Multigrid Solution of the Steady-State Lattice Boltzmann Equation

Dimitri J. Mavriplis

Department of Mechanical Engineering
University of Wyoming
Laramie, WY 82071
USA

Abstract

Efficient solution strategies for the steady-state Latttice Bolztmann equation are investigated. Stable iterative methods for the linearized lattice Boltzmann equation are formulated based on the linearization of the lattice Boltzmann time-stepping procedure. These are applied as relaxation methods within a linear multigrid scheme, which itself is used to drive a Newton solver for the full non-linear problem. Although the linear multigrid strategy provides rapid convergence, the cost of a linear residual evaluation is found to be substantially higher than the cost of evaluating the non-linear residual directly. Therefore, a non-linear multigrid approach is adopted, which makes use of the non-linear LBE time-stepping scheme on each grid level. Rapid convergence to steady-state is achieved by the non-linear algorithm, resulting in one or more orders of magnitude increase in solution efficiency over the LBE time-integration approach. Grid-independent convergence rates are demonstrated, although degradation with increasing Reynolds number is observed, as in the case of the original LBE time-stepping scheme. The multigrid solver is implemented in a modular fashion by calling an existing LBE time-stepping routine, and delivers the identical steady-state solution as the original LBE time-stepping approach.

Key words: Lattice Boltzmann; Multigrid

1 Introduction

The last decade has seen rapid progress in the theoretical understanding, algorithmic development, and use of lattice Boltzmann methods [1, 2]. As opposed to traditional computational fluid dynamics approaches, which compute macroscopic fluid dynamic properties by discretizing the continuum governing equations, the independent variables in a lattice Boltzmann approach consist

of particle distribution functions in phase space, from which macroscopic continuum fluid properties can be recovered. LBE methods are thus based on kinetic theory, but are truly mesoscopic methods, since the variables are neither based on micro (molecular) scales, or macro (continuum) scales. This results in computational approaches which are both tractable for macroscopic predictions, and well-suited for problems involving non-simple physical phenomena such as interfacial dynamics and multi-phase flows, the effects of which are more easily incorporated at the mesoscale level [3, 4].

Historically, LBE methods have been derived from Lattice Gas Automata (LGA) methods [5]. LGA methods model macroscopic fluid motion through the evolution of a set of discrete particles on a regular lattice (cartesian grid). Many of the deficiencies of LGA methods in reproducing accurate macroscopic behavior were originally resolved by the LBE method [6, 7], by neglecting individual particle motion and adopting a particle distribution function approach.

More recently, LBE methods have also been shown to correspond to a finite difference discretization of the Boltzmann equation, which converges to the continuous Navier-Stokes equations, in the limit of incompressibility [8, 9]. While various discretizations of the Boltzmann equation are possible (see [10, 11] for example), it is important to note that LBE represents a specific discretization which is not necessarily intuitive, without knowledge of the LGA origins of the scheme. Nevertheless, this interpretation enables the analysis of the LBE scheme in terms of accuracy, stability, and convergence, using traditional numerical analysis techniques.

LBE schemes for incompressible flow simulations have been shown to be first-order accurate in time and second-order accurate in space (using the optimum value of the relaxation parameter) [12]. Due to particular aspects of their specific discretization, LBE methods achieve remarkable spatial accuracy and temporal efficiency in spite of these common asymptotic estimates [13]. These discretizations contain several desirable properties, such as linear convection terms and nearest-neighbor stencils. On a regular cartesian grid, LBE methods can be implemented in a two-step procedure: a collision operator evaluation which involves only local operations (at a grid point), and an advection operation, where values are shifted in memory to adjacent grid points without performing any computation, thus achieving very high computational rates on common computer hardware.

In spite of these advantages, LBE methods represent explicit time-integration approaches, which are notoriously inefficient for steady-state simulations or time-dependent problems with large separations in relevant time and spatial scales. For a steady-state problem with N unknowns, explicit methods generally require O(N**2) operations to attain the steady-state, while optimal solvers can be devised which require O(N) operations. Due to the asymp-

totic nature of these estimates, it should be clear that LBE methods in their present form will always become non-competitive for large enough problems, regardless of the efficiency of their implementation.

The objective of this work is the development of asymptotically efficient solution strategies for the LBE approach for steady-state problems. In previous work, implicit solution methods [14] and multigrid approaches [11] have been devised for the steady-state discrete Boltzmann equation. However, the spatial discretizations used in [11] are not identical with the LBE approach, thus compromising the compact stencil, proven accuracy, and efficient implementation of the original LBE discretization. In this work, we seek an efficient solution method to the exact LBE discretization in order to preserve these properties, and to demonstrate the possibility of efficiently solving these equations, which have their origins in the LGA world, using standard numerical techniques. In this manner, we guarantee that the steady-state solution, and thus spatial accuracy of our result is identical to that of the original LBE approach. Another important objective is the preservation of the simplicity and efficiency of implementation of LBE, and the ensuing parallel efficiency.

2 LBE Discretization

The continuous Boltzmann equation can be written as

$$\frac{\partial f}{\partial t} + \xi . \nabla f = \Omega(f) \tag{1}$$

where $f = f(\mathbf{x}, \xi, t)$ is the particle distribution in phase space, ξ is the molecular velocity, and $\Omega(f)$ represents the collision integral. This equation must be discretized in velocity space, physical space, and time. In two dimensions, a simple discretization into 9 velocity components, which makes use of the following discrete velocity set:

$$\xi_a = c\,\mathbf{e}_a, \qquad a = 0, 1, ..., 8 \tag{2}$$

with

$$\begin{aligned} &\mathbf{e}_0 = (+0, +0)^T \\ &\mathbf{e}_1 = (+1, +0)^T\, \mathbf{e}_2 = (+0, +1)^T\, \mathbf{e}_3 = (-1, +0)^T\, \mathbf{e}_4 = (+0, -1)^T \\ &\mathbf{e}_5 = (+1, +1)^T\, \mathbf{e}_6 = (-1, +1)^T\, \mathbf{e}_7 = (-1, -1)^T\, \mathbf{e}_8 = (+1, -1)^T \end{aligned} \tag{3}$$

has been shown to be sufficient for reproducing accurate macroscopic fluid dynamics [15, 8]. In the above equations, c represents a magnitude of molecular

velocity, and the $\mathbf{e}_a$ are the direction vectors of the discrete velocities. Using this velocity discretization, we obtain nine distribution functions $f_a(\mathbf{x}, t), a = 0, ..., 8$, one for each component of the discrete velocity set. The macroscopic fluid variables are obtained as moments of these distribution functions :

$$\rho = \sum_a f_a, \qquad \rho\mathbf{u} = \sum_a f_a \xi_a \tag{4}$$

where ρ is the fluid density, and $\mathbf{u}$ is the fluid macroscopic velocity vector.

We will restrict our study to single time relaxation (SRT) BGK-type models for the collision term [16], resulting in the following form of the Boltzmann equation:

$$\frac{\partial f}{\partial t} + \xi.\nabla f = -\frac{1}{\lambda}\left[f - f^{eq}\right] \tag{5}$$

where λ is a relaxation time, and f^{eq} is the equilibrium distribution, which depends on f through the macroscopic fluid variables ρ and $\mathbf{u}$. The relaxation time is related to the physical viscosity ν as:

$$\nu = \frac{2\lambda - \Delta t}{6} c^2 \tag{6}$$

A standard form for the equilibrium distribution function using the 9-component velocity discretization described above is given by the so-called D2Q9-model [15, 8]:

$$f_a{}^{eq} = w_a \rho \left[1 + \frac{3\mathbf{e}_a.\mathbf{u}}{c^2} + \frac{(9\mathbf{e}_a.\mathbf{u})^2}{2c^4} - \frac{3\mathbf{u}^2}{2c^2}\right] \tag{7}$$

with $w_0 = 4/9$, $w_1 = w_2 = w_3 = w_4 = 1/9$, $w_5 = w_6 = w_7 = w_8 = 1/36$. For our purposes, it is sufficient to note that $f_a{}^{eq}$ evaluated at a particular spatial location (grid point) is a non linear function of the f_a values at that same location, which follows from equations (4) and (7).

The lattice Boltzmann equation (LBE), or in this case the LBGK method, is obtained by discretizing equation (5) as:

$$f_a(\mathbf{x} + \xi_a \Delta t, t + \Delta t) - f_a(\mathbf{x}, t) = -\frac{\Delta t}{\lambda}\left[f_a(\mathbf{x}, t) - f_a{}^{eq}(\mathbf{x}, t)\right] \tag{8}$$

An important feature of the LBE approach is the use of a cartesian grid with constant spacing h, and a time-step size defined as

$$\Delta t = \frac{h}{c} \tag{9}$$

which corresponds to a CFL number of unity, based on the molecular velocity c. This choice results in advection of the f_a values by exactly one grid spacing at each time step, resulting in the following discrete equations at a grid point x_i, y_i:

$$f_a{}^{n+1}(x_i + he_{a_1}, y_i + he_{a_2}) - f_a{}^n(x_i, y_i) = -\frac{1}{\tau}\left[f_a{}^n(x_i, y_i) - f_a{}^{eq^n}(x_i, y_i)\right] \tag{10}$$

where $n+1$ and n denote the new and old time levels, respectively, and e_{a_1} and e_{a_2} refer to the first and second components of the $\mathbf{e}_a$ vector defined in equations (3). For example, for $a = 1$, the first term in the above equation corresponds to the value $f_1(x_i + h, y_i)$, i.e. the value of f_1 at the neighboring grid point in the positive direction along the x-axis. From equations (6) and (9), the relaxation parameter $\tau = \frac{\lambda}{\Delta t}$ is given as:

$$\tau = \frac{3\nu}{ch} + \frac{1}{2} \tag{11}$$

Hence, τ varies linearly with the fluid viscosity and must be greater than $1/2$, which corresponds to the limiting case of inviscid flow.

The LBGK time-stepping scheme of equation (10) can be implemented in a two-step procedure as:

$$f_a{}^*(x_i, y_i) = f_a(x_i, y_i) - \frac{1}{\tau}\left[f_a{}^n(x_i, y_i) - f_a{}^{eq^n}(x_i, y_i)\right] : Collision \tag{12}$$

$$f_a{}^{n+1}(x_i + he_{a_1}, y_i + he_{a_2}) = f_a{}^*((x_i, y_i) : Advection \tag{13}$$

The first (collision) step in entirely local, involving only values at the current grid point, while the second (advection or streaming) operation corresponds to a shift of the solution variables in memory without any floating point operations. Note that the intermediate values $f_a{}^*$ need not be stored separately, but can be overwritten temporarily on the f_a values, using a single field solution array. These aspects result in a very efficient time-stepping strategy in terms of both time per iteration and required memory.

At steady-state, the time-level superscripts may be dropped and we obtain:

$$f_a(x_i + he_{a_1}, y_i + he_{a_2}) - f_a(x_i, y_i) = -\frac{1}{\tau}\left[f_a(x_i, y_i) - f_a{}^{eq}(x_i, y_i)\right] \tag{14}$$

which represents a set of non-linear spatially coupled equations. Alternatively, we may re-write this in residual form as:

$$
\begin{aligned}
R_{a_i}(f) = & \\
& -(1-\frac{1}{\tau}) f_a(x_i, y_i) - \frac{1}{\tau} {f_a}^{eq}(x_i, y_i) + f_a(x_i + he_{a_1}, y_i + he_{a_2}) \\
& = 0
\end{aligned} \tag{15}
$$

These equations may be solved using Newton's method, by first linearizing the residual and inverting the resulting linear system at each Newton iteration:

$$
\left[\frac{\partial R(f)}{\partial f}\right] \Delta f = -R(f) \tag{16}
$$

to obtain the update $\Delta f = f^{new} - f$. In practice, this approach works well, provided a suitable linear solver can be found for the solution of equation (16). The Jacobian matrix $\frac{\partial R(f)}{\partial f}$ has a block matrix structure, with 9x9 dense block sub-matrices on the diagonal corresponding to the coupling of the 9 discrete f_a values at each grid point. On the other hand, the non-zero off-diagonal blocks which correspond to coupling between neighboring grid points are sparse and contain a single non-zero entry equal to 1 on the sub-matrix diagonal, corresponding to the linearization of the first term in equation (14).

Using a Jacobi iterative approach, equation (16) may be solved iteratively as:

$$
[D]_i \, \Delta f_i = -R_{a_i}(f) - \Delta f_a(x_i + he_{a_1}, y_i + he_{a_2}) \tag{17}
$$

where $[D]_i$ is the block diagonal corresponding to the point (x_i, y_i) of the full Jacobian matrix and is given as:

$$
\left[-(1-\frac{1}{\tau})\,[I] - \frac{1}{\tau}\left(\frac{\partial {f_a}^{eq}(x_i, y_i)}{\partial f_b(x_i, y_i)}\right)\right] \tag{18}
$$

where $[I]$ is the 9x9 identity matrix, and the last term corresponds to the linearization of the collision term at a point, yielding a 9x9 matrix for $a = 0, ...8$ and $b = 0, ...8$. If the Jacobian matrix $\frac{\partial R(f)}{\partial f}$ is diagonally dominant (in the block sense), then this approach will result in a convergent algorithm. However, it has been found that this approach is convergent only for low values of τ, i.e. high values of fluid viscosity or low Reynolds numbers. In general, the matrix $\frac{\partial R(f)}{\partial f}$ is not diagonally dominant for τ values of interest, and the method diverges. The reason for this difficulty can be seen by examining the terms in the Jacobian matrix. The contribution to the diagonal elements in the Jacobian from the convective term (second term on left of equation (14)) acts to reduce the magnitude of the $-(1-\frac{1}{\tau})$ term in equation (18), since we require $\tau > \frac{1}{2}$.

At this point, it is useful to contrast this situation with the strategy reported

in [11]. In this work, the LBE discretization of equation (14) is replaced with the following discretization (for the first-order scheme reported in [11]):

$$f_a(x_i, y_i) - f_a(x_i - he_{a_1}, y_i - he_{a_2}) = -\frac{1}{\tau}\left[f_a(x_i, y_i) - f_a{}^{eq}(x_i, y_i)\right] \qquad (19)$$

which is obtained by applying a first-order upwind finite-difference scheme to the convective term in equation (5), and dropping the time-dependent term for the steady-state case. Note that in this case, the diagonal term of the corresponding Jacobian matrix is now enhanced (positive coefficient) by the contribution of the first term on the left-hand side of the above equation (i.e. $f_a(x_i, y_i)$), as opposed to the previous case where it is the second term on the left-hand side of equation (14) which contributes to the Jacobian diagonal with a negative coefficient, thus reducing diagonal dominance. Therefore, the discretization of equation (19) can be expected to be more stable for the Jacobi iteration scheme. Efficient solutions of this discretization using non-linear Gauss-Seidel and non-linear multigrid schemes have been demonstrated in [14, 11].

In the current work, our objective is the efficient solution of the original LBE discretization given by equation (14). However, as noted above, a straight-forward application of a point-implicit Jacobi iteration to equation (14) or (15) is most often unstable. In order to devise a stable iterative scheme, we re-examine the original LBE time-integration strategy given by equation (10). The linear stability of this procedure has been examined in the literature [17]. The explicit time-integration scheme of LBE is found to be linearly stable for a wide range of parameters (i.e. $\tau > \frac{1}{2}$ with bounds on the maximum fluid velocity $\mathbf{u}$). Thus, we can surmise that an iterative solution strategy for equation (14), which emulates the explicit time-integration scheme of the LBE approach in pseudo-time, should be stable whenever the LBE scheme itself is linearly stable. The linear iteration equivalent of the LBE time-stepping procedure applied to equation (15) is given as:

$$\Delta f_a^{k+1}(x_i + he_{a_1}, y_i + he_{a_2}) =$$
$$-R_{a_i}(f) + (1 - \frac{1}{\tau})\Delta f_a^k(x_i, y_i) + \frac{1}{\tau}\left[\frac{\partial f_a}{\partial f_b}\right]_i \Delta f_b^k(x_i, y_i) \qquad (20)$$

At first glance, this procedure would appear to be less effective than the iteration strategy given by equation (17), since the collision term is treated implicitly in equation (17), while it is treated explicitly in this approach. On the other hand, this approach requires no matrix inversion, since the unknowns at $(x_i + he_{a_1}, y_i + he_{a_2})$ are uncoupled. The correspondence of this scheme to the LBE time-stepping scheme implies stability when LBE itself is stable. In order to understand why this approach is more stable than the iteration

strategy given by equation (17), we re-write the steady-state LBE residual of equation (15), using a shift in the grid point index i, resulting in:

$$\begin{aligned} R_{a_i}(f) = & \\ f_a(x_i, y_i) - (1 - \frac{1}{\tau}) f_a(x_i - he_{a_1}, y_i - he_{a_2}) - \frac{1}{\tau} f_a{}^{eq}(x_i - he_{a_1}, y_i - he_{a_2}) & \\ = 0 & \end{aligned} \quad (21)$$

The Jacobian resulting from the linearization of this residual now contains dense 9x9 block matrices at off-diagonal locations, and identity matrices along the diagonal. This is akin to swapping the off-diagonal and diagonal blocks of the Jacobian defined by equation (15), resulting in a more diagonally dominant system. Another way of expressing this result, is that equation (15) should be considered as the definition of the residual at $(x_i + he_{a_1}, y_i + he_{a_2})$, rather than at (x_i, y_i). For a system of linear equations, this corresponds to permuting the rows of the matrix prior to inversion of the matrix, in order to achieve stronger diagonal dominance, a standard technique known as pivoting for the solution of linear systems [18]. Thus, in spite of the fact that the majority of the stencil entries and all the non-linearity in equation (14) reside at off-diagonal locations in equation (20), this approach provides a more stable iteration strategy than that described by equation (17). Re-writing equation (20) as an update for the point (x_i, y_i), we obtain the final form of the linear Jacobi iteration:

$$\begin{aligned} \Delta f_a^{k+1}(x_i, y_i) = & - R_{a_i}(f) \\ & + (1 - \frac{1}{\tau}) \Delta f_a^k(x_i - he_{a_1}, y_i - he_{a_2}) \\ & + \frac{1}{\tau} \left[\frac{\partial f_a}{\partial f_b} \right]_i \Delta f_b^k(x_i - he_{a_1}, y_i - he_{a_2}) \end{aligned} \quad (22)$$

where $R_{a_i}(f)$ is the non-linear residual defined by equation (21).

In spite of this revelation, nothing has been gained so far, since the iterative scheme based on equation (22) simply corresponds to a linearized explicit LBE time-stepping procedure, which can be expected to converge at the same rate as the original LBE procedure. However, having recast this procedure as an iterative strategy in pseudo-time, we may abandon time-accuracy in the interest of speed of convergence. The simplest example of this approach is to replace the Jacobi iteration with a Gauss-Seidel iteration in equation (22), relaxing the grid point values in a prescribed order, using latest available values from neighboring stencil points, thus neglecting time accuracy. Over-relaxed Gauss-Seidel (SOR) strategies may also be attempted. Another approach is to replace the Jacobi iteration with an under-relaxed Jacobi iteration, a strategy that will prove beneficial in the context of the multigrid solver discussed below. Other possible strategies include the use of (block) tridiagonal line

solvers applied along horizontal or vertical lines in a line-Jacobi or line-Gauss-Seidel strategy. Finally, any of these schemes may be used as the driver for a linear multigrid scheme, using coarse grid levels to accelerate the solution of equations (21) discretized on the fine grid.

3 Linear Iteration Solution of Steady-State LBE

As an example test case, we consider the solution of the driven cavity problem using the LBE discretization. The solution of this test problem using the LBGK discretization has been reported previously in the literature [19], and we omit any discussion on the accuracy of the steady-state solution using LBGK, since our approach delivers exactly the same steady-state solution as the original LBGK time-integration procedure. A qualitative illustration of the solution of the driven cavity problem using LBGK on a 129 x 129 cartesian grid at a Reynolds number of 100 is given in Figure 1.

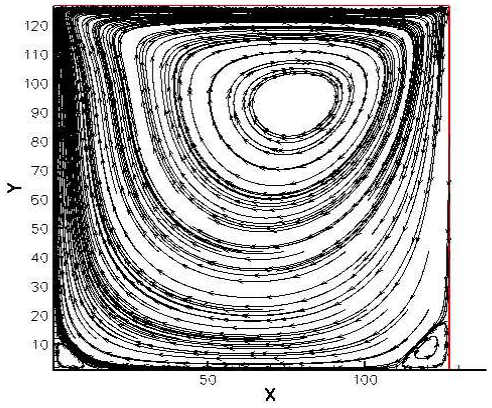

Fig. 1. Streamlines of steady-state driven cavity solution using LBE method on 129 x 129 grid for Reynolds number of 100.

Although we will examine the convergence of this problem on various grid sizes in subsequent sections, the computations discussed in this section were performed on a 33 x 33 grid. For the linearized approaches, convergence is monitored by observing the reduction in the L_2 norm of the linear residuals defined as:

$$RMS = \frac{1}{N}\sqrt{\sum_i \sum_a {r_{a_i}}^2} \tag{23}$$

where N is the number of grid points, and the linear residual r_{a_i} is defined from equation (22) as:

$$r_{a_i}(f) = R_{a_i}(f) - (1 - \frac{1}{\tau})\Delta f_a(x_i - he_{a_1}, y_i - he_{a_2}) \\ - \frac{1}{\tau}\left[\frac{\partial f_a}{\partial f_b}\right]_i \Delta f_b(x_i - he_{a_1}, y_i - he_{a_2}) + \Delta f_a(x_i, y_i) \qquad (24)$$

$R_{a_i}(f)$ being the non-linear residual, previously defined by equation (21).

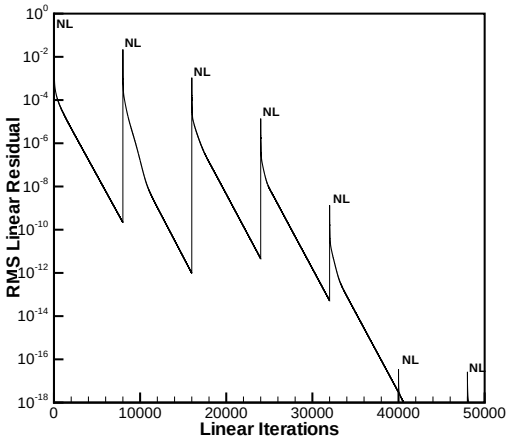

Fig. 2. Convergence of linearized steady-state LBE equation using linear Jacobi iteration as a driver for outer non-linear Newton iteration for driven cavity problem on 33 x 33 grid at Reynolds number of 100. NL symbols denote initiation of new non-linear Newton iteration, and demonstrate quadratic convergence of non-linear system.

Figure 2 depicts the convergence for the driven cavity problem on a 33 x 33 grid using a linear Jacobi iteration to solve the linear problem at each stage within an outer Newton iteration. A linear residual reduction of approximately 9 orders of magnitude is achieved using 8000 Jacobi cycles between non-linear updates. By solving the linear system to this high degree of completion, the Newton scheme achieves its characteristic quadratic convergence, and the non-linear problem is solved in six Newton iterations, as evidenced by the jumps in linear residuals falling below machine round-off error after the fifth non-linear update. The convergence of the linear Jacobi scheme at a given non-linear cycle is compared with the convergence of the non-linear LBE time-stepping scheme in Figure 3, using the L_2 norm of the non-linear residuals as the measure of convergence for the LBE time-stepping scheme. Both schemes achieve nearly identical asymptotic convergence rates (per cycle), as expected, since the Jacobi scheme represents the analogue of the LBE time-integration scheme for the linearized problem. The only differences appear in the initial

convergence history, where non-linearities are still important. When a Gauss-Seidel strategy is employed, sweeping sequentially along all grid points from left to right, beginning with the bottom grid line and proceeding to the top grid line, the convergence rate is roughly doubled versus the Jacobi schemes, as is well known theoretically for simple problems such as the Poisson equation [20]. Furthermore, using an over-relaxed Gauss-Seidel method (successive over-relaxation or SOR) with a relaxation factor of 1.2 (determined empirically as the maximum stable value), a further improvement in convergence rate is observed. Although these schemes can be used to deliver faster linear system convergence, a more optimal strategy is to use these iteration methods as smoothers within a multigrid scheme as described below.

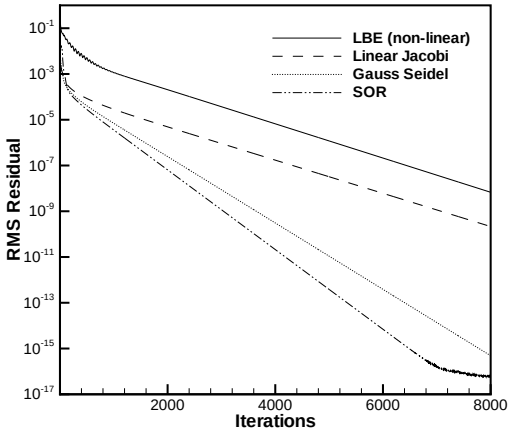

Fig. 3. Linear system convergence for Jacobi, Gauss-Seidel and over-relaxed Gauss-Seidel at a given non-linear iteration compared with the convergence of the non-linear LBE time-integration approach, for driven cavity problem on 33 x 33 grid at a Reynolds number of 100.

4 Linear Multigrid Solver

A truly optimal (O(N)) solver can be constructed using a linear multigrid approach driven by the local linear iteration techniques discussed above, on each grid level. The idea of a multigrid solver is to accelerate the solution of a fine grid discretization by computing corrections to this discrete problem on coarser grid levels [21, 22, 23]. If the original fine grid linear problem is given as:

$$\mathbf{L}_h u = g \tag{25}$$

the corresponding correction equation is given as:

$$\mathbf{L}_h \Delta u_h = g - \mathbf{L}_h u = -r_h \tag{26}$$

assuming a linear operator $\mathbf{L}_h$, and denoting the residual of the linear system on the fine grid by r_h. This correction equation is transfered to the coarse grid as:

$$\mathbf{L}_H \Delta u_H = -I_h^H r_h \tag{27}$$

where the subscripts h and H refer to the fine and coarse grids, respectively. The operator I_h^H denotes the restriction operator, which transfers residuals from the fine grid to the coarse grid. Equation (27) is solved on the coarse grid, and the resulting corrections are transfered back to the fine grid as:

$$u_h^{n+1} = u_h^n + I_H^h \Delta u_H \tag{28}$$

where I_H^h denotes the prolongation operator which transfers corrections from coarse to fine grid. In this work, the prolongation operator is taken as a bilinear interpolation operator in space between coarse and fine grid point locations [22]. In cases where the discrete equations are written in integral form, the restriction operator is most often taken as the transpose of the prolongation operator. In the present finite-difference approach, for consistency the transposed prologation operator must be rescaled by the factor $\frac{1}{4}$, such that each row of coefficents in the resulting restriction operator sum to unity. For the linearized LBE equation, the fine grid correction equation may be written as:

$$\left[\frac{\partial R(f)}{\partial f}\right]_h (\Delta f^{k+1} - \Delta f^k) = -R(f) - \left[\frac{\partial R(f)}{\partial f}\right]_h \Delta f^k \tag{29}$$

where k denotes the linear iteration number. The right hand side of the above equation corresponds to the negative of the linear residual (c.f. equation (24)), and $\left[\frac{\partial R(f)}{\partial f}\right]_h$ is the (block) matrix which produces the three last terms on the right-hand side of equation (24) when multiplied by the vector Δf.

In order to formulate the coarse grid correction equation for this problem, a form of the operator $\left[\frac{\partial R(f)}{\partial f}\right]_H$ on the coarse grid which is consistent with the fine grid discretization must be devised. Most often this is achieved by simply rediscretizing the original governing equations on the coarse grid. However, in the case of the LBE equations, the coarse grid discretization must be performed with a rescaled value of the parameter τ in order to remain consistent with the fine grid problem. This is evident from equation (11), which indicates that a constant value of τ leads to an increasing value of viscosity ν on coarser

grid levels, and thus inconsistent fine and coarse grid physical problems. However, the coarse grid levels in a multigrid algorithm operate on a correction equation, rather than a discretized physical problem. As such, it is misleading to assume that the same physical problem (i.e. viscosity) should be solved on each grid level. A more rigorous approach is to contruct the coarse grid operator by Galerkin projection [22, 23], or more simply to note that the effective time-step of a single LBE iteration on a coarse grid is twice the effective time step used on the fine grid (assuming the coarse grid spacing is twice the fine grid spacing in all diretions). Using these arguments, the consistent value of τ for the coarse grid level is found to be $1/2$ the value of τ on the fine grid. However, for a multigrid sequence involving a complete set of sequentially coarser grid levels, the value of τ must be reduced on each successively coarser level, which leads to instability of the coarse grid iteration procedure when $\tau < 1/2$ is attained at some grid level (c.f. equation (11)).

In fact, because the convergence rate of the LBE scheme (and thus our linearized Jacobi iteration) degrades with smaller τ values, we prefer to devise a multigrid scheme which employs the same value of τ on all grid levels. The inconsistency arising from the use of a constant value of τ on coarse on fine levels can be remedied by rescaling the corrections generated by the coarse grid. Because using the same value of τ on the coarse and fine grids is equivalent to using half the desired time-step on the coarse level, the resulting corrections, which correspond to the integrated changes in time of the solution values, must be rescaled by the factor 2 upon prolongation back to the fine grid level. Alternatively, this rescaling may be achieved by multiplying the restricted residuals by the factor 2, and leaving the prolongation operator unchanged. An alternate interpretation of this scaling can be obtained by considering the full collision term on the coarse level to be comprised of two equal parts, one part which is time-stepped on the coarse level, and a second part which is restricted from the fine level, and is thus held frozen on the coarse level during the time-integration procedure.

The two level multigrid system described above is replaced by a multigrid system where each coarse grid is solved recursively with the multigrid algorithm, using a complete set of sequentially coarser grid levels. On each grid level, Jacobi iterations are employed to approximately solve or smooth the correction equations. The function of this iterative process is to annihilate high-frequency errors on each grid level [21, 22]. Thus, the chosen iterative methods must have good high-frequency damping properties. Figure 4 illustrates the convergence obtained for a 5 level multigrid algorithm using a W-cycle strategy [21] with 4 linear Jacobi iterations on each grid level. Plotting the convergence as a function of the number of iterations performed on the fine grid (i.e. 4 fine grid iterations per multigrid cycle), the linear Jacobi multigrid scheme is seen to provide almost no benefit over the single grid scheme. This is due to the poor high-frequency damping properties of the Jacobi scheme. Point Jacobi

iterations are well known to have poor high-frequency damping properties for the Poisson equation, while an under-relaxed Jacobi iteration offers much better smoothing properties. For this reason, we employ an under-relaxed Jacobi iteration on each grid level, using a relaxation factor of 0.8. The under-relaxed iteration is written as a two step approach:

$$\begin{aligned}\Delta f_a^*(x_i, y_i) = & - R_{a_i}(f) \\ & + (1 - \frac{1}{\tau})\Delta f_a^k(x_i - he_{a_1}, y_i - he_{a_2}) \\ & + \frac{1}{\tau}\left[\frac{\partial f_a}{\partial f_b}\right]_i \Delta f_b^k(x_i - he_{a_1}, y_i - he_{a_2}) \qquad (30)\end{aligned}$$

$$\Delta f_a^{k+1}(x_i, y_i) = \quad \gamma \Delta f_a^*(x_i, y_i) + (1 - \gamma)\Delta f_a^k(x_i, y_i)) \qquad (31)$$

where γ is the relaxation factor, taken as 0.8. On each level, 4 under-relaxed linear Jacobi iterations are employed using a W-cycle multigrid strategy.

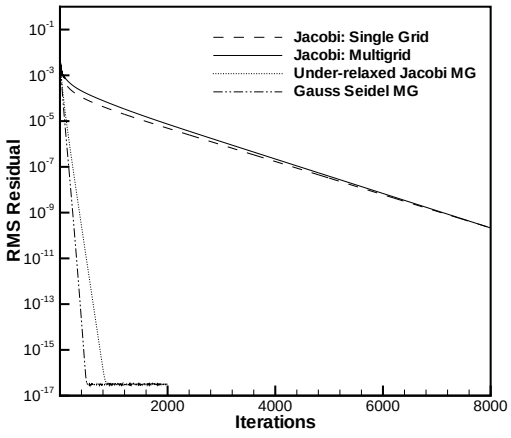

Fig. 4. Convergence of various linear multigrid algorithms at a given Newton iteration for the driven cavity problem on 33 x 33 grid, as compared with the convergence of the single grid linear Jacobi iteration scheme.

This small change can be seen to dramatically improve the multigrid convergence rate in Figure 4, illustrating the importance of good smoothing properties in a multigrid algorithm. Similarly, using the linear Gauss-Seidel iteration as a multigrid smoother (using 4 iterations on each grid level with a W-cycle), results in even faster convergence, achieving machine zero in slightly over 100 multigrid cycles (400 fine grid relaxations).

In Figure 5, the linear multigrid scheme using a Gauss-Seidel smoother is used as the driver in a Newton scheme to solve the full non-linear problem. In this case, 150 multigrid cycles are employed to converge the linear system at each

non-linear cycle, which is sufficient to drive the linear residuals to machine zero. The non-linear system converges quadratically in 6 steps, as reported previously in Figure 2, while the solution of the linear system at each stage has been accelerated by an order of magnitude compared to that reported in Figure 2, where a single-grid Jacobi linear iteration was used.

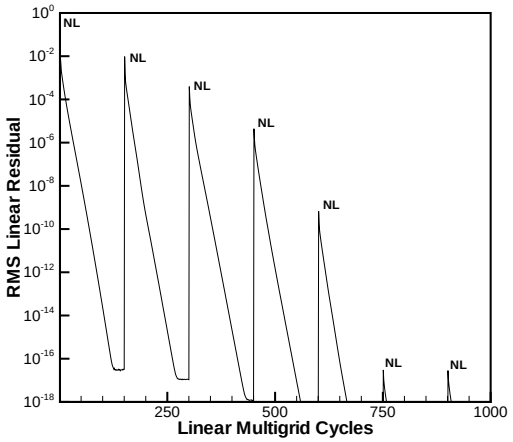

Fig. 5. Convergence of linearized steady-state LBE using Gauss-Seidel driven linear multigrid algorithm as a driver for outer non-linear Newton iteration for driven cavity problem on 33 x 33 grid. NL symbols denote initiation of new non-linear Newton iteration, and demonstrate quadratic convergence of non-linear system.

5 Non-Linear Multigrid Approach

While the above results demonstrate rapid linear system convergence using the multigrid approach for solving the linearized steady-state LBE equations, the overall cost in terms of time to solution remains larger than that delivered by the simple LBE time-stepping strategy for moderate size grids. On the one hand, this is due to the overly tight convergence criterion on the linear system at each non-linear iteration. A more efficient procedure would estimate the optimum level of linear system convergence required to efficiently drive the non-linear Newton iteration scheme. However, more importantly, the cost of a single linear iteration is found to be substantially higher than the cost of a non-linear LBGK explicit time step, as shown by the number of floating point operations required by each scheme in Table 1. The cost of the linear scheme is dominated by the 9 x 9 matrix vector product. Note that even the simple under-relaxation scheme given by equation (31) requires 27 operations (3 per f variable, assuming $1-\gamma$ is precomputed), which becomes a substantial fraction of the number of floating point operations for the entire non-linear

LBE time-step. The added cost of the linear relaxation scheme over the non-linear approach is even larger than inferred by a simple comparison of floating point operations, due to the larger and more complex memory access pattern involved in the linear discretization stencil.

Table 1
Number of floating point operations required per grid point by various components of linear and non-linear LBE iterations.

Non-Linear LBE Time-Step	Linear LBE Jacobi Relaxation
Computation of Macro. Variables: 18	9 x 9 Matrix Vector Mult: 146
Collision (eq (12)) : 75	Other terms (eq (30)): 36
Advection (eq (13)) : 0	Under-relaxation (eq (31)): 27
Total : 93	Total : 209

Comparisons between linear and non-linear solver performance have been investigated previously by the author for finite-element and finite-volume type discretizations [24, 25]. In the asymptotic limit, non-linear iterations and their consistently linearized linear iteration counterparts should converge at identical rates [25]. However, there can be large differences in the cost of a non-linear iteration versus its linearized counterpart. While the cost of the non-linear iteration scheme is dominated by the cost of evaluating the non-linear functions, the cost of the linear iteration is solely dependent on the sparsity pattern of the stencil, since these are evaluated as matrix-vector multiplications (excluding the cost of forming the linearization itself, which can be amortized over a number of linear iterations). In [25, 24], the cost of a linear iteration was found to be substantially lower than the cost of the corresponding non-linear iteration, for finite-element discretizations of radiation diffusion problems. In the current work, the situation is reversed, with the cost of a non-linear LBE time-step being substantially lower than the cost of the equivalent linearized iteration. This is due in part to the efficient two-step approach (collision followed by advection) for evaluating the non-linear LBE time-step, versus the rather complex stencil produced by the linearization procedure, where the $\frac{\partial f^{eq}}{\partial f}$ matrix represents a dense 9 x 9 matrix at each grid point, which must be multiplied by the solution vector at each iteration. Additionally, the amount of storage required by the linearized iteration strategy is substantially higher than that of the non-linear LBE time-step, since the latter requires only a single copy of the solution variables f (in addition to the macroscopic variables ρ and $\mathbf{u}$), which are progressively overwritten during the collision and advection operations.

Because the low storage requirements and efficiency of the LBE time-integration approach are important assets, an optimal steady-state LBE solver must at-

tempt to retain these attributes. Thus, the use of a non-linear multigrid strategy, driven by a non-linear iteration scheme on each grid level, offers the potential for substantial gains over the linearized approach described previously.

In order to design such a solver, we construct the equivalent non-linear scheme of the linearized schemes described above. For the Jacobi iteration, this corresponds to using the explicit LBE time-integration scheme on each grid level, which then is modified to enable under-relaxation using the following three-step procedure:

$$f_a{}^*(x_i, y_i) = f_a(x_i, y_i) - \frac{1}{\tau}\left[f_a{}^n(x_i, y_i) - f_a{}^{eqn}(x_i, y_i)\right] : Collision \tag{32}$$

$$f_a{}^{**}(x_i + he_{a_1}, y_i + he_{a_2}) = f_a{}^*((x_i, y_i) : Advection \tag{33}$$

$$f_a{}^{n+1} = \gamma f_a{}^{**} + (1-\gamma) f_a{}^n : Relaxation \tag{34}$$

where γ is the relaxation factor (usually taken as 0.8). The spatial coordinates have been dropped in the description of the relaxation step, since it is understood that this is a local operation performed at all grid points. The first two-steps are identical to the original LBE time-stepping sequence, and these are implemented by calling an existing LBE subroutine. The additional under-relaxation step requires the extra storage for the old solution values (or the non-linear residual), which is not required in the LBE or non-relaxed Jacobi iteration, where the solution values are continuously overwritten.

The Full Approximation Storage (FAS) non-linear multigrid method [21, 22, 23] can be used to solve the non-linear problem directly. If the non-linear problem to be solved on the fine grid is written as:

$$\mathbf{L}_h u = g \tag{35}$$

and the non-linear residual is defined as:

$$R_h(u) = \mathbf{L}_h u - g \tag{36}$$

the coarse grid correction equation is given by:

$$\mathbf{L}_H u_H - \mathbf{L}_H \hat{I}_h^H u_h = -I_h^H R_h(u_h) \tag{37}$$

which can be re-written as:

$$\mathbf{L}_H u_H = \mathbf{L}_H \hat{I}_h^H u_h - I_h^H R_h(u_h) \tag{38}$$

where the term on the right-hand side is known as the defect-correction. The solution of this equation yields the coarse grid variables u_H, which are then used to update the fine grid variables as:

$$u_h^{n+1} = u_h^n + I_H^h \left(u_H - \hat{I}_h^H u_h^n \right) \tag{39}$$

where h and H denote the fine and coarse grids, respectively. The restriction and prolongation operators I_h^H and I_H^h are identical to those described for the linear multigrid method. In addition, for the non-linear multigrid method, restriction of the solution variables to the coarse grid through the operator $\hat{I}_h^H$ is required. This operator is taken as pointwise injection i.e. the values at coarse grid points are prescribed the values from the fine grid points with which they coincide.

For the LBE discretization, the steady-state problem on the fine grid is given by:

$$R_h(f) = 0 \tag{40}$$

where the residual is computed as:

$$R_h(f) = \\ f(x_i, y_i) - (1 - \frac{1}{\tau}) f(x_i - he_{a_1}, y_i - he_{a_2}) - \frac{1}{\tau} f^{eq}(x_i - he_{a_1}, y_i - he_{a_2}) \tag{41}$$

The corresponding FAS multigrid coarse grid equation can be written as:

$$R_H(f_H) = D_H \\ D_H = R_H(\hat{I}_h^H f_h) - 2 I_h^H R_h(f_h) \tag{42}$$

where D_H represents the defect correction, which is constant throughout a multigrid iteration, and where the coarse grid residual $R_H(f_H)$ is evaluated analogously to equation (41), and the factor 2 is employed in the restricted residuals to account for the different scaling of τ in the coarse and fine grid problems, as described previously in the linear multigrid section. The coarse grid equation is solved using an under-relaxed Jacobi iteration, in a three-step procedure, similar to that described by equation (32), but modified to incorporate the defect correction as:

$$f_a{}^*(x_i, y_i) = f_a(x_i, y_i) - \frac{1}{\tau} \left[f_a{}^n(x_i, y_i) - f_a{}^{eqn}(x_i, y_i) \right] : Collision \tag{43}$$

$$f_a{}^{**}(x_i + he_{a_1}, y_i + he_{a_2}) = f_a{}^*((x_i, y_i) : Advection \tag{44}$$

$$f_a{}^{n+1} = \gamma(f_a{}^{**} + D_H) + (1 - \gamma) f_a{}^n : Relaxation \tag{45}$$

As for the fine grid, the first two steps of the coarse grid iteration are identical to the time-integration of the LBE equation, and are implemented by calling an LBE subroutine, providing a modular setting for applying multigrid to existing LBE codes. On the coarse grids, two additional arrays must be allocated for storing the defect correction (i.e. the last term in equation (42), and the restricted solution variables $\hat{I}_h^H f_h$). In summary, the FAS multigrid scheme requires 2 copies of the solution vector on the fine grid, and four copies on the coarser levels. However, since each coarser grid is 4 times smaller than the next finer level (8 times in 3D), the total memory count for the multigrid solver is less than three times that required for the original single grid LBE solver. The cost of a multigrid W-cycle using 4 iterations per grid level in the coarsening phase, and 4 iterations in the refinement phase, is approximately 24 times the cost of a single fine grid explicit LBE time-step. Since the multigrid cycle performs a total of 8 fine grid iterations (4 initial and 4 final iterations), the multigrid overhead per fine grid iteration is roughly a factor of 3. In two dimensions, the complexity of each coarser grid level is a factor of 4 lower than the next finer level, giving the W-multigrid cycle a theoretical complexity of 2. Our observed overhead (factor of 3) is 50% larger than this complexity argument, and can be attributed to the extra work required by the under-relaxation stage (c.f. equation (45)) and the inter-grid transfer operations. Because of the inherent efficiency of the LBE time-integration procedure, even relatively simple additional operations can substantially increase the overall cost per cycle, and it is imperative to keep all additional operations to a minimum.

Figure 6 illustrates the convergence of the original LBE explicit time-integration approach for the driven cavity problem versus the non-linear multigrid approach for various grid sizes ranging from 33 x 33 up to 513 x 513 in terms of number of iterations, while the same results are displayed in terms of normalized work units in Figure 7. A normalized work unit is defined as the CPU time required for a single LBE time-step on the subject grid. As expected, the convergence of the original LBE time-stepping approach degrades as the grid is refined, while the non-linear multigrid approach shows relatively grid-independent convergence rates, achieving convergence to machine zero in 100 to 150 multigrid cycles. In fact, the multigrid convergence rates are seen to improve slightly for finer grids. This is attributed to the use of a constant Reynolds number of 100 for all cases, which results in a lower cell Reynolds number as the grid is refined. Similar multigrid behavior was noted in [11].

For a Reynolds number of 100, the multigrid algorithm delivers an approximate efficiency gain of 5 over the LBE time-integration approach for the 33 x 33 grid, which increases to a factor of 30 for the 129 x 129 grid and reaches approximately 100 for the 513 x 513 grid. The 513 x 513 driven cavity problem can be converged to machine zero on a 2.8GHz Pentium PC in about 5 minutes of cpu time using 100 Mbytes of memory in double precision.

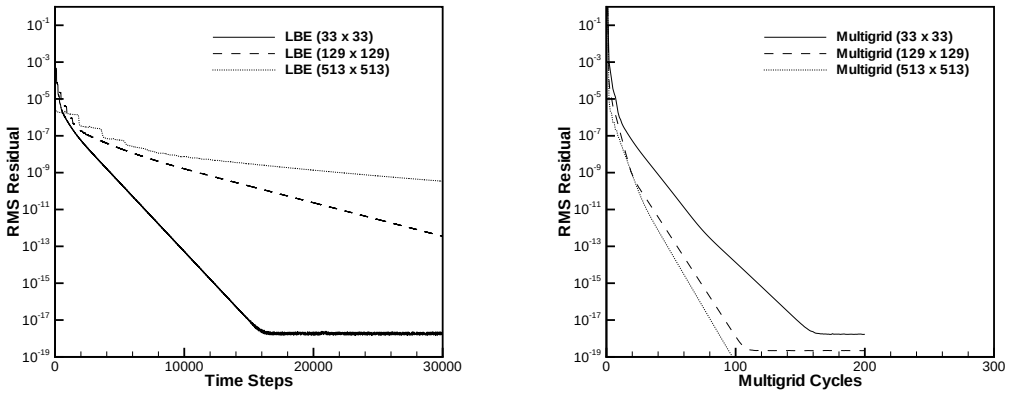

Fig. 6. Convergence of time-integration scheme (on left) and non-linear multigrid scheme (on right) for driven cavity problem at Reynolds number 100 for various grid sizes as a function of the number of iterations. Note different horizontal axis scales of both figures.

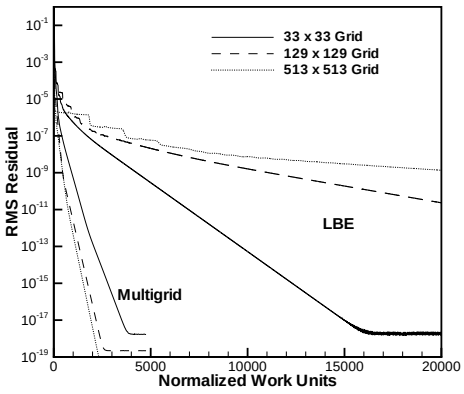

Fig. 7. Comparison of convergence of non-linear multigrid scheme and LBE time-integration scheme for driven cavity problem at Reynolds number 100 for various grid sizes as a function of normalized work units

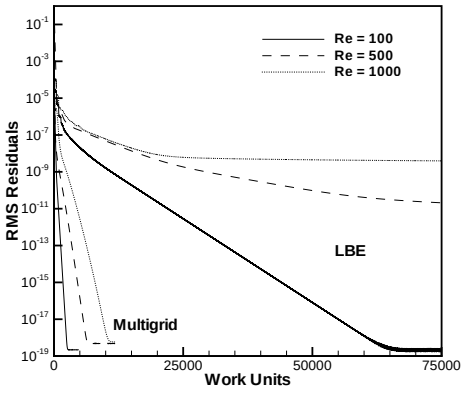

Fig. 8. Comparison of convergence of LBE time-integration scheme and non-linear multigrid scheme in terms of normalized work units for driven cavity problem on 129 x 129 grid for various Reynolds numbers.

The final solution using the multigrid scheme is identical to that obtained using the original LBE scheme. Figure 8 compares the convergence of the LBE time-integration scheme and the non-linear multigrid scheme for different Reynolds numbers on a fixed grid size of 129 x 129. As is well known, the convergence of the LBE scheme decreases as the Reynolds number is increased, eventually becoming unstable for large enough values of the Reynolds number. Since the

non-linear multigrid algorithm relies on the LBE time-stepping scheme as a smoother on each grid level, it is not un-expected that the performance of the multigrid scheme degrades as well with increasing Reynolds number, as seen from the figure. However, for this case, the degradation in the multigrid convergence rates are not as severe as those experienced by the LBE time-integration method. Nevertheless, increasing the Reynolds number from 100 to 1000 requires almost four times the number of multigrid cycles to reduce the residuals to machine zero. Further improvement in the dependence of the multigrid convergence rate on Reynolds number will require the design of a more efficient smoother for use on each grid level of the multigrid sequence.

6 Non-Linear Multi-Color Gauss-Seidel Smoother

Improved smoothers can be devised by replacing the original LBE time-step, which corresponds to a non-linear Jacobi iteration, by a Gauss-Seidel procedure, simply by altering the order of evaluation of the updates as described in the linear solution approaches. Rather than implementing a global mesh sweeping strategy, we prefer the use of multi-color (i.e. Red-Black) Gauss Seidel strategies, in order to preserve the favorable parallel computation characteristics of the original LBE approach. The 9-point nature of the LBE stencil dictates the use of a 4 color Gauss-Seidel iteration strategy, as depicted in Figure 9. In a single pass over all four color groups, all variables at all grid points are updated, using only neighboring information from distinct colors or (sub)-iteration levels.

Figure 10 compares the multigrid convergence of the multi-color Gauss-Seidel smoother versus that of the under-relaxed Jacobi smoother for two different Reynolds numbers on a 129 x 129 grid. The Gauss-Seidel smoother demonstrates faster convergence in terms of multigrid iterations for both cases, providing a slightly larger benefit for the higher Reynolds number case. In addition to producing faster convergence per cycle, the Gauss-Seidel smoother requires fewer floating point operations, since under-relaxation is not required, and the third step in equation (45) may be omitted. Even though this step contains relatively few operations, its omission is noticeable on total run-time due to the low number of operations in the LBE time-integration scheme. On the other hand, the Gauss-Seidel smoother requires re-ordering of the LBE time-stepping updates, which may adversely impact cache locality, and which also complicates the task of producing a modular multigrid solver which can call existing LBE time-integration routines.

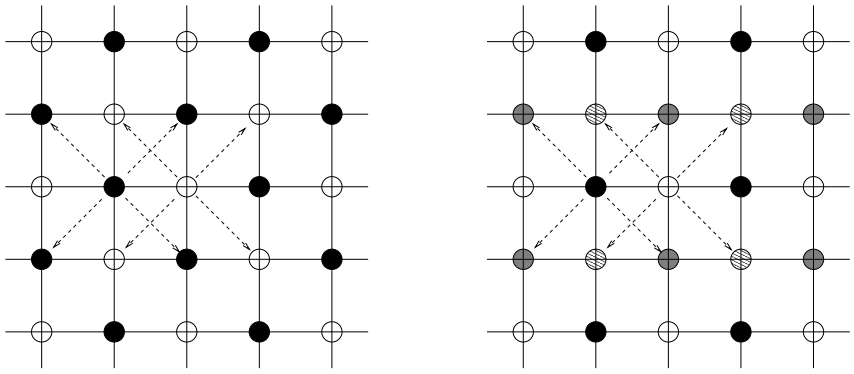

Fig. 9. Traditional Red-Black Gauss-Seidel scheme (left) results in diagonal stencil connections of LBE scheme having same color. Right figure illustrates four-color Gauss-Seidel scheme used for relaxing LBE discretization.

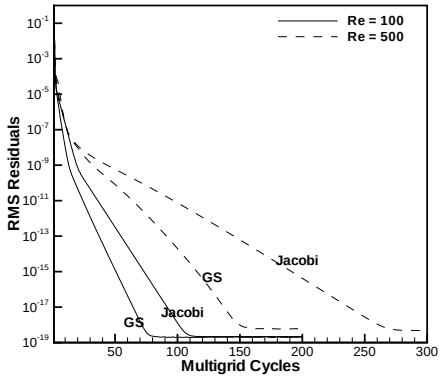

Fig. 10. Comparison of multigrid convergence using four color Gauss-Seidel smoother versus damped Jacobi smoother for driven cavity problem on 129 x 129 grid at various Reynolds numbers.

7 Conclusion

A non-linear multigrid solver has been developed for the steady-state lattice Boltzmann equation. The multigrid solver delivers grid independent convergence rates, and produces the identical steady-state solution as the original LBE time-stepping method. Overall solution efficiency for steady-state problems is increased by one or more orders of magnitude, particularly for problems on highly resolved grids. In its final form, the resulting non-linear multigrid solver appears as a straight-forward implementation of the non-linear LBE time-stepping procedure on each grid level of the multigrid sequence. This enables the design of a modular multigrid solver, which can be implemented by calling existing LBE solution routines. However, the synthesis of this multigrid

scheme was devised by examining the linearized LBE equations, and initially devising a linear multigrid solver for these equations, where stability analysis and algorithmic design are considerably simpler as opposed to the non-linear case. Although the linear multigrid scheme was found to provide equivalent convergence efficiency, the cost of evaluation of a linear relaxation for the LBE equations was found to be substantially higher than the cost of the equivalent non-linear relaxation, thus favoring the non-linear approach. The use of a multi-color Gauss-Seidel scheme is also shown to improve convergence of the non-linear multigrid scheme, although this approach requires modifications to the original LBE solution routines, thus complicating modularization goals.

From an algorithmic viewpoint, future work may include the search for better iterative smoothers, particularly for higher Reynolds number cases. The formulation of a multigrid solver for implicit time-integration of the LBE equation is also an important area of research, in order to enable faster solution of unsteady problems with disparate time and spatial scales. This task will be complicated by the need to compare the effect of the implicit solver on temporal and spatial accuracy, as compared to the explicit time-integration LBE approach. In the steady-state case, such issues do not arise, since the final steady-state is identical for both solution schemes. Extension of the current solver to complex geometries, involving cut-cell cartesian grids is also of interest both for steady and implicit time-integration methods.

References

[1] S. Chen and G. D. Doolen. Lattice Boltzmann method for fluid flows. *Annual Review of Fluid Mechanics*, 30:329–364, 1998.

[2] D. Yu, R. Mei, L.S. Luo, and W. Shyy. Viscous flow computations with the method of Lattice Boltzmann equation. *Progress in Aerospace Sciences*, 39:329–367, 2003.

[3] O. Vangenabeek and D. H. Rothman. Macroscopic manifestations of microscopic flows trough porous media phenomenology from simulation. *Ann. Rev. Earth Planet. Sci.*, 24:63–87, 1996.

[4] X. Shan and H. Chen. Lattice Boltzmann model for simulating flows with multiple phases and components. *Phys. Rev. E*, 47:1815–1819, 1993.

[5] U. Frisch, B. Hasslacher, and Y. Pomeau. Lattice-gas automata for the Navier-Stokes equations. *Phys. Rev. Lett.*, 56:1505–1508, 1986.

[6] G. McNamara and G. Zanetti. Use of the Boltzmann equation to simulate lattice-gas automata. *Phys. Rev. Lett.*, 61:2332–2335, 1988.

[7] F. J. Higuera and J. Jimenez. Boltzmann approach to lattice gas simulations. *Europhys Lett.*, 9:663–668, 1989.

[8] X. He and L.-S. Luo. A priori derivation of the lattice Boltzmann equation. *Phys. Rev. E*, 55:6333–6336, 1997.

[9] T. Abe. Derivation of the lattice Boltzmann method by means of the discrete ordinate method for the Boltzmann equation. *Journal of Computational Physics*, 131:242–246, 1997.

[10] X. Shi, J. Lin, and Z. Yu. Discontinuous Galerkin spectral element lattice Boltzmann method on triangular element. *Int. J. for Numerical Methods in Fluids*, 42:1249–1261, 2003.

[11] J. Töolke, M. Krafczyk, and E. Rank. A multigrid solver for the Discrete Boltzmann equation. *Journal of Statistical Physics*, 107(1):573–591, 2002.

[12] M. Junk, A. Klar, and L.-S. Luo. Asymptotic analysis of the lattice Boltzmann equation. *submitted to Journal of Computational Physics*, 2004.

[13] L.-S. Luo, D. Qi, and L. P. Wang. Applications of the lattice boltzmann method to complex and turbulent flows. In *Lecture Notes in Computational Science and Engineering, eds. M. Breuer, F. Durst, and C. Zenger*, volume 21, pages 123–130, 2002.

[14] J. Tölke, M. Krafczyk, M. Schulz, and E. Rank. Implicit discretization and nonuniform mesh refinement approaches for FD discretizations of LBGK models. *International Journal of Modern Physics C*, 9(8):1143–1158, 1998.

[15] Y. H. Qian, D. d'Humières, and P. Lallemand. Lattice BGK models for Navier-Stokes equations. *Europhys. Lett.*, 17(6):479–484, 1992.

[16] P. L. Bhatnagar, E. P. Gross, and M. Krook. A model for collision processes in gases, I. Small amplitude processes in charged and neutral one-component system. *Phys. Rev.*, 94:511–525, 1954.

[17] J. D. Sterling and S. Chen. Stability analysis of Lattice Boltzmann methods. *Journal of Computational Physics*, 123:196–206, 1996.

[18] E. Isaacson and H. B. Keller. *Analysis of Numerical Methods.* Wiley, New York, NY, 1966.

[19] W. Miller. Flow in the driven cavity calculated by the lattice Boltzmann method. *Phys. Rev. E*, 51:3659–3669, 1994.

[20] C. Hirsch. *Numerical Computation of Internal and External Flows, Volume I: Fundamentals of Numerical Discretization.* Wiley, New York, NY, 1988.

[21] W. L. Briggs, V. E. Henson, and S. F. McCormick. *A Multigrid Tutorial.* SIAM, Philadelphia, PA, 2000.

[22] U. Trottenberg, A. Schuller, and C. Oosterlee. *Multigrid.* Academic Press, London, UK, 2000.

[23] D. J. Mavriplis. Multigrid techniques for unstructured meshes. In *VKI Lecture Series VKI-LS 1995-02*, March 1995.

[24] D. J. Mavriplis. An assessment of linear versus non-linear multigrid methods for unstructured mesh solvers. *Journal of Computational Physics*, 175:302–325, January 2002.

[25] D. J. Mavriplis. Multigrid approaches to non-linear diffusion problems on unstructured meshes. *Journal of Numerical Linear Algebra with Applications*, 8(8):499–512, 2001.