Finite-volume versus streaming-based lattice Boltzmann algorithm for fluid-dynamics simulations: A one-to-one accuracy and performance study

Kalyan Shrestha,* Gilmar Mompean, and Enrico Calzavarini

Laboratoire de Mécanique de Lille CNRS/UMR 8107, Université Lille 1 & Polytech’Lille, Cite Scientifique, Av. P. Langevin, F-59650 Villeneuve d’Ascq, France

(Received 11 August 2015; published 22 February 2016)

A finite-volume (FV) discretization method for the lattice Boltzmann (LB) equation, which combines high accuracy with limited computational cost is presented. In order to assess the performance of the FV method we carry out a systematic comparison, focused on accuracy and computational performances, with the standard streaming lattice Boltzmann equation algorithm. In particular we aim at clarifying whether and in which conditions the proposed algorithm, and more generally any FV algorithm, can be taken as the method of choice in fluid-dynamics LB simulations. For this reason the comparative analysis is further extended to the case of realistic flows, in particular thermally driven flows in turbulent conditions. We report the successful simulation of high-Rayleigh number convective flow performed by a lattice Boltzmann FV-based algorithm with wall grid refinement.

DOI: 10.1103/PhysRevE.93.023306

I. INTRODUCTION

Since its introduction, the lattice Boltzmann (LB) equation method for fluid-dynamics simulations has enjoyed increasing success [1]. The reasons are twofold: on one hand, the mesoscopic level of description on which it is based goes beyond the Navier-Stokes continuum matter description of fluids and it eases, as compared to other macroscopic methods, to accommodate for complex effects, such as the interaction between different fluid components, phase-change processes, and non-Newtonian rheology. The extensions of the LB methods in such directions are countless (e.g., multiphase and multicomponent flows [2–7], flows with suspensions [8–12], emulsions [13], porous media [14–17], natural convection [18], reactive transport [19,20], combustion [21,22], and magnetohydrodynamics [23,24]). On the other hand, the LB method has also very appealing features from a computational point of view. It is simple to implement, free of numerical diffusion and stability issues, and suitable for parallelization due to its local-in-space character.

However, when it comes to the simulation of turbulent flows one shortcoming of the method, the limitation to equispaced grids, becomes evident. We recall here that a developed turbulent flow in the presence of any sort of bounding geometry (or any local forcing term) develops space inhomogeneities and as such grid refinement in numerics becomes necessary. It shall be made clear that in such a context, grid refinement is not an additional requirement in order to increase the accuracy of a simulation but is rather an unavoidable need in order to save memory usage and computational power and being able to access higher—read more realistic—turbulent flows regimes.

In summary any state-of-the-art computational fluid dynamics method (CFD) calls for grid refinement.

Several approaches have been proposed in order to overcome the shortcoming of equi-spaced grids in lattice Boltzmann (LB) equation simulations. Here we mention (i) the grid refinement methods, which make use of locally nested equispaced grids [25], (ii) the techniques based on off-lattice interpolation schemes [26,27], (iii) the finite-difference [28], finite-volume [29,30], or finite-elements [31] LB discretization methods, and (iv) the extension of the LB equation to general manifolds [32]. There are, however, important drawbacks. All such reformulation are computationally more expensive, or introduce extra stability limitations enforced by space and/or time discretization, which were not present in the original so-called streaming-based implementation. Presently, the only viable way seems to be the nested monospaced grid method (i), which has allowed us to simulate turbulent channel flows [33] and even more complex flow geometries [34]. However, in this case the advantage in terms of accuracy and efficiency compared to state-of-the-art direct numerical simulation (DNS), e.g., spectral methods is limited.

This paper presents a finite-volume (FV) discretization method for the lattice Boltzmann equation, which besides a high level of accuracy also displays a contained computational cost. In order to assess the performance of this FV method we carry out a systematic comparison with the standard streaming (ST) formulation. We present a methodical comparison of accuracy and computational performances. We aim at clarifying whether and in which conditions the proposed FV algorithm can be taken as the method of choice in fluid-dynamics LB simulations.

The paper is organized as follows. In the next section we describe two different discretizations of the LB equation. To begin with, we briefly review the key points of the streaming implementation (Sec. II A), then we present the FV-based formulation (Sec. II B). Our guidelines in the development of the FV method are the the simplicity and computational efficiency of the implementation, yet retaining a level of accuracy, which takes the ST method as the baseline. Results of this study are reported in Sec. III. First, we address the accuracy comparison, later on the computational efficiency of the algorithm. A further section (Sec. V) takes the analysis to more realistic flows, in particular thermal flows in turbulent conditions. Here, a high-Rayleigh number convective flow with wall grid refinement is simulated by a LB-based code. Final remarks and perspectives are reported in the conclusions.
II. METHOD

We focus here on the LB method with the Bhatnagar-Gross-Krook (BGK) collision operator, which is characterised by a single relaxation time $\tau$ towards a local equilibrium state. The equation of motion reads:

$$\frac{\partial f_\alpha}{\partial t} + \mathbf{c}_\alpha \cdot \nabla f_\alpha = \frac{1}{\tau} (f_\alpha^{eq} - f_\alpha) + F_\alpha \quad \alpha = 0, \ldots, N_{\text{pop}},$$

where $f_\alpha(x,t)$ is one of the $N_{\text{pop}}$ distribution functions for the lattice populations $\mathbf{f}_\alpha$ at position $x$ and time $t$. The set of $\mathbf{f}_\alpha$ distribution functions relax towards a local equilibrium state $f_\alpha^{eq}(x,t)$, which is prescribed in terms of local macroscopic variables (in nonthermal models, as here, they are just the fluid velocity and density) [1]. The macroscopic fluid mass and momentum density can be computed as

$$\rho = \sum_\alpha f_\alpha \quad \text{and} \quad \rho \mathbf{u} = \sum_\alpha f_\alpha \mathbf{c}_\alpha,$$

where the constant $\mathbf{c}_\alpha$ stands for the so-called lattice speed of sound, whose value depends on the specific velocity lattice topology (see for instance Succi’s book [1] for details). Finally, $F_\alpha$ is a forcing term, constructed in such a way to model the effect of a macroscopic body force term.

Note that Eq. (1) is discrete just in the velocity space (for this reason it is also known as discrete velocity Boltzmann equation). Up to this level no discretization has been taken, either in the spatial domain or in the temporal one. Such further discretizations can take different paths as we describe in the following sections.

A. Outline of the streaming lattice Boltzmann algorithm

It is here useful to briefly recall the steps that have to be made in order to obtain from Eq. (1) the standard streaming (ST) LB algorithm. First, by applying to the above partial differential equation (PDE) the technique of the characteristics (ST) LB algorithm. First, by applying to the above partial differential equation (ODE) the semi-implicit Crank-Nicolson technique of the characteristics

$$\frac{df_\alpha}{dt} + \mathbf{c}_\alpha \cdot \nabla f_\alpha = \frac{1}{\tau} (f_\alpha^{eq} - f_\alpha) + F_\alpha,$$

where $f_\alpha(x,t)$ is one of the $N_{\text{pop}}$ distribution functions for the lattice populations $\mathbf{f}_\alpha = f_\alpha \Delta \mathbf{c}_\alpha = f_\alpha^{eq} \mathbf{c}_\alpha$, which makes the scheme explicit, leading to [35]:

$$\begin{align*}
\tilde{f}_\alpha(x + \mathbf{c}_\alpha \Delta t, t + \Delta t) &= f_\alpha(x,t) + \Delta t \left[ f_\alpha^{eq}(x,t) - f_\alpha(x,t) \right] \\
&\quad + \Delta t \left( 1 - \frac{\Delta t}{2\tau} \right) F_\alpha,
\end{align*}$$

(3)

where $\tilde{\tau} = \tau + \Delta t/2$ is a redefined relaxation time ($\tilde{\tau} > \Delta t/2$). It is easy to derive the relations between the macroscopic variables and the tilded ($\tilde{\cdot}$) quantities, they are respectively:

$$\rho = \sum_\alpha \tilde{f}_\alpha \quad \rho \mathbf{u} = \sum_\alpha \mathbf{c}_\alpha \tilde{f}_\alpha \quad \text{and} \quad \nu = \tilde{\tau} \Delta t/2 \mathbf{c}_\alpha^2.$$

Note that a factor $1 - \frac{\Delta t}{2\tau}$ in front of the forcing term needs to be introduced after discretization to give the same hydrodynamics limits as (1) [36].

The numerical implementation of (3) is straightforward. It can be divided in two steps: (i) the computation of the right-hand side and (ii) the displacement (or streaming) of the computed values on the lattice according to the direction and intensity of $\mathbf{c}_\alpha$. It is important to note that the integration along the characteristics introduces a link between the space and time discretization, which reads $\mathbf{c}_\alpha = \Delta \mathbf{x}_\alpha / \Delta t$. This means that if one chooses the Cartesian components of the set of $\mathbf{c}_\alpha$ velocities to be either ±1 or 0, it implies that $\Delta \mathbf{x}_\alpha,t = \Delta t$ or 0. The standard choice (but not the only possible one) is $\Delta \mathbf{x}_\alpha,t = \Delta t = 1$ [1].

B. Lattice Boltzmann finite-volume formulation

The method of characteristics is very convenient from a computational point of view because it reduces the complexity of the integration of a PDE to a simple ODE, however, at the same time it introduces a tight link between the shape of the velocity lattice and the spatial discretization mesh. Such a constraint can be removed if one takes the more usual numerical approach based on (i) a direct spatial discretization of Eq. (1) combined with (ii) an independent time discretization phase. For the first step, several standard options are available, such as finite elements, finite differences, or finite-volume methods.

The idea of using a finite-volume method to decouple the spatial numerical mesh from the velocity lattice structure was first proposed by Namelli and S. Succi [30] (see also Ref. [29]). In this seminal paper a low-order upwind scheme was suggested for the discretization of the advection (or flux) term. The idea was further refined in Amati et al. [37], where piecewise linear interpolation scheme was suggested for the treatment of the flux term. While these first works were limited to stretched Cartesian grids, Chen [38] presented a volumetric formulation, based on a cell-centered discretization scheme, which allowed for the adoption of arbitrary structured meshes. The formulation was further developed by Peng et al. [39,40] through cell-vertex FV scheme, which displayed enhanced stability properties. Sbragaglia and Sugiyama [41] applied Peng’s scheme to an energy-conserving LB model to study for the first time thermal convective flows. More recently Ubertini et al. [42] addressed the problem of unstructured bidimensional triangular meshes, which allow great flexibility on one hand, but also reintroduce known issues related to numerical stability. This has been further refined in a work by Zarzhami et al. [43] through a cell-centered FV approach on arbitrary mesh in two dimensions. A total variation diminishing (TVD) formulations for LB FV algorithm has been suggested by Patil et al. [44], where stability and accuracy can be efficiently enhanced. Despite all these contributions, at present the situation is still far from being solved. If on one side it has been shown that a satisfactory level of precision can be reached by the FV method, on the other hand this is often at the price of the high computational costs needed to obtain a stable algorithm. As an example, in a recent work [43], where a series of laminar but relatively complex flows over nonhomogeneous meshes were simulated, a fifth-order Runge-Kutta scheme had to be adopted for the time discretization in order to have stable results. The consequence on the computational cost is evident since in
such a scheme the advection terms of (1) need to be computed five times per time step (while in the ST method it is performed just by means of a memory shift, the streaming). As a consequence the FV LB is rarely a method of choice in fluid-dynamics simulations (see also the discussions in Refs. [45] and [46]).

The present paper further develops the finite-volume lattice Boltzmann method in order to simulate fluid flow problems with higher accuracy, greater stability properties, and comparable performance as the ST method. The FV method that we propose is of the type denoted as cell centered (as opposed to vertex centered, see Fig. 1(a)). Its most original features concern the approach taken for the time discretization (Sec. II B 2) and the method of fluxes computation, which adopt a quadratic upwind interpolation scheme (QUICK) for the estimating the value of $f_{\alpha}$ at the cell boundary position $x_s$. Note that the interpolation method makes use of different nodes according to the direction of the population velocity $\vec{c}_{\alpha}$.

1. Space discretization

Upon integration of Eq. (1) over a volume $V$ (of surface $S$) and by applying the flux theorem we get

\[
\int_V \frac{\partial f_{\alpha}}{\partial t} \, dV + \int_S \vec{c}_\alpha \cdot \vec{n} \, f_{\alpha} \, dS = \int_V \frac{1}{\tau} (f_{\alpha}^{eq}-f_{\alpha}) \, dV + \int_V F_{\alpha} \, dV.
\]

We then assume that every term in the volume integrals can be considered as constant and its magnitude taken at a reference location $x$ (also called node) inside $V$.

The term in the surface integral, however, carries some kind of spatial variability. When such a surface is decomposed in $M$ faces (as in a structured grid of nodes with connectivity index $M$) it is convenient to make the assumption that $f_{\alpha}$ is constant on each of the $S_j$ surfaces perpendicular to $\vec{n}_j$ and denoting its value with $[f_{\alpha}]_j$. This altogether leads to:

\[
\frac{\partial f_{\alpha}}{\partial t} + \frac{S_j}{V} \vec{c}_\alpha \cdot \vec{n}_j [f_{\alpha}]_j = \frac{1}{\tau} (f_{\alpha}^{eq}-f_{\alpha}) + F_{\alpha},
\]

where summation over the repeated index $j$ is applied.

2. Time discretization

If the time derivative is discretized by the explicit Euler scheme, we get:

\[
f_{\alpha}^{(t+\Delta t)} = f_{\alpha}^{(t)} - \Delta t \frac{S_j}{V} \vec{c}_\alpha \cdot \vec{n}_j [f_{\alpha}^{eq}]_j + \frac{\Delta t}{\tau} \left( f_{\alpha}^{eq} - f_{\alpha}^{(t)} \right) + \Delta t F_{\alpha}.
\]

where the superscript indexes $(t)$ and $(t+\Delta t)$ denote, respectively, the current and the next discrete time instant. Such an approach, however, puts tight bounds on the maximum allowed $\Delta t$. It is easy to show that if we discard the advection and the forcing terms and assume $f^{eq}$ to be constant, the stability region of the method is $0 < \Delta t \leq 2\tau$. Empirically it is possible to show that this range becomes even narrower when the nonlocal advection term, the forcing and the time dependency in $f^{eq}$ are taken into account. The fact that $\Delta t_{\text{max}}$ depends on and is bounded by the value of $\tau$ is a known problem in FV LB implementations. It poses, among others, a severe limitation for the simulations of turbulent flows (i.e., low viscosity flows). On the opposite, such a constraint does not exist in the ST approach (where $\Delta t$ is independent of $\tau$). Different solutions have been proposed in the literature, often resorting explicit time discretization schemes of higher order, for example multistage Runge-Kutta schemes. However, as we mentioned above such schemes only produce marginal improvements at the expenses of considerably increasing the computations. The Runge-Kutta schemes for example requires multiple evaluations of the full right-hand-side terms on (5). We opt for a different approach, with a better trade off between the enhancement of the stability limit for $\Delta t$ and the growth in computational cost.

Similarly to what is done for the classic LB streaming method, in the steps from Eqs. (2) to (3), a possible improvement consists in taking also for the FV algorithm a semi-implicit integration scheme. However, this is not directly possible for Eq. (5) because of the presence of the advection term. Therefore, we propose to limit such an approach only to the collision and forcing terms, after few manipulations (more details in the Appendix) one gets the discretized form:

\[
\tilde{f}_{\alpha}^{(t+\Delta t)} = \tilde{f}_{\alpha} - \Delta t \frac{S_j}{V} \vec{c}_\alpha \cdot \vec{n}_j \left[ \tilde{f}_{\alpha} + \frac{\Delta t}{2\tau} (\tilde{f}_{\alpha}^{eq} - \tilde{f}_{\alpha}) + \frac{\Delta t}{2\tau} F_{\alpha} \right]_j + \frac{\Delta t}{\tau} (\tilde{f}_{\alpha}^{eq} - \tilde{f}_{\alpha}) + \Delta t \left( 1 - \frac{\Delta t}{2\tau} \right) F_{\alpha}.
\]

The above equation share the same definitions of (3) for the tilted distribution function, $\tilde{f}_{\alpha}$ and the relaxation time ($\tilde{\tau}$). The rule of computing the macroscopic fields and the viscosity $\nu = \tau c_s^2 = (\tilde{\tau} - \Delta t/2)c_s^2$ are exactly the same as for the ST algorithm. Correspondingly, the term $1 - \frac{\Delta t}{2\tau}$ in front of the forcing has been introduced a posteriori to keep the same
hydrodynamic limit. However, one can immediately note the advec
ted field in the equation is not simply a distribution function
but rather a complex term involving also the equilibrium distribution and the for
cing. The main advantage of this approach is that a stability analysis under the same
hypothesis mentioned above (neglecting advection, forcing
time and dependencies in the equilibrium function) shows
now that every time step length $\Delta t$ is stable. However, we
find that the situation reached so far is not yet satisfactory.
From simple numerical tests we observe that even with this
discretization scheme the time-step size is still restricted by the
relaxation time, particularly for small relaxation time values.
The origin of this still limited stability of the scheme lies now in the
advection term. For this reason a further refinement
is proposed. We set it into place by applying the so-called
Heun predictor-corrector scheme to the advection term. In
other words we use the calculation of the population based on
(7), now called $f^*$, as an intermediate value for constructing an
explicit trapezoidal integration rule applied to the advection:

$$
\tilde{f}_a^{(s+1)} = f_a - \frac{\Delta t}{V} S_j \left[ \tilde{f}_a + \frac{\Delta t}{2} \left( \tilde{f}_a - \tilde{f}_a^s \right) + \Delta t \tilde{f}_a^s \right] + \frac{\Delta t}{2} \left[ \tilde{f}_a + \frac{\Delta t}{2} \left( \tilde{f}_a - \tilde{f}_a^s \right) + \frac{\Delta t}{2} \tilde{f}_a^s \right],
$$

(8)

where $\tilde{F}_a^s$ indicates the LB forcing term computed from $\tilde{f}^s$. This scheme enjoys greater stability at the additional
computational price of a second evaluation of the advection
term. In order to make this observations more quantitatie we
should first specify the way in which the flux terms $[\ldots]$ are
computed. Indeed, the exact stability properties of the method
depends upon the implementation of the advection term, that
we discuss in the following section.

3. Approximation of the advection term

There exist several ways to estimate the nonlocal term
$[f_a]$, and each one can be characterised by a spatial order of
accuracy. The complexity of such an estimation also depends
on the grid characteristics. Even for structured but irregular
grids a high-order estimation of $[f_a]$ becomes expensive in
computation terms. In order to simplify such a problem, we
limit the following discussion to the case of structured regular
grids, that is to say, to the case where the nodes lie on lines.
This is the case, for instance, of a nonuniform Cartesian grid
(the typical case of wall refinement), but it also apply to a
uniformly skewed nonorthogonal grids.

It has been long known that fluxes in advection equations
are better approximated by upwind schemes, which are
interpolation schemes biased in the direction determined by
the sign of the characterisitc speeds (the set of $c_\alpha$ in our
case). At the lower order of accuracy, and easiest level of
implementation, there exists the first-order upwind scheme,
increasing the refinement leads to linear interpolation schemes
or even to more refined quadratic schemes (which are of
third order of spatial accuracy). While low-order schemes
introduce artificial numerical dissipations, higher-order ones
lead to spurious oscillations, especially evident near the
boundaries. This is also true in the present cell-centered FV
implementation, in particular zero-order or linear upwind
interpolation schemes leads to inaccurate results. Even a
cell-centered symmetric scheme, which here does not display
extra dissipation, produces inaccurate results in the presence of
boundaries. Empirically, we find that the quadratic upstream
interpolation, known as the QUICK method [47], is the
simplest one to give accurate results both in open (i.e., periodic)
and bounded domains.

According to this approach, on each surface $S_j$ at position,
say $x_j$, $[f_a]$ is approximated via a combination of the value of
$f_a$ in the two nodes bracketing the surface (denoted with $x$ and
$x^\pm$) and a third node that is located upstream with respect to
the direction of the projection of $\hat{c}_\alpha$ on $\hat{n}$ (denoted either $x^+$
or $x^-$). The interpolant function is a parabola $a + b \xi + c \xi^2$,
with $\xi$ the linear coordinate spanning on the line connecting
the nodes [see sketch in Fig. 1(b)]. This leads to:

$$
[f_a(x_j)] = (1 - \gamma_1 - \gamma_2) f_a(x) + \gamma_1 f_a(x^+) - \gamma_2 f_a(x^-) + \left(1 - \gamma_3 - \gamma_4\right) f_a(x^+) + \gamma_3 f_a(x^-) + \gamma_4 f_a(x^+) |_{x^\alpha} : \hat{c}_\alpha \hat{n}_{\alpha} > 0,
$$

(9)

where the $\gamma_{1,2,3,4}$ are four coefficients, that shall be evaluated
and/or stored for each surface of the control volumes.

4. Force term

Finally, a brief remark on the forcing term $F_a$ in the
LB equation. The simplest way to implement it is by the expression:

$$
F_a = w_a \frac{c_\alpha \cdot \rho a}{c^2},
$$

(10)

where the summation over index $\alpha$ is not implied, $w_a$ is
a lattice-dependent weight, and the product $pa$ represents the
force per unit volume in physical space (for example in case of
a gravitational external field $a = g$). The above expression
satisfies the conditions $\Sigma a F_a = 0$ and $\Sigma a c_\alpha F_a = \rho a$, which
are required for Eq. (1) to give the correct macroscopic effect
of a body force term. However, when the body force is time and
space dependent and Eq. (1) is discretized in space and time,
such as in (3), the above expression needs to be refined in order
to remove spurious discretization terms that would otherwise
appear in the macroscopic limit. The corrected expression, first
proposed by Guo et al. [36], is

$$
F_a = w_a \left( \frac{c_\alpha - u}{c^2} + \frac{(c_\alpha \cdot u) c_\alpha}{c^2} \right) \rho a
$$

(11)
with an overall multiplicative factor $1 - \Delta t/(2\tau)$ when the distribution functions $\tilde{f}_a$ are evolved in place of $f_a$. Note that accordingly (by employing the relation between $f_a$ and $\rho u$ and $\rho$ given in Sec. II A) one gets the fluid velocity as $u = \Sigma_a c_a \tilde{f}_a / \Sigma_a \tilde{f}_a + \Delta_t^2 / a$.

5. Boundary conditions

In the following we consider the implementation of two types of boundary conditions (BC): (i) no-slip walls and (ii) fixed density (or equivalently pressure) boundaries. The physical domain boundaries lie on the faces of the external control volumes. Similarly to the bounce-back approach for the streaming LB algorithm, we introduce in-wall ghost cells. However, in the QUICK treatment of the advection two ghost cells are needed instead of one. The ghosts cells are located in wall and have centers at position mirroring the first and second nodes in the fluid domain, as illustrated in Fig. 2. Let us suppose that the quantity to be advected is $f_a$ and that the boundary condition is to be imposed on the $S$ cell surface, whose center is at $x_S$. For simplicity we assume that $S$ lies along the plane $(y,z)$ perpendicular to $x$, with the $x$ axis pointing inward (i.e., in the fluid bulk direction). Consequently, the first two nodes in the fluid domain are located at position $x_1 = x_S + \Delta x_1/2$ and $x_2 = x_S + \Delta x_1 + \Delta x_2/2$, where $\Delta x_1$ and $\Delta x_2$ represents the linear size of the two first discretization volumes. Accordingly, the ghosts cells are at positions $x_{-1} = x_S - \Delta x_1/2$ and $x_{-2} = x_S - \Delta x_1 - \Delta x_2/2$. A no-slip boundary condition requires $u(x_S) = 0$, while the density at $\rho(x_S)$ is free to take any arbitrary value. This corresponds to the constraint $\Sigma_a c_a f_a(x_S) = 0$. The simplest way (but not the only one) to enforce it, is to set the in-wall nodes as the following:

$$f_a(x_{-1}) = f_{inv}(x_1),$$
$$f_a(x_{-2}) = f_{inv}(x_2),$$

where $inv(a)$ is an integer valued function that selects the population moving along the opposite direction with respect to $c_a$. We have verified that such a choice does not introduce artificial fluctuations at the boundary, that would quickly generates instabilities. This implementation of BC, which we dub double reflection, has a first order of accuracy in space (it does not implement the quadratic interpolation) and therefore it leaves room for further improvements.

If instead we are interested to impose a density value at the border, say $\rho_S = \Sigma_a f_a(x_S)$, we need to resort to an extrapolation strategy. We proceed as follows: first, the density value $\rho(x_{-1})$ is linearly extrapolated from the values $\rho_S$ and $\rho(x_1)$, similarly $\rho(x_{-2})$ is derived from $\rho_S$ and $\rho(x_2)$. Second, we assign the in-wall distribution functions as follows:

$$f_a(x_{-1}) = \frac{\rho(x_{-1})}{\rho(x_1)} f_a(x_1),$$
$$f_a(x_{-2}) = \frac{\rho(x_{-2})}{\rho(x_2)} f_a(x_2).$$

Also the above choice, a rescaling of the bulk distribution functions, is not the only viable way for the implementation of fixed density BC, however, it is one that has been shown to not introduce wall disturbances. As a final remark, we shall note that in the implementation of (8) the boundary conditions need not be implemented on the redefined distribution function $\tilde{f}_a$, but rather on the original $f_a = \tilde{f}_a + \Delta_t^2 \Sigma_a a f_a$. The algorithm presentation given so far is independent of the particular microscopic velocity lattice topology. In the present work and for the accuracy study presented in the reminder of this manuscript we make the choice to always use the so-called $D3Q19$ lattice, which is a standard option for three-dimensional LB simulations and reduces to the $D2Q9$ lattice for two-dimensional flow problems [1].

III. ACCURACY TESTS

In this section we address the accuracy of the present LB FV algorithm and we compare it with the ST algorithm. In particular we approach the following questions: (i) To what degree does the FV algorithm correctly describe the dynamics of a low Reynolds number viscous flow? What is its order of spatial accuracy and how does it compare with ST? (ii) Is there any optimal usage of the FV algorithm in order to take advantage of the grid refinement and obtaining highly accurate solutions?

A. Viscosity evaluation

A simulation is performed on a physical domain of size, $[L_x, L_y, L_z] = [1, 64, 1]$. For this test, the number of grid nodes per direction (indicated with $N_{x,y,z}$) is also the same. The flow is initialized with a one-dimensional sinusoidal velocity amplitude profile of the form

$$u(x,y,z) = [u_x(y), u_y, u_z] = \left[\sin\left(\frac{2\pi x}{L_x}\right), 0, 0\right]$$

and it is left to decay in time. We monitor the behavior of the total kinetic energy in time, $k_{tot}(t)$, which is expected to decrease exponentially as $k_{tot}(t) = \frac{1}{4} A^2 L^2 e^{-2\pi^2/\nu t}$, with $\nu$ representing the fluid kinematic viscosity. The reproduced value of $\nu$ can be deducted from a least-square fit of $k_{tot}(t)$ and then compared to the theoretically expected value $\nu = \tau c_s^2 = (\tau - \Delta t/2)c_s^2$. The degree of accuracy of the FV method measured in such a way is compared with the ST method and reported in Fig. 3. While it is known and expected that accuracy carries some form of dependency with the relaxation time $\tau$, we observe that both FV and ST methods reach the maximal accuracy (minimal value of the error denoted $E_\tau$) around $\tau = 0.5$. Such behavior has been already reported in the works of Holdych et al. [48] and Kruger et al. [49], however ST in that very same case performs better by a factor 10. Moreover, in general the ST error grows less than the FV one for all $\tau < 0.5$. 

FIG. 3. Relative error of measured kinematic viscosity $v_{num}$ with respect to the expected one $v_0 = \tau c_s^2$ as a function of the relaxation time $\tau$. In the finite-volume case $\Delta t = 1$ for $\tau \geq 0.13$ (marked with a vertical line) and $\Delta t = 0.1$ for $\tau < 0.13$, while in the Streaming case $\Delta t = 1$ always. In the inset, the absolute value of the same error in log-log scale.

B. Steady Poiseuille flow

Our second test addresses the case of a simple bounded flow in the same spatial domain as above, $[L_x, L_y, L_z] = [1, 64, 1]$. The flow is initiated with a parabolic Poiseuille velocity profile $U_c(y) = 4 U_{max} L_x^2 y (L_y - y)$ corresponding to a Reynolds number $Re = L_y U_{max} / \nu = 10$. A uniform volume forcing along the $x$ direction and no-slip boundary conditions at $y = 0$ and $y = L_y$ positions are imposed, while periodicity is implemented along the horizontal, $x$. The forcing is implemented via a constant acceleration $a_i = \frac{4}{L_y} \frac{\nu}{\tau} U$ using Eq. (11). The simulated flow profile (denoted with $u_i(x, y)$) keeps the original theoretical shape $U_c(x, y)$ with tiny adjustments depending on the method. In order to compare these two functions we use the relative difference $||u_i - U||_2/||U||_2$ where $|| \cdot ||_2$ denotes the $L_2$ norm, which in its discretized form is computed as:

$$||f(x)||_2 = \left(\int_{L} f(x)^2 dx \right)^{1/2} = \left(\sum_{i=1}^{N} f_i^2 \Delta x_i \right)^{1/2}. \tag{15}$$

In Fig. 4, we show the $L_2$ relative difference results at varying the physical domain size in $y$ direction, i.e., changing $L_y$ and at the same time $N_y$ (or in other words keeping fixed the grid spacing $\Delta x \equiv L_x/N_x = 1$). Note that in this test we are actually varying the maximal Mach, $Ma = U_{max}/c_s$, number of the flow. Since the LB equation is $O(Ma^2)$ accurate [1], the figure proves that both the FV and ST methods possess the same level of incompressibility accuracy. However, we can clearly notice that ST in this condition is still on average more accurate by a factor $\sim 8$–$10$ as compared to FV. Such a difference is inherited from the behavior at $\tau = 0.5$ of the viscosity accuracy highlighted in the previous test.

As a further step, we address the effect of a stretched spatial grid on the overall accuracy of the Poiseuille flow simulation. To this end we implement three types of commonly used wall-normal stretched grids. The $y$-coordinate value of the cell volume centres (or simply nodes) is given by

$$y_i = \frac{\xi_i + \xi_{i+1}}{2} \quad \text{with} \quad 0 \leq i < N_y,$$

where the $\xi_i$, the coordinates at the volume boundaries, are defined as Chebychev nodes:

$$\xi_i = \frac{L}{2} \left(1 - \cos \left[\frac{(i - 1/2)\pi}{N}\right]\right) \quad \text{where} \quad 0 \leq i \leq N \tag{16}$$

hyperbolic tangent:

$$\xi_i = \frac{L}{2} \left(1 + \frac{1}{s_1} \tanh \left[\left(\frac{2}{N} i - 1\right) \text{atanh}(s_1)\right]\right) \quad \text{where} \quad 0 \leq i \leq N \tag{17}$$

hyperbolic sine:

$$\xi_i = \left\{\begin{array}{ll}
\frac{L/2}{\sinh(s_2/2)} \sinh \left(\frac{N_i}{N}\right), & \text{if} \quad 0 \leq i \leq \frac{N}{2} \\
L - \left(\frac{L/2}{\sinh(s_2/2)}\right) \sinh \left(\frac{s_2 (N_i - 1)}{N}\right), & \text{if} \quad \frac{N}{2} \leq i \leq N.
\end{array}\right. \quad \tag{18}$$

$L$ and $N$ denote here, respectively, the physical domain size and the grid size (the subscript index $y$ have been dropped for brevity), and $s_1, s_2$ are stretching factors (we have chosen $s_1 = 0.98$ and $s_2 = 6.5$). We then perform the same, $Re = 10$, Poiseuille flow simulation with the above three different grid arrangements with the FV method, and for completeness we also include the results obtained on a uniform grid by both the FV and the ST methods. In order to have a better understanding on the accuracy of the methods, this time we change the number of grid points $(N)$ while keeping fixed the physical domain size $L = 64$. In other words, what we vary here is the average grid spacing $(\Delta x) = L/N$.

Figure 5 reports the results of the described test. Three sources of inaccuracy lead to the overall error behavior observed here. The asymptotic behaviors are respectively
dominated by the spatial accuracy error $E_{\Delta x}$ and by the finite Mach correction $E_{Ma}$, while the transition between these two regimes is also affected by the relaxation time error $E_{\tau}$. At small resolutions (small values of $N$ or equivalently, values of $\Delta x > 1$ in our numerical experiment) the spatial discretization error of ST method goes as $E_{\Delta x} \sim O(\Delta x^2)$ while the one of the uniform-grid FV method behaves roughly as $E_{\Delta x} \sim O(\Delta x^3)$ (due to the QUICK flux computation). In the opposite limit (large $N$, or equivalently $\Delta x < 1$) the compressibility correction comes into play. This effect that goes as $E_{Ma} \sim O(Ma^2)$ has a constant behavior, both in the ST and FV method, because in this numerical experiment $U$ is kept fixed. This explains the observed plateau in the same figure. At the crossover between the two regimes, around the value $\Delta x \simeq 1$ it also happens that $\Delta t \simeq 2\tau$ and this corresponds to the range of best accuracy on the viscosity term (minimal $E_{\tau}$) (same as in Fig. 3). In conclusion, there exists an optimum value of $N$ linked to the relaxation time error $E_{\tau}$ for which the error is minimum; this happens both for the FV and ST algorithms, both on uniform and on stretched grids. The ST method (which can only be based on a uniform grid) performs better than the uniform-grid FV implementation for almost all the values of $N$, furthermore, its absolute accuracy is the highest. However, the situation becomes interesting when the nonuniform grid FV method is employed. There we notice that one can get the same accuracy of the ST algorithm but with a smaller amount of grid points. For instance, in the case of the hyperbolic-tangent grid with $N = 11$ one can obtain the same accuracy as the ST algorithm with $N \simeq 46$. This leads to a saving in memory and potentially in computational costs. In conclusion, the reduction in memory occupation at comparable accuracy seems to be the main benefit one can get from employing the wall-refined FV method rather than the standard ST.

However, a situation that often occurs in the simulation practice is that one wants to use all the available memory of a computer and using an algorithm with the best possible accuracy. The interesting question is then: How can we increase the accuracy at comparable memory costs? Let us imagine one wants to perform again the same Poiseuille simulation at $Re = 10$ but wants to reach a higher level of accuracy (with accuracy defined in the sense of $L_2$ norm). One new possibility is to adjust $L$ and $U_{max}$ in a way that the averaged grid spacing $\langle \Delta x \rangle = L/N$ is kept unchanged, the one that offers the best accuracy performance for a given grid. For the above case of the hyperbolic-tangent grid this would be around $\langle \Delta x \rangle = L/N = 64/11 = 5.8$. The result of this novel Poiseuille flow test is shown in Fig. 6. We can see that when $N = 64$ the best choice is to adopt a grid with tanh or Chebychev spacing and with $\langle \Delta x \rangle$ much larger than unit. The optimum is here reached when $\langle \Delta x \rangle \simeq 20$, this produce an increase in accuracy of a factor greater than 100 compared to the case of a simulation with the ST algorithm (and this independently of the value of $\langle \Delta x \rangle$ chosen for the ST method).

IV. PERFORMANCE EVALUATION

From a computational point of view the FV algorithm has more operations per time step than the ST algorithm. This comes from the fact that while the streaming process can be implemented simply as a shift in memory the computation of the flux term in FV involves many arithmetic operations. According to our measurement the present FV algorithm is about 8–10 times computationally more expensive than ST algorithm per time step. However, as discussed in Sec. II B 2, differently from the ST algorithm, in the FV the time step $\Delta t$ is a function of $\tau$. The functional relation linking the maximum time step to $\tau$ for the proposed time discretizations can be measured and it is reported in Fig. 7. We observe that the method based on (8), semi-implicit integration in time of the collision term plus a trapezoidal correction for the advection, is
FIG. 7. Maximum allowed time step in the decaying Kolmogorov flow by using Eq. (6) (FV Euler), Eq. (7) (FV semi-implicit), and Eq. (8) (FV semi-implicit + Heun). The very same result is obtained in the steady Poiseuille flow at Re = 10. The horizontal dashed line represents the standard choice of the time step for the ST implementation, i.e., \( \Delta t = 1 \) independently of \( \tau \).

superior to the others. In particular, for this method \( \Delta t_{\text{max}} > 1 \) for \( \tau > 1/8 \), that is to say that the time step can be larger than the one used in the ST method (which is bounded to the value 1 for \( \Delta x = 1 \)). The most advantageous case occurs for \( \tau \sim 1/2 \)—which as we have shown above is also the best condition for accuracy—in that case \( \Delta t_{\text{max}} \sim 1.7 \). This reduces the ratio of the computational cost FV/ST to a factor 5–6. We note that all this reasoning did not take into account the effect of nonuniform grids. As we have seen for the simple Poiseuille flow this brings further saving in terms of computational costs as compared to the ST method.

Finally, we shall mention that memory occupation is also part of the performance of an algorithm: According to our estimate FV in the present formulation needs twice more memory allocation as compared to ST.

V. BENCHMARK IN A COMPLEX FLOW: HIGH RAYLEIGH NUMBER THERMAL CONVECTION

Several LB finite-volume methods proposed in the past have been tested just on laminar flows as proof of principle of the proposed algorithms. Few exceptions exist in the literature in which the FV method have been benchmarked on much more complex, three-dimensional, developed turbulent flows. One of such exceptions is the model proposed by Amati et al. [37], which was probed in a three-dimensional plane turbulent channel flow. In such a case, however, the grid wall refinement was based on a very simple structure of halved-grid spacing near the walls and the accuracy of the method turned out to be not satisfactory (the computed mean-velocity profile could not properly reproduce the log-law of the wall).

In this section the proposed lattice Boltzmann FV algorithm is tested to simulate a complex three-dimensional statistically steady turbulent flow. Our choice is here for the well-studied flow in the Rayleigh-Bénard (RB) cell, the prototype of a thermal convection driven system [50]. The RB setup considered in this study deals with a cubic domain (of height \( H \) and equal lateral sizes \( L \)); it has periodic BC on the lateral walls, while on the horizontal walls no-slip and isothermal conditions are imposed. In this system, the fluid is heated from below and as such (when the heating is large enough and a small perturbation is introduced in the system) an instability arises and brings the system into convective condition. The dimensionless control parameters are the Rayleigh (Ra) and Prandtl (Pr) numbers and aspect ratio \( AR = L/H \) [50].

For the LB simulation we use a double population approach [18]. This means that beside Eq. (1) we integrate an analogous equation for the distributions \( g_\alpha \):

\[
\frac{\partial g_\alpha}{\partial t} + c_\alpha \cdot \nabla g_\alpha = \frac{1}{\tau_\alpha \tilde{g}_\alpha} (g_\alpha^{eq} - g_\alpha) \quad \alpha = 0, \ldots, N_{\text{pop}}
\]

with equilibrium function \( g_\alpha^{eq} = (T/\rho) \tilde{f}_\alpha \) where the macroscopic temperature is computed as \( T = \sum_\alpha g_\alpha \) and the thermal diffusivity corresponds to \( \kappa = \tau_\alpha c_\alpha^2 \). Furthermore, in the equation for \( f_\alpha \) the forcing term \( F_\alpha \) is assigned in order to model the buoyancy force as represented in the Boussinesq approximation. In physical space the added buoyant acceleration has the form \( a = -\beta(T - T_0)g \) where \( \beta \) is the volume thermal expansion coefficient and \( T_0 \) is a reference temperature here as the mean temperature between the ones at the top and bottom plates.

In order to validate the double population approach also for the FV method, we first address a rather elementary simulation in steady convective laminar conditions, adapting it from a test case already conducted for the ST algorithm in Ref. [18]. The system is two-dimensional (2D) with control parameters fixed at \( Ra = 10^4 \), \( Pr = 1 \), and \( AR = 2.02 \). The fluid is initially at rest \( (u = 0) \), while the temperature field is initialized by a linear conductive profile, \( T_\alpha(z) = -\Delta T(z/H + 1/2) \), plus a small perturbation [of order \( O(10^{-2}) \Delta T \)] breaking the left-right symmetry. Given the weak, but not negligible, compressibility of the simulated flow the initial density stratification due to gravity should be also taken into account. This avoids the generation of pressure waves at the startup of the simulation. We do it via the barometric equation, which leads to \( \rho(z) = \rho_0 \exp\left[ -c_\beta^2 g \int_0^z T_\alpha(z')dz' \right] \), where \( \rho_0 \) is a reference density value taken at temperature \( T_0 \). Note that in a 2D system, in order not to suppress the linear hydrodynamic instability, the cell aspect ratio (AR) must be slightly larger than \( 2\pi/k_c \) (where \( k_c = 3.117 \) is the wave vector of the most unstable linear mode) [18]. Indeed, when \( AR = 2.02 \) the initial perturbation produce an immediate kinetic energy growth and a steady convective flow pattern establishes. The dimensionless heat flux [or Nusselt number (Nu)] goes from the conductive unit value up to around \( Nu \simeq 2.66 \), see Ref. [51]. In Fig. 8 we report the results of simulations conducted with the two LB algorithms. We can observe [Fig. 8(a)] that the temporal dynamics of the dimensionless global heat flux, i.e., the Nusselt number \( Nu(t) \), is identical for the two simulations, furthermore they both agree with the analytical asymptotic value given by Clever and Busse [51]. A grid convergence study, performed by increasing the number of grid points of the same factor in each Cartesian direction, shows that the absolute error on Nusselt number as compared to the Clever and Busse value reaches the second decimal digit [inset of
by the control parameters value \( Ra = 10^4 \), \( Pr = 1 \), and \( AR = 2.02 \). (a) Temporal dynamics of dimensionless global heat flux \([Nusselt number Nu(t)]\) as a function of time, in dissipative time units \( t_D = H^2/\kappa \). The Nu steady state value is compared to a value linearly interpolated from Clever and Busse calculations \([51]\). In the inset, the grid convergence study displaying the absolute error of the measured Nusselt number \((Nu)\) with respect to Clever and Busse calculations \([51]\). In the inset, the number \( Nu = 104, Pr = 0.1, AR = 1 \), and \( H = 3 \). The control parameter value \( Ra = 10^4 \), \( Pr = 1 \), and \( AR = 2 \) (see also Ref. \([53]\)). In this condition the 3D system dynamics is already highly chaotic (or moderately turbulent). In Ref. \([52]\) the authors employed a direct numerical simulation based on a staggered finite-difference discretization of the Navier-Stokes–Boussinesq equation system. The grid they adopted has size \((N_x,N_y,N_z) = (128,128,64)\), it is uniform in the horizontal directions and has a sinh-type refinement [the same as in \((16)\)] in the vertical direction. Our benchmark is as follows: we perform two series of simulations, one with the ST method and the other with the FV approach, the dimensionless parameters for the two cases are the same as the ones of Kunnen et al., as well as the number of grid points per direction. However, while the ST uses a uniform grid in the FV case we use exactly the same grid as the one adopted in the finite-difference simulation \([52]\). Table I reports the numerical values of the parameters adopted for the two LB simulations. Note that the large-scale velocity \( U \), which is roughly proportional to the so-called free-fall velocity, i.e., \( U \sim \sqrt{\beta g \Delta T H} \), is the same in both simulations. It is a good practice in LB simulations to always keep control of the large-scale velocity in order to prevent it to take too large values: it is worth reminding that in order to reproduce the incompressible fluid dynamics the condition \( U \ll 1 \) is required (a commonly accepted rule of thumb in LB practice is \( U \sim 0.1 \)). In order to reach a good convergence of the statistical observables in the system the RB simulations are carried on for a total time \((t_{tot})\), which spans over several large-eddy turnover times \((T)\). We estimate that \( t_{tot} \approx 12 \ T \) for both FV and ST simulations, with \( T \) computed from the zero-crossing time value of the autocorrelation function of the total kinetic energy.

In Figs. 10 and 11 we show a comparison of the vertical mean temperature profile \((T_m)\) (averaged over horizontal planes and time) and of the vertical root-mean-square

![FIG. 9. Illustration of the three-dimensional Rayleigh-Bénard system.](image)

![FIG. 8. Comparison of streaming (ST) and finite-volume (FV) LB algorithms in a simulation of the Rayleigh-Bénard system in steady convective state. The system is two-dimensional, and characterised by the control parameters value \( Ra = 10^4 \), \( Pr = 1 \), and \( AR = 2.02 \). (a) Temporal dynamics of dimensionless global heat flux \([Nusselt number Nu(t)]\) as a function of time, in dissipative time units \( t_D = H^2/\kappa \). The Nu steady state value is compared to a value linearly interpolated from Clever and Busse calculations \([51]\). In the inset, the grid convergence study displaying the absolute error of the measured Nusselt number \((Nu)\) with respect to Clever and Busse \((Nu_{ith})\) vs. number of grid points in the \( y \) direction of a 2D grid. (b) Comparison of temperature isolines in the asymptotic steady state. Levels are taken at values \( T_n = T_0 \pm n \ \Delta T /8 \), with \( n = 0,1,2,3 \).](image)

TABLE I. Parameter values used for the RB simulations at \( Ra = 2.5 \times 10^6 \), \( Pr = 1 \), and \( AR = 2 \) for a three-dimensional (3D) simulation of a RB system (Fig. 9) characterized by: \( Ra = 2.5 \times 10^6 \), \( Pr = 1 \), and \( AR = 2 \) (see also Ref. \([53]\)). In this condition the 3D system dynamics is already highly chaotic (or moderately turbulent). In Ref. \([52]\) the authors employed a direct numerical simulation based on a staggered finite-difference discretization of the Navier-Stokes–Boussinesq equation system. The grid they adopted has size \((N_x,N_y,N_z) = (128,128,64)\), it is uniform in the horizontal directions and has a sinh-type refinement [the same as in \((16)\)] in the vertical direction. Our benchmark is as follows: we perform two series of simulations, one with the ST method and the other with the FV approach, the dimensionless parameters for the two cases are the same as the ones of Kunnen et al., as well as the number of grid points per direction. However, while the ST uses a uniform grid in the FV case we use exactly the same grid as the one adopted in the finite-difference simulation \([52]\). Table I reports the numerical values of the parameters adopted for the two LB simulations. Note that the large-scale velocity \( U \), which is roughly proportional to the so-called free-fall velocity, i.e., \( U \sim \sqrt{\beta g \Delta T H} \), is the same in both simulations. It is a good practice in LB simulations to always keep control of the large-scale velocity in order to prevent it to take too large values: it is worth reminding that in order to reproduce the incompressible fluid dynamics the condition \( U \ll 1 \) is required (a commonly accepted rule of thumb in LB practice is \( U \sim 0.1 \)). In order to reach a good convergence of the statistical observables in the system the RB simulations are carried on for a total time \((t_{tot})\), which spans over several large-eddy turnover times \((T)\). We estimate that \( t_{tot} \approx 12 \ T \) for both FV and ST simulations, with \( T \) computed from the zero-crossing time value of the autocorrelation function of the total kinetic energy.

Figs. 10 and 11 we show a comparison of the vertical mean temperature profile \((T_m)\) (averaged over horizontal planes and time) and of the vertical root-mean-square
temperature ($T_{rms}$) profile, which are defined as follows:

$$T_m(z) = \frac{1}{t_{tot} L^2} \int_0^{t_{tot}} \int_0^L \int_0^L T(x,y,z,t) \, dx \, dy \, dt$$

$$T_{rms}(z) = \left( \frac{1}{t_{tot} L^2} \int_0^{t_{tot}} \int_0^L \int_0^L [T(x,y,z,t) - T_m(z)]^2 \, dx \, dy \, dt \right)^{1/2}.$$  

We find good agreement among all the three types of simulations. Furthermore, we observe that when the thickness of the boundary layer $\lambda_T$ is defined by the so-called slope definition, $\lambda_T \equiv \Delta T (2 \partial T_m(z)/\partial z = 0)$ (see Fig. 10) both the Kunnen et al. data and the FV ones have about ten points in the thermal boundary layer (BL), while the ST despite its remarkable agreement with the other methods, has only three points in the BL. Small systematic differences can be seen on the vertical profile of the mean (turbulent) kinetic energy, $k_m(z) = \tau_{tot}^{-1} L^{-2} \int_0^{t_{tot}} \int_0^L \int_0^L \epsilon u^2 \, dx \, dy \, dt$, reported in Fig. 12. $k_m(z)$ has a slower rate of convergence than the temperature variance, this is the reason why small residual statistical discrepancies remain present here despite of the large number of turnover times of the simulation. In order to appreciate more sensible differences between the FV and ST simulations, one has to address either observables involving temperature-velocity correlation or small-scale quantities, which are more sensitive to the spatial resolution of the mesh, particularly close to the walls. For this reason in Fig. 13 we compare the time-averaged quota-dependent Nusselt number $Nu(z) = \kappa \rho \partial \langle T \rangle / \langle \kappa U \partial H \rangle$ (where for short $(\ldots)$ denotes time and space horizontal averages) and the so-called Bolgiano length $L_B(z) = (\beta g)^{-3/2} \langle \epsilon_u \rangle^{5/4} \langle \epsilon_T \rangle^{-3/4}$ (where $\epsilon_u$ and $\epsilon_T$ are respectively the velocity and temperature dissipation rates) [54]. As far as the Nusselt number is concerned, despite a very close mean value, we see important differences at the wall. This is due to the combined effect of the boundary conditions and the gradient computations in postprocessing the data. The temperature gradient computations involved a second-order central finite difference scheme at the bulk nodes and a second-order forward-backward finite difference scheme at the boundary nodes. These schemes have been applied to both FV and ST algorithms. The FV method exhibits wall oscillations, which are a factor 10 smaller than the ones seen for the ST method, making more reliable the total heat flux estimate. Furthermore, in the $L_B(z)$ measure we observe near a 50% discrepancy at the wall and a smaller but non-negligible difference in the bulk of the cell. Clearly a wall-clustered grid is needed to resolve observables built on sharp temperature and velocity gradients.

We now would like to address the matter of determining which LB method is computationally more convenient. The choice to have the same $U$ in the FV and ST simulations has an implication on the determination of the large-eddy turnover time and therefore the total number of time steps.
needed to perform a simulation of equivalent physical time span. The reasoning is as follows: the large turnover time goes as \( T \sim H/U \) therefore on the total number of time steps \( M \) for a simulation that should span a time \( T \) scale as \( M \sim H/\Delta t \). It follows that the FV simulation will need in this case a number of time steps larger by a factor 10/4 as compared to the ST one (see Table I). Since the FV is more expensive than ST by a factor 8–10 per time step, we get that the added computational cost of the FV method is up to \( \simeq 25 \) larger than the ST method. However, such an increase in computational cost shall be properly weighted by the enhancement in the spatial resolution due to the wall-stretched grid. A univocal guideline is not available in this context. A commonly employed criterion in the numerics of bounded flows is to count the number of grid nodes in the BL (another, although less restrictive, rule would be to take into account the distance of the first collocation point from the wall). Here, if we adopt such a criterion the ratio is in favor of the FV method over ST by a factor 10/3, that means that we need approximately 3–4 more nodes in the ST simulation. However, if we want to also keep the same aspect ratio of the simulation domain, and since ST is bounded to cubic grids, such an increase of the resolution shall be applied to every Cartesian direction, which makes the FV grid advantage greater by a factor of \((10/3)^3 \simeq 37\). In conclusion, in a ST RB simulation we need 37 times more computation nodes to perform a simulation with comparable resolution of the FV method. By combining the above estimates, we see that a simulation of same physical time span and same boundary layer resolution is about \(1 - 25/37 \sim 33\% \) less expensive for the FV method than the ST one. In summary, even if the cost per unit physical time in a FV simulation is higher than ST, when a criterion for the minimal spatial resolution (particularly near walls or obstacles and in a three-dimensional geometry) is chosen, the FV method becomes advantageous.

Finally, we have performed RB simulations at increasingly higher Ra numbers (\(Ra = 10^6, 10^9\)). All these simulations have around ten grid points in the thermal BL as shown in Fig. 14. No numerical instabilities were noticeable as Ra was increased, demonstrating that the FV algorithm can deal with turbulence at high Rayleigh number conditions.

**VI. CONCLUSIONS**

In this paper we have presented a finite-volume algorithm for the lattice Boltzmann equation. The method has been validated through a systematic comparison with the standard streaming LB approach, by means of test case simulations in laminar as well as in unsteady and turbulent flows with heat transfer. The tests have shown that the FV has the same order of incompressibility accuracy as the ST algorithm and an improved spatial accuracy (third compared to second order). It has, however, a much more elevated computational cost that we estimate to be around 8–10 times per time step. One notable advantage of the FV method is the possibility to adopt stretched...
rectilinear grids, which makes it suitable for the simulation of turbulent bounded flows. Taking this into account, i.e., taking into consideration (for instance) the minimal number of collocation points required in a boundary layer for a proper simulation, the FV algorithm surpasses the ST method.

A number of improvements can still be made on the proposed algorithm. First, the boundary conditions can be improved. We have noticed that in strongly sheared flows, such as channel flow turbulence, some spurious oscillations at the boundaries can destabilize the simulations. Second, in stretched Cartesian grids the number of interpolant coefficients to be stored is considerably more limited than for other cases of structured grids. In the former case the treatment of advection can be further optimized compared for other cases of structured grids. In the former case the boundary conditions can be improved. We have noticed that in strongly sheared flows, such as channel flow turbulence, some spurious oscillations at the boundaries can destabilize the simulations. Second, in stretched Cartesian grids the number of interpolant coefficients to be stored is considerably more limited than for other cases of structured grids. In the former case the treatment of advection can be further optimized compared for other cases of structured grids.

The FV discretization proposed in this work builds on the standard streaming algorithm, as a consequence the two algorithms do not differ much. The major difference is of course the way in which the advection is computed. However, for sharing the RB simulation results. E.C. thanks Federico Toschi for useful discussions.

APPENDIX: MIXED SEMI-IMPLICIT FINITE-VOLUME LB SCHEME

In the following we briefly show how to derive the discretized form (7) from the finite-volume LB equation (5). First, the discretization in time is applied by adopting a mixed approach: while for the collision and forcing terms a semi-implicit method is used, for the advection a simple explicit Euler is implemented. This leads to the form:

\[ f^{(t+\Delta t)}_\alpha = f^{(t)}_\alpha - \Delta t \frac{S}{V} c_\alpha \cdot n_j \left[f^{(t)}_\alpha \right] + \frac{\Delta t}{2} \left( f^{eq(t)}_\alpha - f^{(t)}_\alpha + F^{(t)}_\alpha + \frac{1}{\tau} \left( f^{eq(t+\Delta t)}_\alpha - f^{(t+\Delta t)}_\alpha \right) + F^{(t+\Delta t)}_\alpha \right) \]  

(A1)

Note that by bringing on the left-hand side all the terms to be evaluated at \( (t + \Delta t) \):

\[ f^{(t+\Delta t)}_\alpha = \frac{\Delta t}{2\tau} \left( f^{eq(t+\Delta t)}_\alpha - f^{(t+\Delta t)}_\alpha + \tau F^{(t+\Delta t)}_\alpha \right) = f^{(t)}_\alpha - \Delta t \frac{S}{V} c_\alpha \cdot n_j \left[f^{(t)}_\alpha \right] + \frac{\Delta t}{2\tau} \left( f^{eq(t)}_\alpha - f^{(t)}_\alpha + \tau F^{(t)}_\alpha \right). \]  

(A2)

One now introduces the redefined distribution function \( \tilde{f}_\alpha = f_\alpha - \frac{\Delta t}{\tau} (f^{eq} - f_\alpha + \tau F_\alpha) \). It shall be noted that, at equilibrium condition, from the above definition we get \( f^{eq}_\alpha = f^{eq}_\alpha - \frac{\Delta t}{\tau} F_\alpha \), and therefore the relation can be easily inverted leading to

\[ f_\alpha = \tilde{f}_\alpha + \frac{\Delta t}{2\tau} (f^{eq} - \tilde{f}_\alpha) + \frac{\Delta t}{2} F_\alpha. \]  

(A3)

At this point the new relaxation time is introduced \( \bar{\tau} = \tau + \frac{\Delta t}{\tau} \), and the equation can be written as

\[ \tilde{f}^{(t+\Delta t)}_\alpha = \tilde{f}_\alpha - \Delta t \frac{S}{V} c_\alpha \cdot n_j \left[ \tilde{f}_\alpha + \frac{\Delta t}{2\bar{\tau}} (f^{eq}_\alpha - \tilde{f}_\alpha) + \frac{\Delta t}{2} F_\alpha \right] + \frac{\Delta t}{\bar{\tau}} (f^{eq}_\alpha - \tilde{f}_\alpha) + \Delta t F_\alpha. \]  

(A4)

The above equation coincides with (7), once the correction factor \( 1 - \frac{\Delta t}{\bar{\tau}} \) is applied to the force intensity \( F_\alpha \) [36].


