Benchmark on anisotropic problems

Using lattice Boltzmann scheme for anisotropic diffusion

François Dubois [1] [2], Pierre Lallemand and Mohammed Mahdi Tekitek [2]

[1] *Conservatoire National des Arts et Métiers, Paris, France.*

[2] *Numerical Analysis and Partial Differential Equations*

Department of Mathematics, Paris Sud University, Orsay, France.

francois.dubois@math.u-psud.fr, pierre.lal@free.fr, mohamed-mahdi.tekitek@math.u-psud.fr

12 March 2008 *

Abstract

We use lattice Boltzmann method to model anisotropic diffusion problem called "oblique flow". We have adapted a general methodology for equivalent equations to the explicit determination of discrete gradient and fluxes for this problem. We validate this numerical approach with a detailed comparison with finite differences.

Keywords : Anisotropy benchmark, Lattice Boltzmann Method.

1 Lattice Boltzmann scheme

The lattice Boltzmann scheme or Lattice Boltzmann Equation "LBE" is a mesoscopic method and deals with a small number of functions $\{f_i\}$ that can be interpreted as populations of fictitious "particles". We consider in this work the particular D2Q9 [dH92] model (*i.e.* $d = 2$ two-dimensional LBE model with nine velocities $q = 9$). The space is discretized by a regular lattice $\mathcal{L}$ parametrized by a spatial scale Δx. This lattice is composed by a set $\mathcal{L}^0 \equiv \{x_j \in (\Delta x \mathbb{Z})^2\}$ of nodes or vertices. We choose the velocities $c_i, i \in (0 \dots 8)$ defined by: $c = (0,0)$, $(1,0)$, $(0,1)$, $(-1,0)$, $(0,-1)$, $(1,1)$, $(-1,1)$, $(-1,-1)$, $(1,-1)$ and we define Δt as the time step of the evolution of LBE and let the celerity $\lambda \equiv \frac{\Delta x}{\Delta t}$. We choose the velocities $v_i, i \in (0 \dots 8)$ such that $v_i \equiv c_i \frac{\Delta x}{\Delta t} = c_i \lambda$.

* Presented at the Fifth FVCA Conference, Aussois, 8-13 June 2008. To appear in *Hermes Science Publications.*

The populations f_i evolve according to the LBE scheme which can be written as follows [Du08]:

$$(1) \quad f_i(x_j, t+\Delta t) = f_i^*(x_j - v_i \Delta t, t), \quad 0 \leq i \leq 8,$$

where the superscript $*$ denotes post-collision quantities. Therefore during each time increment Δt there are two fundamental steps: collision and advection.

• In the advection step the "particles" move from a lattice node x_j to either itself (with the velocity $v_0 = 0$), one of the four nearest neighbors (with the velocity v_i, $1 \leq i \leq 4$), or one of the four next-nearest neighbors (with the velocity v_i, $5 \leq i \leq 8$).

• The collision step consists in the redistribution of the populations $\{f_i\}$ at each node x_j, and it is modeled by the operator superscript $*$ in (1). This step is best described in the space of moments m_k [dH92]. They are obtained by a linear transformation of vectors f_j: $m_k = \sum_j M_{kj} f_j$. Explicit formula for M_{kj} is given by

$$M = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & \lambda & 0 & -\lambda & 0 & \lambda & -\lambda & -\lambda & \lambda \\ 0 & 0 & \lambda & 0 & -\lambda & \lambda & \lambda & -\lambda & -\lambda \\ -4 & -1 & -1 & -1 & -1 & 2 & 2 & 2 & 2 \\ 4 & -2 & -2 & -2 & -2 & 1 & 1 & 1 & 1 \\ 0 & -2 & 0 & 2 & 0 & 1 & -1 & -1 & 1 \\ 0 & 0 & -2 & 0 & 2 & 1 & 1 & -1 & -1 \\ 0 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 \end{pmatrix}.$$

Note that matrix M is invertible and orthogonal. To simulate diffusion problems, we conserve only the first moment $m_0 \equiv T$ in the collision step and obtain one macroscopic scalar equation. For the other quantities (non-conserved moments), we assume that they relax towards equilibrium values m_k^{eq} that are nonlinear functions of the conserved quantities and set:

$$m_k^* = (1 - s_k)\, m_k + s_k m_k^{eq}, \quad 1 \leq k \leq 8,$$

where s_k is a relaxation rate which satisfy $0 < s_k < 2$ to get a numerically stable scheme. The precise values of s_k are given in the second section. With the following choice of equilibrium values: $m_1^{eq} = 0$, $m_2^{eq} = 0$, $m_3^{eq} = \alpha T$, $m_4^{eq} = \beta T$, $m_5^{eq} = 0$, $m_6^{eq} = 0$, $m_7^{eq} = a_{xx} T$ and $m_8^{eq} = a_{xy} T$ and using Taylor expansion [DLT08], we find the diffusion equation up to order three in Δt:

$$\frac{\partial T}{\partial t} - \mathrm{div}(\mathbf{K} \nabla T) = \mathrm{O}(\Delta t^3).$$

where $\mathbf{K} = (k_{i,j})_{1 \leq i,j \leq 2}$ is the diffusion tensor where $k_{11} = \frac{\lambda^2 \Delta t}{6}(\frac{1}{s_1} - \frac{1}{2})(4 + \alpha + 3a_{xx})$, $k_{12} = k_{21} = \frac{\lambda^2 \Delta t}{2}(\frac{1}{s_1} + \frac{1}{s_2} - 1) a_{xy}$ and $k_{22} = \frac{\lambda^2 \Delta t}{6}(\frac{1}{s_2} - \frac{1}{2})(4 + \alpha - 3a_{xx})$.

2 Numerical results

• **Test 3 Oblique flow,** $\min = 0$, $\max = 1$**, uniform rectangular mesh,** `mesh2` We have used LBE D2Q9 scheme to solve the following anisotropic diffusion problem so called "oblique flow":

$$-\mathrm{div}(\mathbf{K}\nabla u) = 0 \quad \text{in} \quad \Omega =]0,1[^2,\ u = \overline{u} \quad \text{on} \quad \partial\Omega. \tag{2}$$

where $\mathbf{K} = R_\theta \,\mathrm{diag}(1, 10^{-3})\, R_\theta^{-1}$, R_θ is the rotation of angle $\theta = 40$ degrees, and $\overline{u} = 1$ on $(0, 0.2) \times \{0\} \cup \{0\} \times (0, 0.2)$, 0 on $(0.8, 1) \times \{1\} \cup \{1\} \times (0.8, 1)$, $\frac{1}{2}$ on $(0.3, 1) \times \{0\} \cup \{0\} \times (0.3, 1), \frac{1}{2}$ on $(0, 0.7) \times \{1\} \cup \{1\} \times (0, 0.7)$. Figure 1 and Figure 2 show the approximate solution on the following uniform rectangular mesh, `mesh2`: $(2^{i+1} \times 2^{i+1})$, $i = 2..7$, calculated by D2Q9 scheme after convergence (*i.e.* 2.10^6 iterations) with $s_1 = 1.3$, $s_2 = 1.8$ and $\beta = 1$ and other parameters are fixed to have $\mathbf{K}$ as the diffusion tensor given (2). To impose $\overline{u}$ on boundary we have use a first order scheme for boundary conditions described in [DLT08].

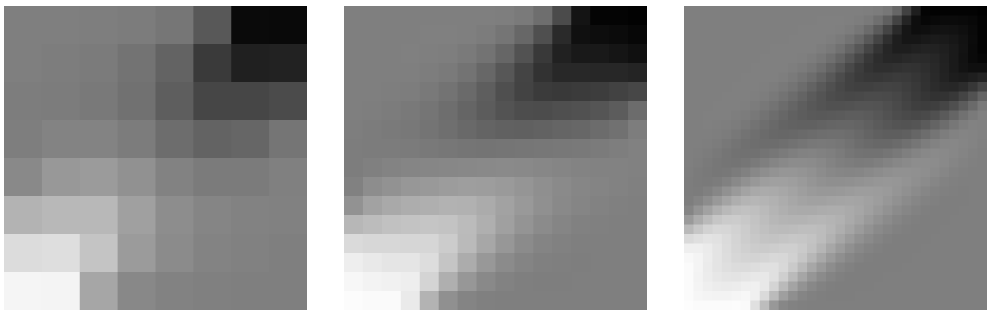

Figure 1. *Solutions for the oblique flow on mesh2_ i for i=2 (left), i=3 (center), i=4 (right). The Grey scale of the figure corresponds to a linear variation from* 0 *(black) to* 1 *(white).*

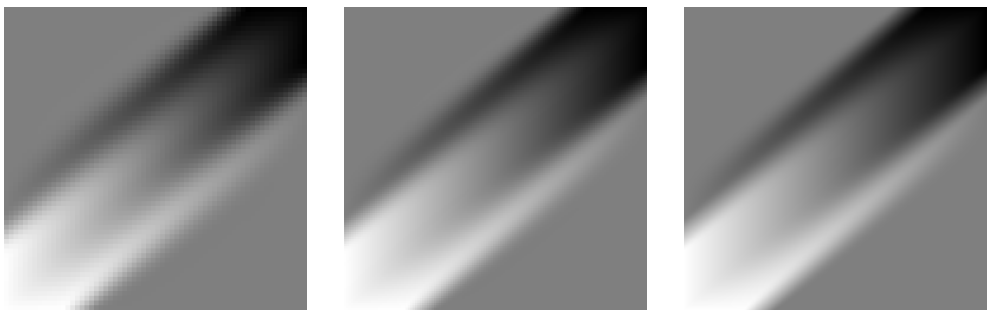

Figure 2. *Solutions for the oblique flow on mesh2_ i for i=5 (left), i=6 (center), i=7 (right).*

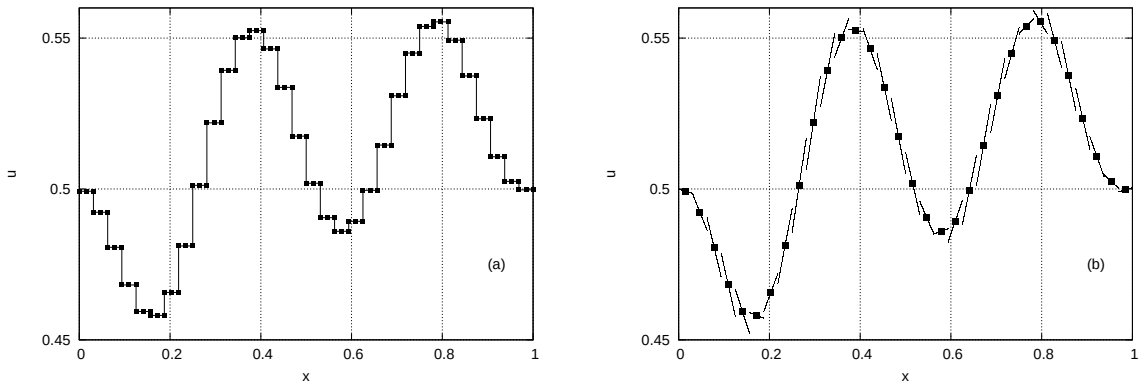

Figure 3. *Solutions for the oblique flow on mesh_4 (*32×32*) at* $y = 17$*. (*a*) Approximate solution vs* x *on the center and boundary of the volume control* K*. (*b*) Approximate solution vs* x *using Taylor expansion on the center of the volume control* K *where discrete* ∇u *is given by LBE.*

We note that we can improve the approximate solution by using the discrete gradient ∇u without any additional computation (as ∇u is given by LBE). Figure 3 (a) shows the interpolate solution and Figure 3 (b) shows the solution using Taylor expansion of order one in Δx, where ∇u is given by LBE.

i	nunkw	nnmat	sumflux	umin	umax
1	9×16	-	5.27E-16	1.14E-01	8.86E-01
2	9×64	-	2.44E-15	3.78E-02	9.62E-01
3	9×256	-	1.01E-14	1.11E-02	9.89E-01
4	9×1024	-	3.59E-14	7.14E-03	9.93E-01
5	9×4096	-	1.42E-13	3.53E-03	9.96E-01
6	9×16384	-	5.75E-13	1.76E-03	9.98E-01
7	9×65536	-	7.88E-10	9.36E-04	9.99E-01

Table 1. *Number of unknowns (nunkw), the discrete flux balance (sumflux), value of the minimum (umin) and value of the maximum (umax) vs the mesh size* i*.*

Table 1 shows the following quantities:

• nunkw: number of unknowns.

• nnmat: number of nonzero terms in the matrix. As the lattice Boltzmann scheme is an explicit method, designed to simulate time dependent problems, we have no matrix to inverse to find solution (like in classical numerical method finite elements or finite

volumes) but we have to make many iterations to reach convergence.

• sumflux: the discrete flux balance, that is: flux0+flux1+fluy0+fluy1-sumf, where flux0, flux1, fluy0, fluy1 are the outward fluxes at the boundaries $x = 0$, $x = 1$, $y = 0$, $y = 1$, for example flux0 is an approximation of $-\int_{x=0} \mathbf{K}\nabla u.n ds$, and sumf= $\sum_{K\in\tau} |K| f(x_j)$ where x_j denotes lattice node and represents the center of the control volume K. Since $f = 0$ in our test (no source term in equation (2)), we have sumf= 0 and sumflux= 0 (see Table 1). Here the discrete gradient ∇u on the boundaries is computed using the Dirichlet boundary condition $\overline{u}$ and the mass flux j [DLT08] on the boundaries. Note that when refining the mesh, the sumflux variable looses 6 orders of magnitude. This indicates the difficulties to reach the steady state.

• umin: value of the minimum of the approximate solution. Table 1 shows that the minimum umin converge to 0 and umin > 0. The variable umax is the value of the maximum of the approximate solution. Table 1 shows that the maximum umax converge to 1 and umax < 1. Note that the effective grid points follow the classical cell center finite volume methodology. Hence the points at which umin and umax are determined are located at $\frac{\Delta x}{2}$ from the actual boundary and thus data in Table 1 have not been extrapolated to the boundary.

i	flux0	flux1	fluy0	fluy1
1	1.46E-01	2.57E-01	1.52E-01	-5.57E-01
2	1.04E-01	-1.04E-01	1.89E-01	-1.89E-01
3	2.46E-01	-2.46E-01	4.90E-02	-4.90E-02
4	1.97E-01	-1.97E-01	9.53E-02	-9.53E-02
5	1.75E-01	-1.75E-01	1.16E-01	-1.16E-01
6	1.89E-01	-1.89E-01	1.02E-01	-1.02E-01
7	1.96E-01	-1.96E-01	9.56E-02	-9.56E-02

Table 2. *Outward fluxes at the boundaries with Taylor expansion. Results obtained using the discrete gradient ∇u given by the LBE method.*

i	flux0	flux1	fluy0	fluy1
1	-3.18E-01	3.18E-01	2.02E-02	-2.02E-02
2	-6.78E-02	6.78E-02	-1.28E-01	1.28E-01
3	2.51E-01	-2.51E-01	1.61E-01	-1.61E-01
4	2.50E-01	-2.50E-01	1.60E-01	-1.60E-01
5	1.76E-01	-1.76E-01	8.30E-02	-8.30E-02
6	1.76E-01	-1.76E-01	8.31E-02	-8.31E-02
7	1.96E-01	-1.96E-01	1.02E-01	-1.02E-01

Table 3. *Outward fluxes at the boundaries with Taylor expansion. Results obtained using the discrete gradient ∇u given by finite difference method.*

In Table 2 and Table 3 we show the values (flux0,flux1,fluy0,fluy1) the outward fluxes at the boundaries *vs* the i which represent the mesh size of mesh2 (equal to $2^{(1+i)} \times 2^{(1+i)}$). The discrete gradient ∇u on the boundaries is obtained in Table 2 by using fluxes at the boundaries [DLT08] (Fourier low), Dirichlet boundaries condition $\overline{u}$ and Taylor expansion [Du08]. In table 3 the fluxes are obtained by using the discrete gradient ∇u on the boundaries obtained by parabolic interpolation. We note here that the outward fluxes computed by both above methods converge to the same value on the fine grids.

i	ener1	ener2	eren
1	2.42E-01	2.02E-01	1.64E-02
2	2.53E-01	2.55E-01	9.39E-04
3	2.58E-01	2.75E-01	6.16E-03
4	2.55E-01	2.66E-01	4.07E-03
5	2.44E-01	2.50E-01	2.72E-03
6	2.42E-01	2.45E-01	8.95E-04
7	2.42E-01	2.43E-01	3.00E-04

Table 4. *Two computations of energy ener1 and ener2. Results obtained using the discrete gradient ∇u given by the LBE method.*

i	ener1	ener2	eren
1	5.83E-01	2.21E-01	6.20E-02
2	7.71E-01	3.35E-01	5.65E-02
3	5.67E-01	4.93E-01	1.30E-02
4	3.56E-01	4.37E-01	1.84E-02
5	2.59E-01	2.88E-01	1.00E-02
6	2.47E-01	2.57E-01	3.92E-03
7	2.43E-01	2.60E-01	6.62E-03

Table 5. *Two computations of energy ener1 and ener2. Results obtained using the discrete gradient ∇u given by finite difference method.*

Table 4 and able 5 show the following quantities:
• ener1: is energy given by ener1 $= \int_\Omega \mathbf{K} \nabla u . \nabla u \, \mathrm{d}x$. To compute ener1 we need the discrete gradient ∇u on all nodes x_i of the mesh. This discrete gradient is given by the method using moments (m_1, m_2) or (m_5, m_6), for more details see [DLT08].
• ener2: is energy given by ener2 $= \int_{\partial\Omega} \mathbf{K} \nabla u . nu \, \mathrm{d}x$. We note here that to compute ener2,

we use only the boundary outward normal fluxes. Since $f = 0$ (*i.e.* no source term in equation (2)), the quantities ener1 and ener2 should converge to the same value. Table 1 shows that these two discrete quantities converge to the same value and shows that relative error between ener1, ener2 given by: $\text{eren} = |\text{ener1} - \text{ener2}|/\max(\text{ener1}, \text{ener2})$, converge to zero on fine grids.
Table 5 shows the value ener1, ener2 and eren computed using discrete gradient ∇u which is obtained by finite differences (using a 9 points stencil). We note here that the results obtained by LBE (see Table 4) are more efficient than those obtained by finite differences method.

3 Comments on the results

The lattice Boltzmann scheme is a mesoscopic method which have a lot of unknowns per node of the lattice (9 unknowns in D2Q9 model), however linear combinations of these local unknown allow to compute the first order and the second order space derivatives of the solution. We have shown how to adapt the lattice Boltzmann scheme to simulate an anisotropic diffusion problem and present numerical results showing that the scheme converges to the solution.
The lattice Boltzmann scheme has been designed for time dependent situations and is founded on the fact that the scheme is exact for particular advection velocities. We have used this method for a steady diffusion problem. The scheme converges slowly towards the stationary solution with a time constant proportional to the number of nodes in the mesh. We have not used any acceleration techniques like embedded grids.

References

[dH92] d'Humières D., «Generalized lattice-Boltzmann equation», *AIAA Rarefied Gas Dynamics: Theory and Simulations Progress in Astronautics and Aeronautics*, vol. 159, 1992, p. 450–458.

[Du08] Dubois F., «Equivalent partial differential equations of a lattice Boltzmann scheme», *Computers & Mathematics with Applications*, vol 55, 2008, p. 1141–1149.

[DLT08] Dubois F., Lallemand P., Tekitek M. M., «Using Lattice Boltzmann scheme for anisotropic diffusion problems», *FVCA5*, (submitted), 2008.