2)/2$, and ν is the kinematic viscosity. The length constants A,B are chosen $A=1$ and $B=2/\sqrt{3}$ to produce the hexagonal Taylor vortex, and $A=B=1$ to produce the orthogonal Taylor vortex. The former is used to test the hexagonal seven-speed model, and the latter is used to test the orthogonal nine-speed model. The flow region of the hexagonal Taylor vortex is $0\le x\le 2\pi$ and $0\le y\le \pi\sqrt{3}$, and can be covered exactly by a hexagonal lattice using periodic boundary conditions. Similarly, the flow region of the orthogonal Taylor vortex is $0\le x\le 2\pi$ and $0\le y\le 2\pi$, and can be covered exactly by an orthogonal lattice using periodic boundary conditions.

The decaying shear flow has the following analytic solution:

$$\begin{aligned} V_x(x,y,t) &= A\ , \\ V_y(x,y,t) &= B\cos(kx-kAt)\exp(-k^2\nu t)\ , \\ P(x,y,t) &= \text{const}\ , \end{aligned} \tag{27}$$

where the constant k is chosen $k=1$ so that x varies between $0\le x\le 2\pi$, and the length constants A,B are chosen $A=B=1$ so that the horizontal velocity is equal to the maximum vertical velocity. The vertical extent of the shear flow is chosen $0\le y\le \pi\sqrt{3}$ for the hexagonal case, and $0\le y\le 2\pi$ for the orthogonal case in complete analogy with the Taylor vortex.

In all of the results reported below the coefficient of shear viscosity is chosen equal to one, $\nu=1$. The measured error V^E denotes the velocity relative error, and is calculated according to the following formula:

$$V^E=\frac{\sum_{x,y}|V_x-V_x^*|}{\sum_{x,y}|V_x^*|}+\frac{\sum_{x,y}|V_y-V_y^*|}{\sum_{x,y}|V_y^*|}\ , \tag{28}$$

where V^* denotes the exact analytic solution, and the sums are taken over the whole grid. In the case of the Hagen-Poiseuille flow and the oscillating plate problem (see Sec. VII) where $\sum_{x,y}|V_y^*|=0$, we use a different normalization as follows:

$$V^E=\frac{\sum_{x,y}|V_x-V_x^*|+\sum_{x,y}|V_y-V_y^*|}{\sum_{x,y}|V_x^*|}\ . \tag{29}$$

Double-precision arithmetic is used in all of the reported results unless stated otherwise (for example, in Fig. 4).

We define the Mach number M using the maximum fluid speed at time zero, which is equal to 1.0 for all of our test cases,

$$M=1/c_s=\Delta t/(\Delta x\sqrt{3w_0})\ . \tag{30}$$

We also define the pseudo-Mach-number or "computational Mach number" M_c,

$$M_c=1/c=\Delta t/\Delta x\ . \tag{31}$$

We use M_c rather than M in our figures because the discretization error of the lattice Boltzmann method depends on M_c rather than M as we will see below. In the

case of the Taylor vortex, which is a solution of the incompressible Navier-Stokes equations, the compressible effects are kept smaller than the discretization error by choosing $w_0=\frac{1}{7}$. Both the compressible effects and the discretization error decrease quadratically with M_c, and the choice $w_0=\frac{1}{7}$ keeps the compressible effects smaller than the discretization error in the Taylor vortex at least (see Sec. VI E). In the case of shear flow, which has zero density gradient and is a solution of the compressible Navier-Stokes equations, the error is independent of the Mach number M and it depends only on M_c.

For the hexagonal seven-speed model the choice $w_0=\frac{1}{7}$ produces a Mach number that satisfies the relation $M=(1.53M_c)=(1.53\Delta t/\Delta x)$. For the orthogonal nine-speed model described in the appendix the choice $y_0=w_0/4$ and $w_0=\frac{1}{7}$ produces $M=1.53M_c$ also. Another choice $w_0=10^{-6}/3$ is discussed briefly in Sec. VI D for the purpose of allowing high Mach numbers with small M_c in particular $M=10^3M_c$. We also note that different values of w_0 are used in Sec. VI E for the purpose of examining the error of the lattice Boltzmann method as a function of Δt while keeping the Mach number constant. In particular the Mach number is kept constant by varying w_0 in proportion to Δt^2 [see Eq. (32)]. This study allows us to distinguish between compressible effects and the discretization error of the lattice Boltzmann method.

A. Initialization error

This section compares the different methods of initialization that are described in Sec. IV and are denoted by d2q7F0, d2q7F1, d2q7F1M, and d2q7H. We recall that the simplified first-order Chapman-Enskog expansion [Eqs. (19) and (20)] is identical to the hybrid method d2q7H, and thus there is no need to test it separately. Figure 2 plots the error as it develops during the first 10 steps of the simulation. A 30×30 grid is used ($\Delta x=2\pi/30=0.2094$). Figure 2(a) plots the error in the case of the hexagonal Taylor vortex, using $\Delta t=0.001$ which gives $\tau=0.5912$ for the standard collision operator. The curves shown correspond to d2q7F0, d2q7F1, d2q7F1M, d2q7H (solid, dashed, dotted, dashed-dotted lines). Figure 2(b) plots the same data using $\Delta t=0.025$ which gives $\tau=2.780$ for the standard collision operator. We can see that the first-order momentum-conserving Chapman-Enskog expansion d2q7F1M and the hybrid method d2q7H produce very similar results, and they are the most accurate in all cases. We can also see that the first-order Chapman-Enskog expansion d2q7F1 that does not conserve momentum is more accurate than the zero-order expansion d2q7F0 when $\tau<1$ and inversely when $\tau>1$. Figures 2(c) and 2(d) plot the same data as Figs. 2(a) and 2(b) for the case of shear flow. The results are qualitatively the same. The experiments demonstrate that the hybrid method can be used to initialize accurately the populations F_i from the fluid variables ρ, V_x, V_y in an initial value problem.

B. Iterating the extended collision operator

This section examines the performance of the extended collision operator when iterated many times. We recall that the extended collision operator uses the gradients of the fluid velocity to control the viscosity. Figure 3 shows the error in simulating the hexagonal Taylor vortex and the hexagonal shear flow using a 30×30 grid. The error is plotted against M_c with Δt varying, and is calculated at the final time $T=1.0$ when the maximum velocity of the hexagonal Taylor vortex is approximately $\frac{1}{10}$th of its initial value. The curves correspond to the hybrid method d2q7H, and to the extended collision operator d2q7X using finite differences to calculate the gradients, and again to the extended collision operator d2q7X using the exact solution to calculate the gradients (solid, dashed, dotted lines). When the curves of Fig. 3 intersect at $M_c=0.026$, the relaxation parameter τ of the standard collision operator is equal to one, and the coefficients w_{31}, w_{32}, z_{32} of the extended collision operator vanish [see Eq. (15)]. At this point the extended collision operator is identical to the standard collision operator.

As M_c decreases below the value $M_c=0.026$, the error of the extended collision operator d2q7X using finite differences to calculate the gradients begins to grow and approaches relative error one as M_c goes to zero (dashed line). By contrast, the error of the extended collision operator d2q7X using the analytic solution to calculate the gradients decreases towards a minimum error (dotted line) which is determined by the spatial discretization error of the 30×30 grid. This shows that the use of finite differences creates problems after repeated iterations. As explained in Sec. IV the inexactness of finite differences produces an error in viscosity which accumulates and becomes large after repeated iterations.

The hybrid method d2q7H does not suffer from the problems of the extended collision operator after repeated iterations because the hybrid method uses the standard collision operator at the inner nodes after the first step (all nodes are inner in this experiment). Figure 3 shows that the hybrid method performs well in the case of periodic boundary conditions and remains accurate as M_c goes to zero (solid line). We note however that in the case of nonperiodic boundary conditions the hybrid method uses the extended collision operator at the boundary nodes in every step. In Sec. VII we will see that the use of finite differences at the boundary nodes does not cause any problems as M_c goes to zero, but it may cause instabilities when M_c is large.

C. Roundoff error

Figure 4 compares the error of the lattice Boltzmann method (d2q7H version) when single-precision arithmetic is used, when single-precision arithmetic together with the algebraic transformation of Sec. V is used, and when double-precision arithmetic is used (dotted, dashed, solid lines). The data come from simulations of the hexagonal Taylor vortex with periodic boundary conditions and 30×30 grid. The error is plotted against M_c with Δt varying and is calculated at the final time $T=1.0$. When

single-precision arithmetic is used and the speed $\Delta x/\Delta t$ exceeds 300 (therefore $M_c < 0.003$), there is a growth of error that is caused by numerical roundoff. The transformation of Sec. V together with single precision arithmetic can reduce the roundoff error but cannot prevent it. Double-precision arithmetic is necessary to prevent the error growth in the Taylor vortex for $M_c < 0.003$. As explained in Sec. V double-precision arithmetic eliminates the roundoff problem for most practical purposes.

D. Comparison with projection method

This section compares the error of the hybrid method d2q7H and the error of an explicit finite difference projec-

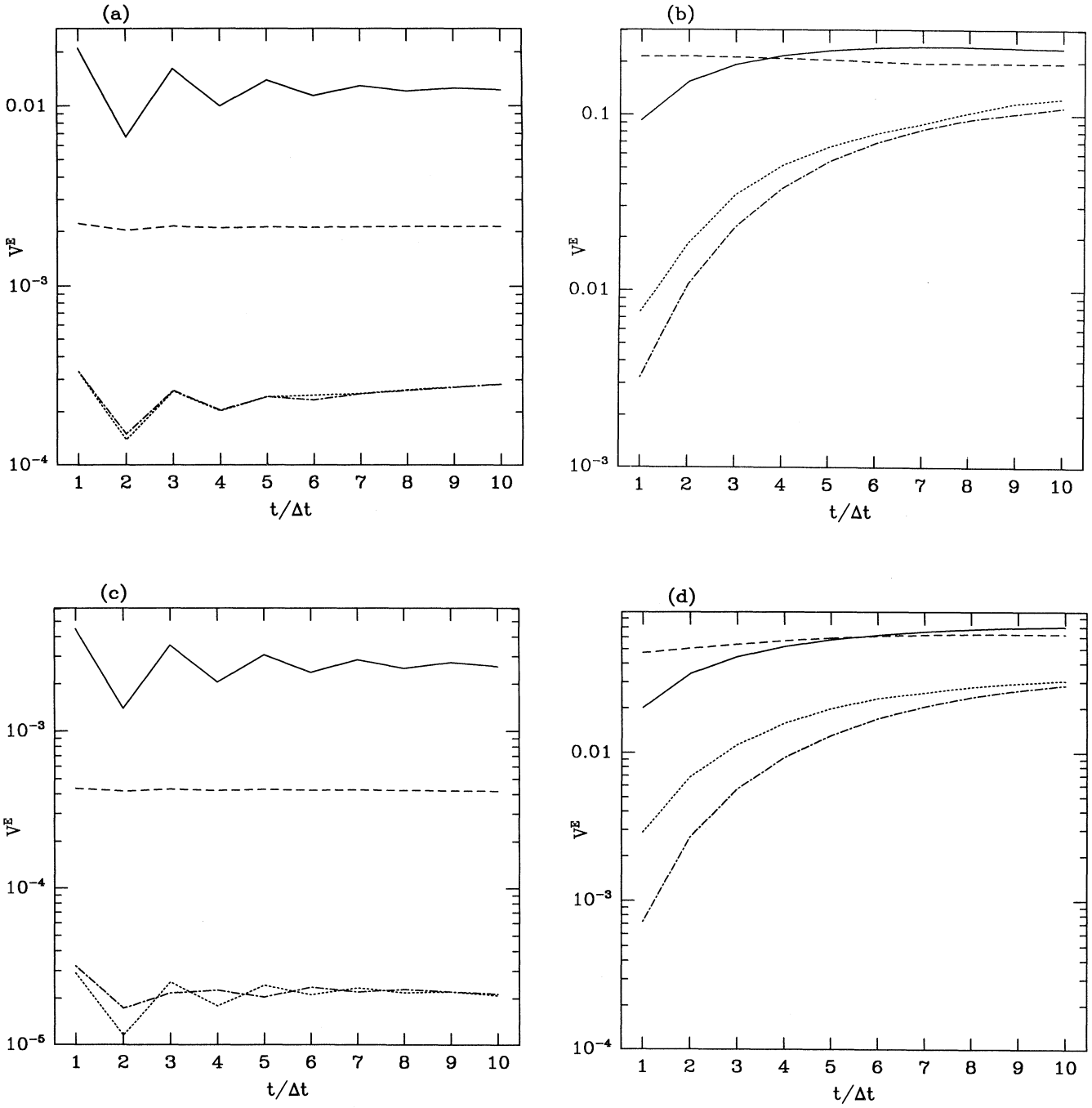

FIG. 2. The four initialization methods d2q7F0, d2q7F1, d2q7F1M, d2q7H (solid, dashed, dotted, dashed-dotted lines) are compared using a 30×30 grid and periodic boundary conditions. (a) and (b) plot the error in simulating the hexagonal Taylor vortex using $\Delta t = 0.001$ and 0.025, respectively ($\tau = 0.5912$ and $\tau = 2.780$). (c) and (d) plot the same data in the case of shear flow.

tion method in simulating the hexagonal Taylor vortex and the hexagonal shear flow with periodic boundary conditions. Both of these flows are defined in the hexagonal region $0 \leq x \leq 2\pi$ and $0 \leq y \leq \pi\sqrt{3}$, which means that the finite difference projection method must use the discretization $\Delta y = \Delta x \sqrt{3}/2$. Below we refer to the projection method with the symbol EP7 (EP stands for explicit projection) when it is applied to a hexagonal region, and with the symbol EP9 when it is applied to an orthogonal region (this is done in later sections).

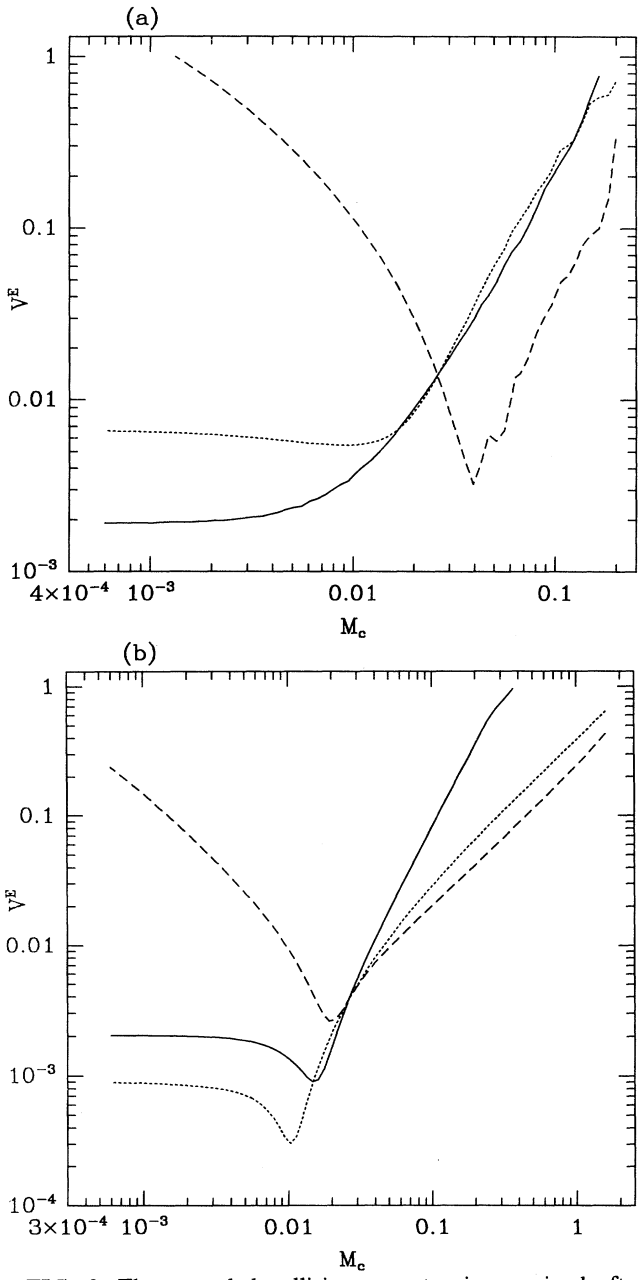

FIG. 3. The extended collision operator is examined after many iterations. The error is plotted against M_c with Δt varying, and is calculated at the final time $T = 1.0$. The curves correspond to the hybrid method d2q7H, to the extended collision operator d2q7X using finite differences to calculate the gradients, and again to the extended collision operator d2q7X using the known analytic solution to calculate the gradients (solid, dashed, dotted lines). (a) shows the error in simulating the hexagonal Taylor vortex and (b) shows the error in simulating the hexagonal shear flow.

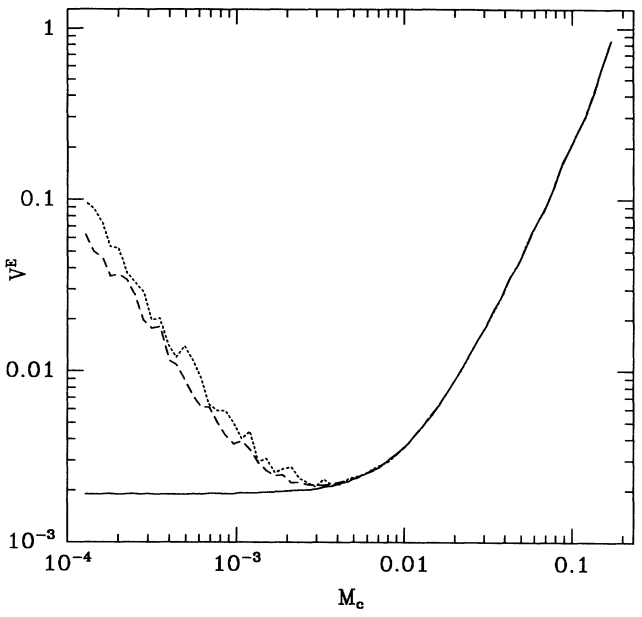

FIG. 4. The error of the lattice Boltzmann method d2q7H is shown when single-precision arithmetic is used, when single-precision arithmetic together with the algebraic transformation of Sec. V is used, and when double-precision arithmetic is used (dotted, dashed, solid lines).

The explicit finite difference projection method computes an estimate of the velocity by solving a discretized Navier-Stokes momentum equation where the pressure term is omitted ([10], p. 160). Then the velocity estimate is corrected in order to satisfy incompressibility by solving a Poisson equation. This correction takes into account the pressure effects that were omitted in the first estimate of the velocity. In addition the solution of the Poisson equation provides an estimate of the pressure at the current time step. In our simulations we use SOR (successive over-relaxation) ([18], p. 680) to solve the Poisson equation, forward Euler to estimate the time derivative, and three-point symmetric differences to calculate the spatial derivatives on a grid that is orthogonal and nonstaggered.

Figure 5(a) plots the error in simulating the hexagonal Taylor vortex against M_c with Δt varying. The error is calculated at the final time $T = 1.0$ when the maximum velocity of the hexagonal Taylor vortex is approximately $\frac{1}{10}$th of its initial value. The curves correspond to d2q7H using a 30×30 grid, d2q7H using 60×60 grid, EP7 using 30×30 grid, and EP7 using 60×60 grid (solid, dashed, dotted, dashed-dotted lines). Figure 2(b) plots the same data against the dimensionless ratio $\Delta t \nu / \Delta x^2$ which facilitates comparison between different grids. Figures 5(c) and 5(d) plot the same data for shear flow. We can see that the Taylor vortex triggers an instability in the explicit projection method EP7 when $\Delta t \nu / \Delta x^2 \geq 0.2$, but the shear flow does not trigger any instability.

With regard to the lattice Boltzmann method we observe that it fails to approximate the solution (has a relative error of 1.0) when M_c is larger than 0.2 approximately. In the case of the Taylor vortex, which is a solution of the incompressible fluid flow equations, it may appear that the problem arises from the compressibility of the lattice Boltzmann fluid (when $M_c \sim 0.2$, the Mach number is approximately $M = 1.53 M_c = 0.3$). In the case of shear flow, however, compressibility is not important. The shear flow is a solution of the compressible fluid flow equations, and it should be easily computed by the lattice Boltzmann method both at low and high Mach numbers. In fact, the shear flow can be computed easily at high Mach numbers by using a smaller w_0, for example $w_0 = 10^{-6}/3$ (see below).

The limitations of the lattice Boltzmann method shown

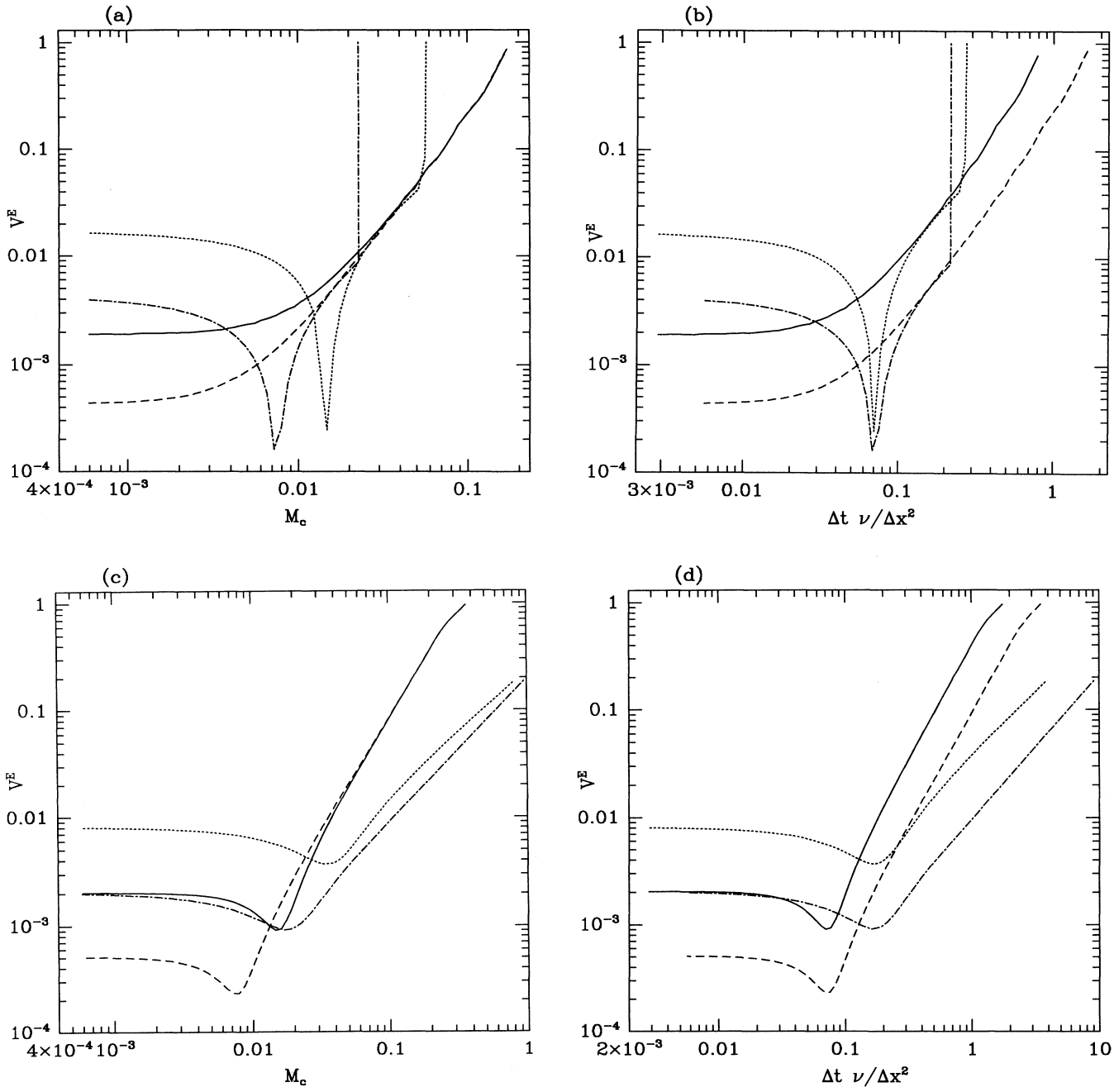

FIG. 5. The error of the lattice Boltzmann method d2q7H is compared against the error of the explicit finite difference projection method EP7. The curves correspond to d2q7H using a 30×30 grid, d2q7H using a 60×60 grid, EP7 using a 30×30 grid, and EP7 using a 60×60 grid (solid, dashed, dotted, dashed-dotted lines). (a) and (b) show the error in simulating the hexagonal Taylor vortex and (c) and (d) show the error in simulating the hexagonal shear flow.

in Fig. 5 when M_c is larger than 0.2 persist independent of the Mach number. The limitations arise because the microscopic speed $\Delta x/\Delta t$ becomes comparable to the fluid speed when M_c approaches 1.0, and the high-order terms in the Chapman-Enskog expansion (which are neglected in deriving the Navier-Stokes equations) become significant, and produce behavior that differs from the Navier-Stokes equations.

With regard to simulating shear flow at high Mach numbers, we can choose $w_0=10^{-6}/3$ which gives $M=10^3 M_c$. The error of the lattice Boltzmann method d2q7H in simulating shear flow with $M=10^3 M_c$ is identical to the error plotted in Fig. 5(c). The error in simulating shear flow is independent of the Mach number because the density gradients are zero everywhere.

E. Quadratic convergence

This section shows that the lattice Boltzmann method has second-order convergence both in space and in time. Second-order convergence in space means that the error decreases quadratically with Δx while keeping the dimensionless ratio $\Delta t\nu/\Delta x^2$ constant ([19], p. 75). Second-order convergence in time means that the error decreases quadratically with Δt while keeping the space discretization Δx constant. Furthermore, we are interested in the true discretization error and not the error that arises from compressibility. When using a compressible fluid code such as the lattice Boltzmann method to simulate incompressible flow such as the Taylor vortex, it is important to distinguish between the error that arises from compressibility and the error that arises from finite discretization.

In Fig. 5 the Mach number decreases in proportion to M_c, and thus the effects of compressibility and finite discretization cannot be distinguished without further analysis. To distinguish between the effects of compressibility and discretization error, we perform the same simulations as those in Fig. 5, while keeping the Mach number constant and varying the density coefficient w_0 as follows:

$$w_0=\frac{1}{3}\left[\frac{\Delta t}{\Delta x M}\right]^2 . \tag{32}$$

In Fig. 6(a) we show the error of d2q7H in simulating the hexagonal Taylor vortex at constant Mach number $M=0.02$ using a 30×30 grid and a 60×60 grid (two dashed lines). For comparison purposes we also show the error of d2q7H using constant $w_0=\frac{1}{7}$ and variable Mach number $M=1.53M_c$ (two solid lines). The constant Mach number curves are identical to the constant w_0 curves except for instabilities which are discussed below. This indicates that the compressible effects at Mach number $M=0.02$ are smaller than the discretization error of both the 30×30 and 60×60 grids. The instability of the constant Mach number curves (dashed lines) is expected and it occurs when the density coefficient w_0 given by Eq. (32) becomes greater than $\frac{1}{6}$ which forces the density coefficient z_0 to become negative. Similar instabilities can be seen in Fig. 6(c) which plots the same experiment for shear flow at constant Mach number $M=0.05$.

It is important to note that if we keep the Mach number constant while decreasing the grid spacing (Δx), then a sufficiently fine grid will eventually bring out the compressible effects. For example, Fig. 6(b) shows the same data as Fig. 6(a) while keeping the Mach number constant at $M=0.1$. In the case of the 30×30 grid the constant Mach number curve is identical to the constant w_0 curve as before, which indicates that the discretization error of the 30×30 grid is larger than the compressible effects of Mach number $M=0.1$. In the case of 60×60 grid however, the constant Mach number curve reaches a minimum error (as Δt goes to zero) that is much greater than the minimum error of the constant w_0 curve. This is because the discretization error of the 60×60 grid becomes smaller than the compressible effects of Mach number $M=0.1$ when $\Delta t\nu/\Delta x^2$ becomes smaller than 0.1 approximately.

In general, we can calculate the Mach number at which compressible effects become larger than the discretization error of any grid by doing more numerical experiments of the kind shown in Fig. 6. Such a study is not necessary for our purposes however. Figures 6(a) and 6(b) are enough to show that the compressible effects in simulating the Taylor vortex are smaller than the discretization error of the 30×30 and 60×60 grids when w_0 is constant and the Mach number varies as $M=1.53M_c$. Accordingly, we can examine the error curves of Fig. 6 and also of Fig. 5 to find out how the discretization error of the lattice Boltzmann method decreases with finer resolution.

If we examine the logarithmic plots of Fig. 5, we see that the error decreases quadratically with Δt (it has a slope of -2) until the minimum spatial discretization error is reached. In addition the error decreases by a factor of 4 when we go from the 30×30 grid to the 60×60 grid while keeping the dimensionless ratio $\Delta t\nu/\Delta x^2$ constant, see Figs. 5(b) and 5(d). In other words, the lattice Boltzmann method has second-order convergence both in space and in time. In Sec. VII we will verify the second-order convergence for boundary value problems also. The explicit finite difference projection method EP7 has first-order convergence in time and second-order convergence in space. The first-order convergence in time of the projection method EP7 can be seen most easily in Figs. 5(c) and 5(d).

F. Seven-speed versus nine-speed

We now compare the accuracy of the hexagonal seven-speed model against the accuracy of the orthogonal nine-speed model. Figure 7 shows the error of d2q7H applied to the hexagonal Taylor vortex, and the error of d2q9H applied to the orthogonal Taylor vortex (solid and dashed lines). In addition the error of the explicit finite difference projection method is shown when the projec-

tion method is applied to the hexagonal Taylor vortex with $\Delta y = \Delta x\sqrt{3}/2$ and also to the orthogonal Taylor vortex with $\Delta y = \Delta x$ (dotted and dashed-dotted lines). A 30×30 grid is used, and the error is calculated at the final time $T = 1.0$. We can see that the explicit finite difference projection method performs similarly on the hexagonal and the orthogonal Taylor vortices. By contrast, the orthogonal nine-speed model d2q9H is significantly more accurate than the hexagonal seven-speed model d2q7H. A simple explanation is that nine speeds per node provide a better discretization of the microscopic velocity [20] than seven speeds per node do.

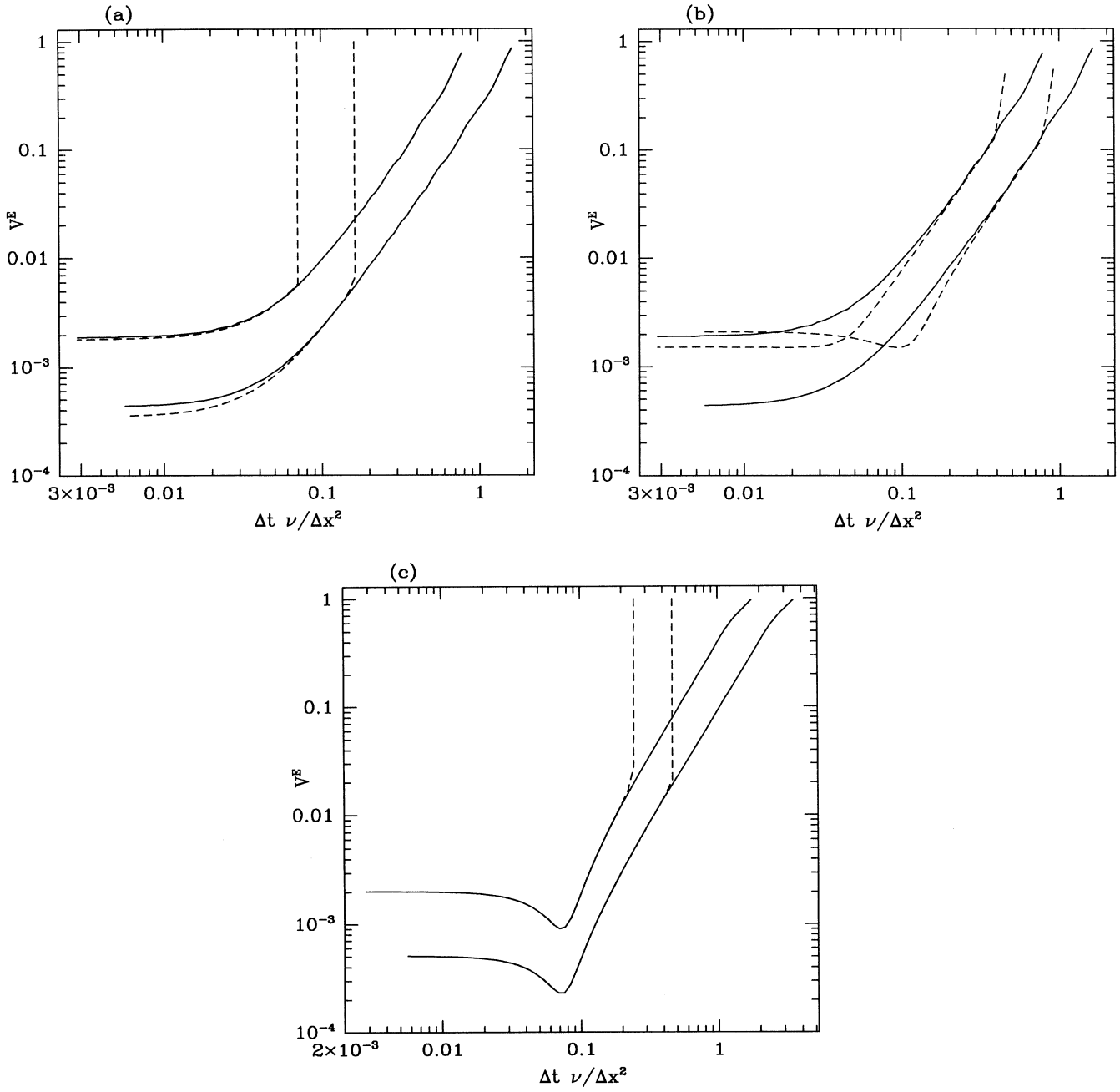

FIG. 6. The error of d2q7H is plotted against M_c with Δt varying, while keeping the Mach number M constant and varying the density parameter w_0 (two dashed lines). For comparison purposes, the error of d2q7H when the Mach number varies and the density parameter $w_0 = \frac{1}{7}$ is held constant is also shown (two solid lines). Results are shown for a 30×30 and a 60×60 grid. (a)–(c) correspond to the hexagonal Taylor vortex at $M = 0.02$, the hexagonal Taylor vortex at $M = 0.1$, and the hexagonal shear flow at $M = 0.05$, respectively.

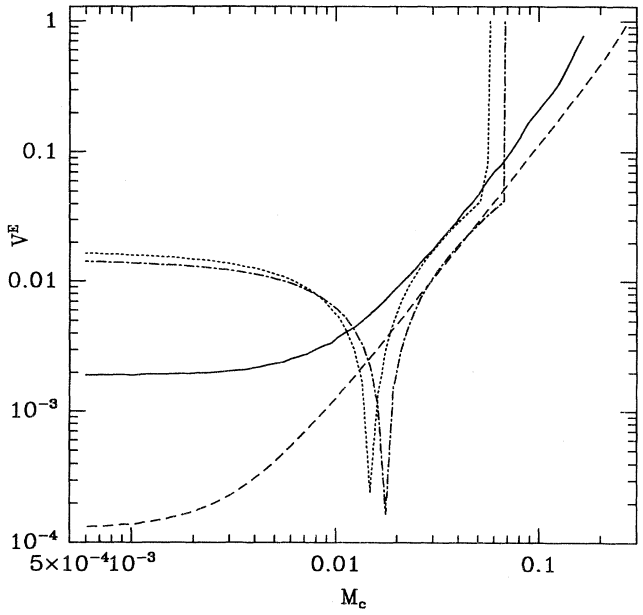

FIG. 7. The error of d2q7H applied to the hexagonal Taylor vortex, and the error of d2q9H applied to the orthogonal Taylor vortex are shown (solid and dashed lines). In addition the error of the explicit finite difference projection method is shown when applied to the hexagonal Taylor vortex with $\Delta y = \Delta x\sqrt{3}/2$ and also the orthogonal Taylor vortex with $\Delta y = \Delta x$ (dotted and dashed-dotted lines).

VII. RESULTS—BOUNDARY VALUE

In this section the orthogonal nine-speed hybrid model d2q9H is applied to several boundary value problems with exact solutions, and is also compared against the explicit finite difference projection method EP9. The boundary value problems are the one-quarter Taylor vortex, the Hagen-Poiseuille flow, and the oscillating plate above a stationary wall. Figure 8 shows the velocity vector fields of these flows, and also indicates the boundary nodes of each flow by drawing a square around the boundary nodes. Figure 8(c) is plotted at time $t=0.4$ when the oscillating plate starts moving to the left while the fluid below is still moving to the right.

The one-quarter Taylor vortex is defined in the region $\pi/2 \le x \le 3\pi/2$ and $\pi/2 \le y \le 3\pi/2$. The exact solution is given by Eq. (26) with $A=B=1$. The velocity and pressure are specified at the boundary by evaluating the exact solution at the horizontal and vertical lines $\pi/2 \le x \le 3\pi/2$ and $\pi/2 \le y \le 3\pi/2$. From the pressure we calculate the density using Eq. (11).

The Hagen-Poiseuille flow is defined in the region $0 \le x \le 1$ and $0 \le y \le 1$. The analytic solution is as follows:

$$\begin{aligned} &V_x(x,y,t) = -(y^2-y)\Delta P/(2\nu) , \\ &V_y(x,y,t) = 0 , \\ &P(x,y,t) = (0.5-x)\Delta P . \end{aligned} \tag{33}$$

The pressure gradient ΔP is chosen $\Delta P = 8.0\nu$ so that the maximum fluid speed is 1.0 when $y = \frac{1}{2}$. The velocity and the density are specified at the boundary by evaluating the exact solution at $0 \le x \le 1$ and $0 \le y \le 1$.

The oscillating plate problem is defined in the region $0 \le x \le 1$ and $0 \le y \le 1$ with periodic boundary conditions in the horizontal direction $x=0$ and 1. The velocity is specified at the top and bottom plates by evaluating the exact solution, namely,

$$\begin{aligned} &V_x = \cos(\omega t) , \quad V_y = 0 , \quad y = 1 , \\ &V_x = 0 , \quad V_y = 0 , \quad y = 0 . \end{aligned} \tag{34}$$

The density at the top and bottom plates is set equal to 1.0 (the exact solution has constant pressure everywhere). The frequency of oscillation ω is chosen $\omega = 20$ so that the oscillating plate executes 3.18 cycles of oscillation during the time interval $T=1.0$ which is used for testing (this is an arbitrary choice). The analytic solution of the oscillating plate problem ([21], p. 88) is given by the following equations:

$$\begin{aligned} V_x(x,y,t) = &[\cosh A \sin A(-2\cosh B \sin B \cos\omega t + 2\cos B \sinh B \sin\omega t) \\ &-\cos A \sinh A(2\cosh B \sin B \sin\omega t + 2\cos B \sinh B \cos\omega t)]/(\cos 2B - \cosh 2B) , \\ V_y(x,y,t) = &0 , \\ P(x,y,t) = &\text{const} , \end{aligned} \tag{35}$$

where $A = y\sqrt{\omega/(2\nu)}$ and $B = \sqrt{\omega/(2\nu)}$, and ν is the kinematic viscosity.

In the case of steady flow such as the Hagen-Poiseuille flow, we initialize ρ, V_x, V_y equal to the exact steady-state solution. Then we iterate for 100 steps, and test whether the fluid is in steady state. If the fluid is in steady state, we measure the velocity relative error V^E. Otherwise we keep iterating until the fluid reaches steady state. The goal of this procedure is to measure the error at steady state and not to characterize how quickly the fluid reaches steady state. Our criterion for steady state is that the relative change in velocity between successive iterations divided by Δt must be less than 10^{-6}; namely we require,

$$\frac{\sum_{x,y} |V_x(t+\Delta t) - V_x(t)|}{\sum_{x,y} |V_x^*|} < 10^{-6}\Delta t \tag{36}$$

and similarly for V_y.

In the case of transient flow such as the one-quarter Taylor vortex and the oscillating plate, we measure the error V^E at the final time $T=1.0$ using Eqs. (28) and (29).

A. Implementation of boundary conditions

The hybrid method d2q9H uses the standard collision operator at the inner nodes, and the extended collision operator at the boundary nodes. An important implementation issue is the calculation of the gradients of the fluid velocity at the boundary nodes. Below we will see that the best results are achieved when the gradients of the fluid velocity are specified using the exact solution. In practice, however, it is possible to specify only some of the velocity gradients at the boundary nodes, and not all of them. For example, the gradient $\partial V_x/\partial y$ at the top and bottom walls of the driven cavity problem (not reported here but see [10], p. 199) cannot be specified because it is part of the solution that one seeks to compute.

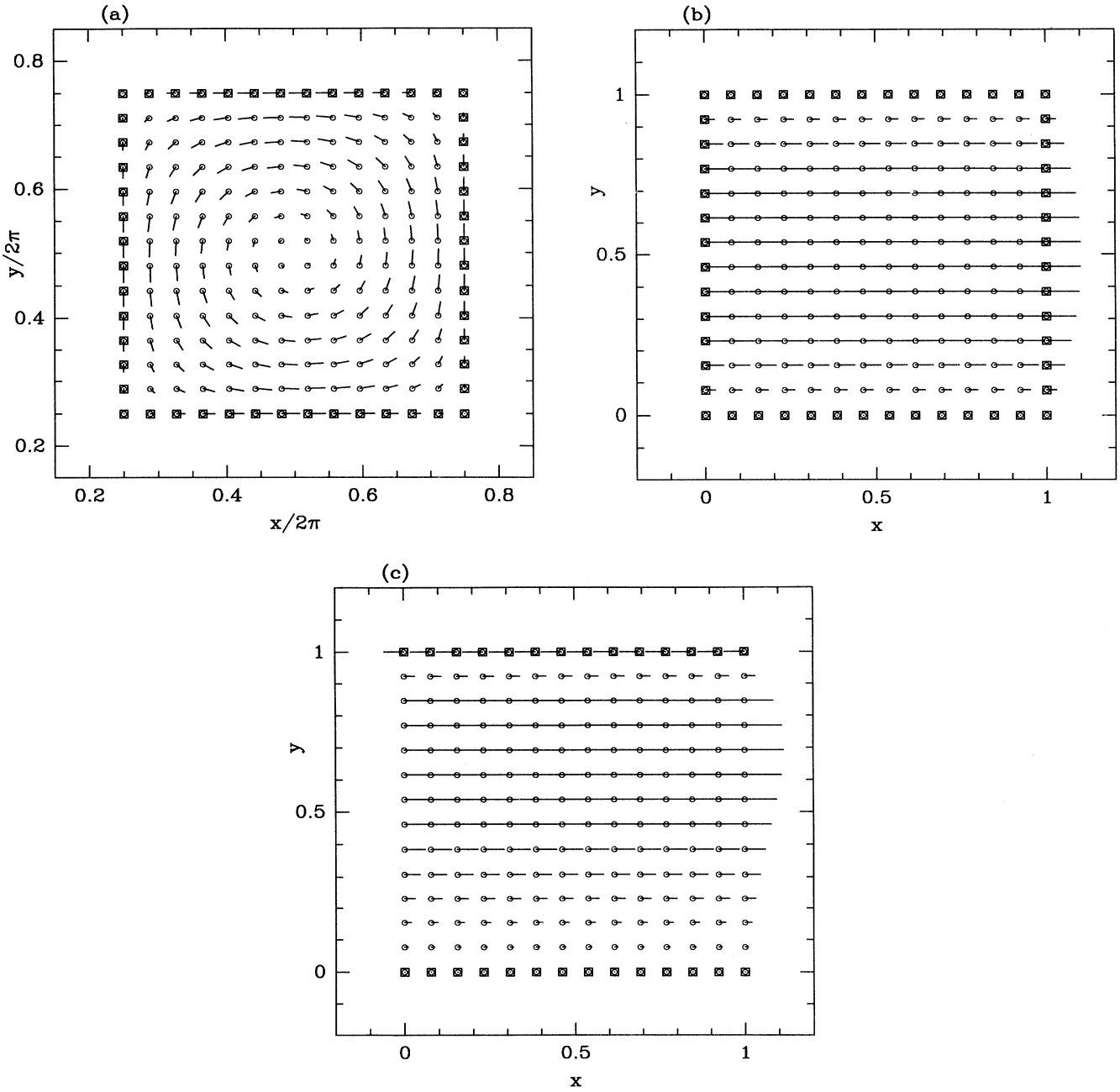

FIG. 8. The velocity field of the one-quarter Taylor vortex, the Hagen-Poiseuille flow, and the oscillating plate problem are shown in (a)–(c), respectively. Boundary nodes are marked with a square. (c) is plotted at time $t=0.4$ when the oscillating plate starts moving to the left and the fluid below is still moving to the right.

When a velocity gradient cannot be specified, finite differences must be used to estimate it.

In our simulations we test two possibilities: first the exact solution is used to specify all of the velocity gradients, and second finite differences are used to calculate all of the velocity gradients at the boundary nodes. When the exact solution is used, we denote the lattice Boltzmann method by $d2q9H_{XD}$ (XD stands for exact derivatives at the boundary). When first-order asymmetric differences are used, we denote the method by $d2q9H_{1FD}$. When second-order asymmetric differences are used, we denote the method by $d2q9H_{2FD}$. We will see that finite differences trigger instabilities when M_c is large, and that first-order differences are more stable than second order differences, but second-order differences are more accurate when M_c is small.

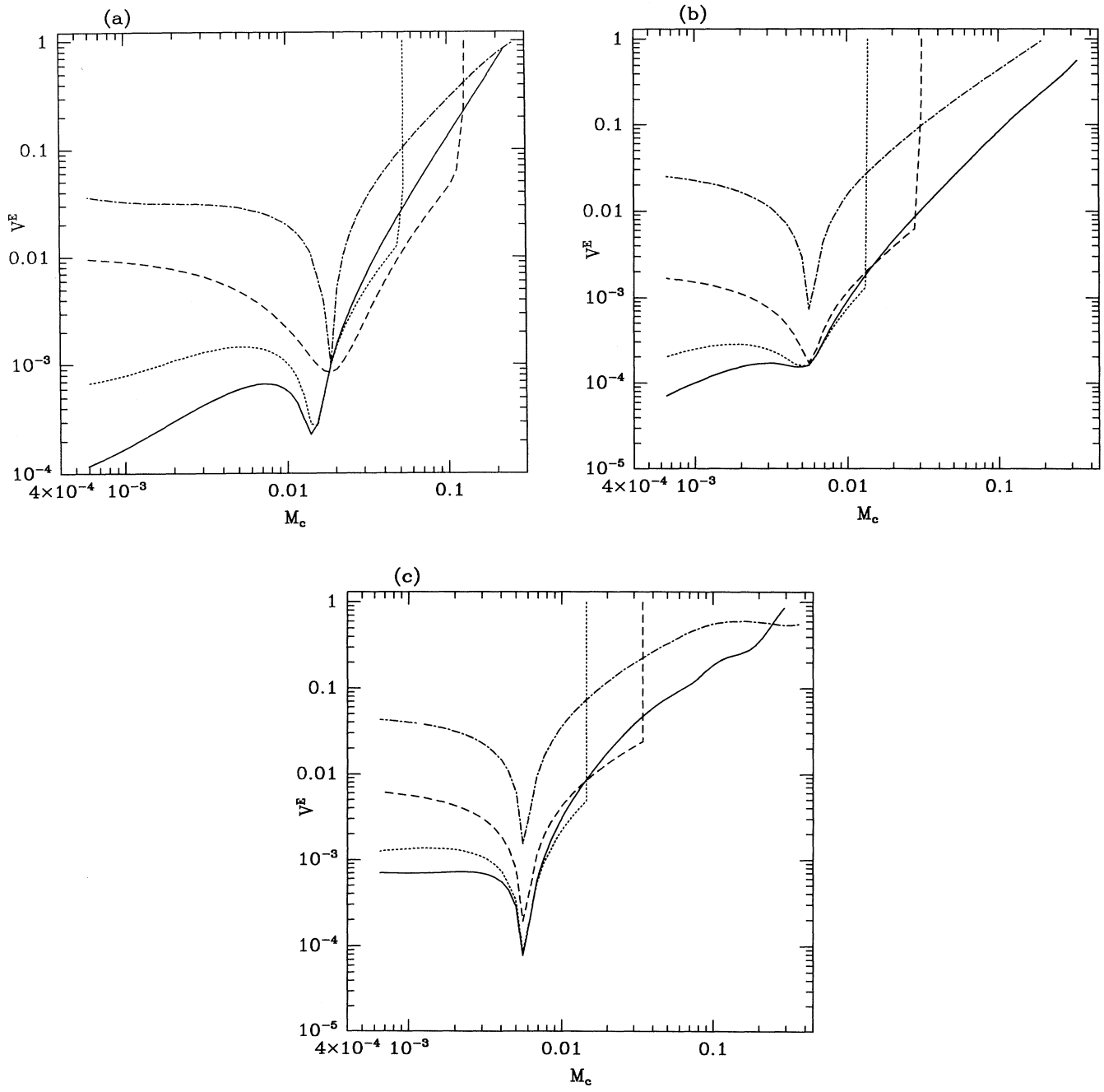

FIG. 9. The error of $d2q9H_{XD}$, $d2q9H_{1FD}$, $d2q9H_{2FD}$, and d2q9F0 (solid, dashed, dotted, and dashed-dotted lines) is shown in simulations of the one-quarter Taylor vortex, the Hagen-Poiseuille flow, and the oscillating plate—figures (a)–(c), respectively.

In our simulations we also test the lattice Boltzmann scheme d2q9F0 which uses the standard collision operator at every node, both boundary and inner nodes. At the boundary nodes the method d2q9F0 sets the populations F_i equal to the equilibrium values F_i^{eq} of the standard collision operator given by Eq. (6). For initialization the method d2q9F0 would normally initialize the F_i equal to the equilibrium values F_i^{eq} of the standard collision operator as described earlier. In this section, however, we use the extended collision operator for initialization in order to distinguish between initial and boundary errors, and we switch to the standard collision operator after the first step.

Regarding boundary conditions for the explicit finite

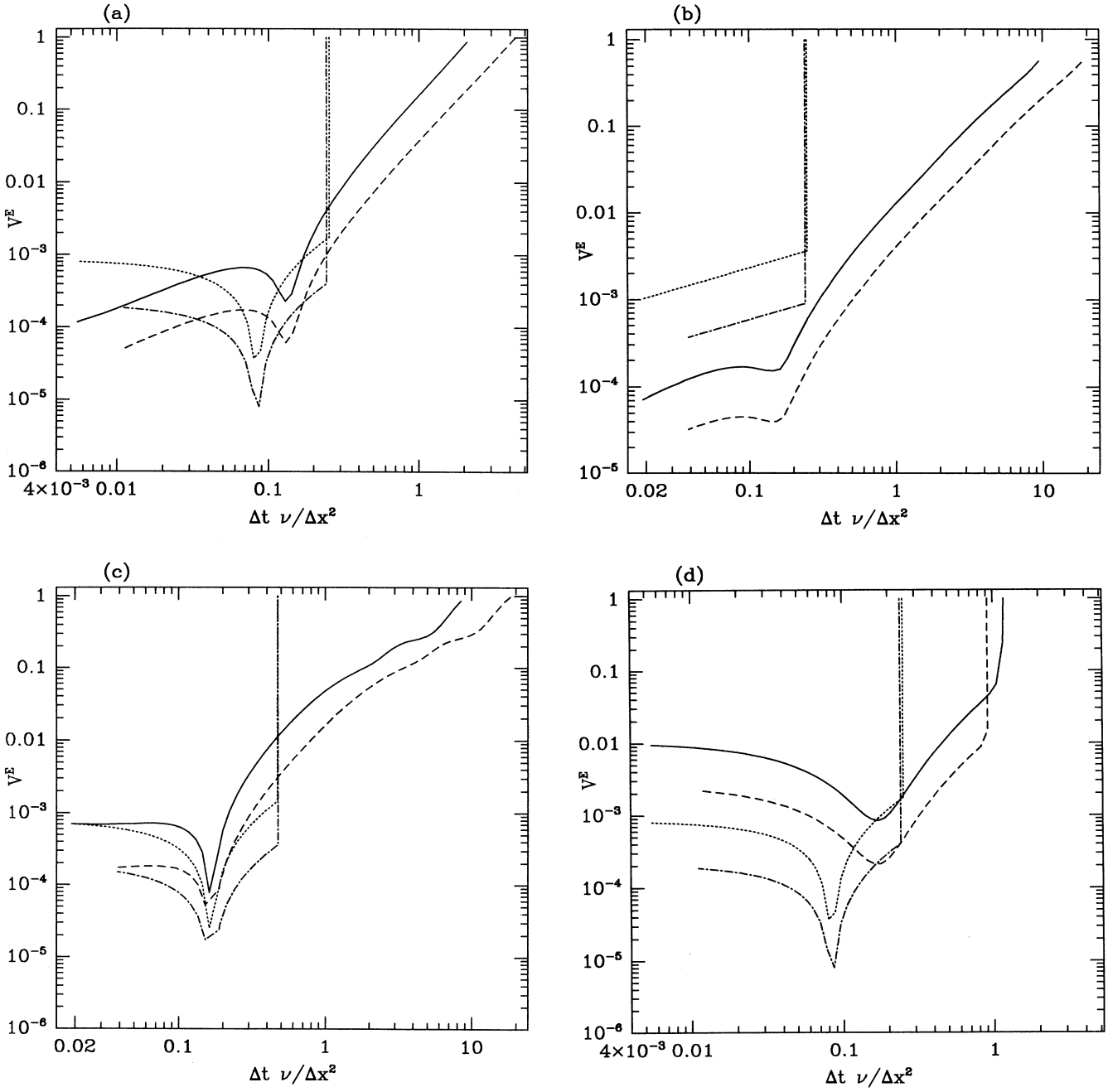

FIG. 10. The error of the lattice Boltzmann method $d2q9H_{XD}$ is compared against the error of the explicit finite difference projection method EP9. The curves correspond to $d2q9H_{XD}$ using 30×30 grid, $d2q9H_{XD}$ using 60×60 grid, EP9 using 30×30 grid, and EP9 using 60×60 grid (solid, dashed, dotted, dashed-dotted lines). (a)–(c) show simulations of the one-quarter Taylor vortex, the Hagen-Poiseuille flow, and the oscillating plate respectively. (d) shows the same experiment as (a) using $d2q9H_{1FD}$ instead of $d2q9H_{XD}$.

difference projection method, the velocity at the boundary is specified from the exact solution, and the pressure P is specified from the requirement $\partial P/\partial n=0$ at the boundary, where ∂n denotes the direction normal to the boundary ([10], p. 160). The condition $\partial P/\partial n=0$ is applied at the beginning of the SOR calculation using the values of P at the previous time step, and the resulting boundary values for the pressure P are held constant throughout the SOR calculation.

B. Comparison of boundary schemes

In Fig. 9 we compare the methods $d2q9H_{XD}$, $d2q9H_{1FD}$, $d2q9H_{2FD}$, and d2q9F0 (solid, dashed, dotted, and dashed-dotted lines) in simulations of the one-quarter Taylor vortex, the Hagen-Poiseuille flow, and the oscillating plate, Figs. 9(a)–9(c), respectively. A 30×30 grid is used, and the error is plotted against M_c with Δt varying, and is calculated at the final time $T=1.0$. The standard collision operator d2q9F0 achieves its smallest error when the relaxation parameter $\tau=1$, at which point the standard and extended collision operators are identical. The hybrid method achieves best results overall when the velocity gradients at the boundary nodes are specified from the exact solution (method $d2q9H_{XD}$). The use of finite differences at the boundary ($d2q9H_{1FD}$ and $d2q9H_{2FD}$) leads to instabilities when M_c is large. First order differences are more stable than second-order differences, while second order differences are more accurate when M_c is small.

C. Comparison with projection method-convergence

Figure 10 compares the error of the lattice Boltzmann method $d2q9H_{XD}$ against the error of the explicit finite difference projection method EP9 in simulations of the one-quarter Taylor vortex, the Hagen-Poiseuille flow, and the oscillating plate, Figs. 10(a)–10(c), respectively. The error is plotted against the dimensionless ratio $\Delta t\nu/\Delta x^2$ to facilitate comparison between different grids. The curves correspond to $d2q9H_{XD}$ using a 30×30 grid, $d2q9H_{XD}$ using a 60×60 grid, EP9 using a 30×30 grid, and EP9 using a 60×60 grid (solid, dashes, dotted, dashed-dotted lines). Figure 10(b) shows most clearly the rate of convergence in time. The lattice Boltzmann method has second-order convergence in time (slope -2), and the finite difference projection method EP9 has first-order convergence in time (slope -1). Both methods have second-order convergence in space.

We also note that the use of first-order differences to calculate the velocity gradients at the boundary nodes does not change the overall second order convergence of the lattice Boltzmann method. This can be seen in Fig. 10(d) which corresponds to the same experiment as Fig. 10(a) but uses the method $d2q9H_{1FD}$ (first-order differences to calculate the velocity gradients at the boundary nodes) instead of the method $d2q9H_{XD}$ (exact values for the velocity gradients at the boundary nodes). Theoretical explanations for the second order convergence of the lattice Boltzmann method are discussed in Refs. [20] and [22].

D. Future work — density at the boundary

In all of our test cases above the density ρ is assumed to be known at the boundary and is specified by evaluating the exact solution. In many practical simulations however, it is appropriate to consider ρ unknown at the boundary. In general there may be density gradients along the boundary that develop as a result of the fluid dynamics. For example, in the driven cavity problem a density gradient develops along the walls of the cavity (pressure gradient divided by the square of the speed of sound) which is part of the fluid flow solution that one seeks to compute. In such cases the ρ must be calculated dynamically from the simulated flow. Although we will not present simulation results concerning this problem here, we will describe one possible approach of calculating the density dynamically at the boundary.

A good method of calculating the density dynamically at the boundary is to calculate ρ as the average of the populations F_i that "bring fluid into the boundary node" from neighboring nodes such as inner nodes and/or other neighboring boundary nodes. For example, in the case of a horizontal wall that bounds the fluid region from below, the density ρ must be calculated as the average of the populations F_0,F_1,F_5,F_6,F_7,F_8 when using an orthogonal grid (see the Appendix). The populations F_2,F_3,F_4 must be omitted in this calculation because they convect into the bottom wall from the outside of the fluid region. Similar calculations of the density must be done for all possible orientations of the boundary. In our simulations of the driven cavity problem (not reported here) and other flows past obstacles we have obtained good results using this approach (this is a qualitative judgment). Future work must be done to evaluate further this kind of dynamic calculation of the density at the boundary.

APPENDIX: ORTHOGONAL 9-SPEED MODEL

We consider an orthogonal (square) lattice with nine populations at each node. The population F_0 is nonmoving, the populations F_i^{II}, $i=2,4,6,8$ move along the diagonal directions at the speed $\sqrt{2}c$, and the populations F_i^{I}, $i=1,3,5,7$ move along the vertical and horizontal directions at the speed $c=\Delta x/\Delta t$. We denote the orthogonal nine-speed model with the symbol d2q9 following Ref. [2]. The relaxation and convection steps are given by the following formulas:

$$F_i(\mathbf{x}+\mathbf{e}_i\Delta t,t+\Delta t)=F_i(\mathbf{x},t)+(-1/\tau)[F_i(\mathbf{x},t)-F_i^{eq}(\mathbf{x},t)]\ ,$$

$$F_0(\mathbf{x},t+\Delta t)=F_0(\mathbf{x},t)+(-1/\tau)[F_0(\mathbf{x},t)-F_0^{eq}(\mathbf{x},t)]\ ,$$

$$i=1,\ldots,8\ , \tag{A1}$$

$$\tau=\frac{1}{2}+\frac{3\Delta t\nu}{\Delta x^2}\ .$$

The relaxation parameter τ is chosen to achieve the desired kinematic viscosity ν given the space and time discretization parameters $\Delta x,\Delta t$. The vector $\mathbf{e}_i$ stands for the eight velocity directions of the orthogonal (square)

lattice,

$$\mathbf{e}_i=\frac{\Delta x}{\Delta t}\left[\cos\frac{2\pi(i-1)}{8},\sin\frac{2\pi(i-1)}{8}\right] . \tag{A2}$$

The velocity $\mathbf{V}(\mathbf{x},t)$ and density $\rho(\mathbf{x},t)$ are computed from the populations $F_i(\mathbf{x},t)$ using the relations

$$\begin{aligned}\rho(\mathbf{x},t)&=\sum_{i=0}^{8}F_i(\mathbf{x},t) ,\\ \rho(\mathbf{x},t)\mathbf{V}(\mathbf{x},t)&=\sum_{i=1}^{8}F_i(\mathbf{x},t)\mathbf{e}_i .\end{aligned} \tag{A3}$$

The variations of density around its mean value (spatial mean which is constant in time) provide an estimate of the fluid pressure $P(\mathbf{x},t)$, according to the following equation:

$$P(\mathbf{x},t)=c_s^2[\rho(\mathbf{x},t)-\langle\rho\rangle] . \tag{A4}$$

The speed of sound is

$$c_s=\sqrt{(2w_0+4y_0)}(\Delta x/\Delta t) , \tag{A5}$$

where the coefficients w_0,y_0 are discussed below. The equilibrium populations $F_i^{eq}(x,t)$ are given by the following equations:

$$\begin{aligned}&F_i^{\mathrm{eqII}}=\rho[y_0+y_1(\mathbf{e}_i\cdot\mathbf{V})+y_{20}(\mathbf{e}_i\cdot\mathbf{V})(\mathbf{e}_i\cdot\mathbf{V})+y_{21}(\mathbf{V}\cdot\mathbf{V})] ,\\ &F_i^{\mathrm{eqI}}=\rho[w_0+w_1(\mathbf{e}_i\cdot\mathbf{V})+w_{20}(\mathbf{e}_i\cdot\mathbf{V})(\mathbf{e}_i\cdot\mathbf{V})+w_{21}(\mathbf{V}\cdot\mathbf{V})] ,\\ &F_0^{\mathrm{eq}}=\rho[z_0+z_{21}(\mathbf{V}\cdot\mathbf{V})] ,\\ &4w_0+4y_0+z_0=1 ,\\ &y_1=1/(12c^2) ,\quad y_{20}=1/(8c^4) ,\quad y_{21}=-1/(24c^2) ,\\ &w_1=1/(3c^2) ,\quad w_{20}=1/(2c^4) ,\quad w_{21}=-1/(6c^2) ,\\ &z_{21}=-2/(3c^2) ,\quad c=\Delta x/\Delta t .\end{aligned} \tag{A6}$$

In our simulations we use $y_0=\frac{1}{4}w_0$ and $w_0=\frac{1}{7}$ unless otherwise indicated. The shear and bulk viscosity of the d2q9 collision operator have the following values (calculated using the Chapman-Enskog procedure):

$$\begin{aligned}\nu&=\frac{c^2\Delta t}{6}(2\tau-1) ,\\ \mu&=\frac{c^2\Delta t}{3}(2\tau-1)(1\quad 3w_0-6y_0) .\end{aligned} \tag{A7}$$

The extended collision operator for the orthogonal nine-speed model (d2q9X) is derived similarly to the hexagonal model of Sec. IV A. Two additional terms based on gradients of the fluid velocity are included in the equilibrium population formulas. Everything else, including all the coefficients $w_1,y_1,w_{20},\ldots,$ of the standard collision operator d2q9 remain the same. The equilibrium population formulas for d2q9X are as follows:

$$\begin{aligned}F_i^{*\mathrm{eqII}}&=\rho[y_0+y_1(\mathbf{e}_i\cdot\mathbf{V})+y_{20}(\mathbf{e}_i\cdot\mathbf{V})(\mathbf{e}_i\cdot\mathbf{V})+y_{21}(\mathbf{V}\cdot\mathbf{V})]\\ &\quad+y_{31}[\mathbf{e}_i\cdot\nabla(\mathbf{e}_i\cdot\rho\mathbf{V})]+y_{32}(\nabla\cdot\rho\mathbf{V}) ,\\ F_i^{*\mathrm{eqI}}&=\rho[w_0+w_1(\mathbf{e}_i\cdot\mathbf{V})+w_{20}(\mathbf{e}_i\cdot\mathbf{V})(\mathbf{e}_i\cdot\mathbf{V})+w_{21}(\mathbf{V}\cdot\mathbf{V})]\\ &\quad+w_{31}[\mathbf{e}_i\cdot\nabla(\mathbf{e}_i\cdot\rho\mathbf{V})]+w_{32}(\nabla\cdot\rho\mathbf{V}) ,\\ F_0^{*\mathrm{eq}}&=\rho[z_0+z_{21}(\mathbf{V}\cdot\mathbf{V})]+z_{32}(\nabla\cdot\rho\mathbf{V}) ,\\ 2c^2w_{31}+4w_{32}+4c^2y_{31}+4y_{32}+z_{32}&=0 ,\\ y_{31}&=w_{31}/4 .\end{aligned} \tag{A8}$$

The velocity gradients are computed using finite differences unless they are known by other means. In our simulations we use second-order symmetric differences ([10], p. 19) at the inner nodes, and first- or second-order asymmetric differences at the boundary nodes as discussed in Sec. VII. First-order differences at the boundary are more stable for large M_c, while second-order asymmetric differences at the boundary are more accurate for small M_c.

The shear and bulk viscosities of the d2q9X operator have the following values (calculated using the Chapman-Enskog procedure):

$$\begin{aligned}\nu^*&=\frac{c^2\Delta t}{6}(2\tau^*-1)-c^4w_{31} ,\\ \mu^*&=\frac{c^2\Delta t}{3}(2\tau^*-1)(1-3w_0-6y_0)\\ &\quad-2c^4w_{31}-2c^2(w_{32}+2y_{32}) .\end{aligned} \tag{A9}$$

In our simulations we use $y_{32}=w_{32}/4$. Once the relaxation parameter τ^* is set equal to one, the coefficient w_{31} is chosen to achieve the desired kinematic viscosity given the discretization parameters $\Delta x,\Delta t$. The coefficient w_{32} is chosen to achieve the desired bulk viscosity. In the case of hybrid method d2q9H, the bulk viscosity of Eq. (A9) is chosen equal to the bulk viscosity of the standard collision operator given by Eq. (A7). The coefficient z_{32} must satisfy $2c^2w_{31}+4w_{32}+4c^2y_{31}+4y_{32}+z_{32}=0$ so that the equilibrium populations conserve mass, and the coefficient y_{31} is set equal to $w_{31}/4$ so that an unwanted (anisotropic) momentum diffusion term in the Chapman-Enskog expansion vanishes.

[1] S. Chen, Z. Wang, X. Shan, and G. D. Doolen, J. Stat. Phys. **68**, 379 (1992).
[2] Y. H. Qian, D. d'Humières, and P. Lallemand, Europhys. Lett. **17**, 479 (1992).
[3] H. Chen, S. Chen, and W. H. Matthaeus, Phys. Rev. A **45**, 5339 (1992).
[4] J. M. V. A. Koelman, Europhys. Lett. **15**, 603 (1991).
[5] F. J. Higuera and J. Jimenez, Europhys. Lett. **9**, 663 (1989).
[6] M. Vergassola, R. Benzi, and S. Succi, Europhys. Lett. **13**, 411 (1990).
[7] A. K. Gunstensen, D. H. Rothman, S. Zaleski, and G. Zanetti, Phys. Rev. A **43**, 4320 (1991).
[8] The equivalence of the hybrid method and the Chapman-

Enskog expansion was first noticed by Dominique d'Humières, who kindly communicated this result to the author.

[9] R. Cornubert, D. d'Humières, and D. Levermore, Physica D **47**, 241 (1991).

[10] R. Peyret and T. D. Taylor, *Computational Methods for Fluid Flow* (Springer-Verlag, New York, 1990).

[11] S. Wolfram, J. Stat. Phys. **45**, 471 (1986); available also in *Lattice Gas Methods for Partial Differential Equations*, edited by G. D. Doolen (Addison-Wesley, Reading, MA, 1990), pp. 19–73.

[12] Y. H. Qian and S. A. Orszag, Europhys. Lett. **21**, 255 (1993).

[13] A related issue regarding lattice gas methods is discussed by S. Chen, Z. She, L. C. Harrison, and G. D. Doolen, Phys. Rev. A **39**, 2725 (1989).

[14] The addition of a viscosity Laplacian term to the first-order Chapman-Enskog expansion (for the purpose of conserving momentum) does not change the derivation of the Navier-Stokes equations via the Chapman-Enskog procedure because the corresponding corrections are higher-order derivatives than the Navier-Stokes equations.

[15] U. Frisch *et al.*, Complex Syst. **1**, 649 (1987), see Eq. (8.20).

[16] D. d'Humières and P. Lallemand, Complex Syst. **1**, 599 (1987), see Eq. (A8).

[17] G. I. Taylor, Philos. Mag. **46**, 671 (1923).

[18] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C* (Cambridge University Press, New York, 1988).

[19] C. A. J. Fletcher, *Computational Techniques for Fluid Dynamics* (Springer-Verlag, Berlin, 1988), Vol. 1.

[20] P. A. Skordos (unpublished).

[21] L. D. Landau and E. M. Lifshitz, *Fluid Mechanics* (Pergamon, New York, 1989), 2nd ed.

[22] M. G. Ancona (unpublished).