

The Lattice Boltzmann Method (continued)

The Lattice Boltzmann Equation in the Bhatnagar-Gross-Krook approximation is

$$n_i(\mathbf{r} + \mathbf{c}_i, t + 1) = n_i(\mathbf{r}, t) - \frac{n_i - n_i^{\text{eq}}}{\tau} .$$

Here τ is a single relaxation time, and n_i^{eq} is the equilibrium configuration, which is given by

$$n_i^{\text{eq}} = \rho w_i \left(1 + \frac{\mathbf{u} \cdot \mathbf{c}_i}{c_s^2} + \frac{(\mathbf{u} \cdot \mathbf{c}_i)^2 - c_s^2 \mathbf{u}^2}{2c_s^4} \right) ,$$

where c_s is the speed of sound given by

$$\sum_i w_i (\mathbf{c}_i)_a (\mathbf{c}_i)_b = \delta_{ab} c_s^2 ,$$

and w_i are a set of directional weights normalized to unity. The discrete lattice velocities and weights are constrained by conservation of mass, momentum, and angular momentum:

$$\sum_i n_i^{\text{eq}} = \rho ,$$

where ρ is the fluid density,

$$\sum_i n_i^{\text{eq}} \mathbf{c}_i = \rho \mathbf{u} ,$$

where $\mathbf{u}$ is the fluid velocity, and

$$\sum_i n_i^{\text{eq}} (\mathbf{c}_i)_a (\mathbf{c}_i)_b = \rho [(\mathbf{u})_a (\mathbf{u})_b + c_s^2 \delta_{ab}] .$$

The conservation laws imply the following conditions:

$$\sum_i w_i = 1 ,$$

$$\sum_i w_i \mathbf{c}_i = 0 ,$$

$$\sum_i w_i (\mathbf{c}_i)_a (\mathbf{c}_i)_b = P \delta_{ab} ,$$

where P is the fluid pressure.

It can be shown that with these restriction the Navier-Stokes equations are obeyed with the fluid pressure given by

$$P = \rho c_s^2 ,$$

and the kinematic viscosity given by

$$\nu = c_s^2 \left(\frac{1}{\omega} - \frac{\delta t}{2} \right) ,$$

where $\omega = 1/\tau$ is the relaxation frequency, and δt is the cellular automaton time step which we have choosen to be $\delta t = 1$.

Choice of Discrete Velocities

The set of allowed velocites in the Lattice Boltzmann Models is restricted by conservation of mass and momentum, and by rotational symmetry (isotropy). However, these restrictions turn out to be much less severe than in the Lattice Gas Cellular Automaton Models.

The following table taken from Sauro Succi's book gives some popular lattices. The weight of the velocity can be thought of as the mass of the fluid particle. The magnitude of the velocity of the particle is determined such that it moves to the the nearest lattice site in the direction of its velocity: this determines its kinetic energy.

The D1Q3 model has a 1-D lattice with one zero velocity and two oppositely directed velocities which move the fluid particle to the left and right neighbor lattice sites. The D1Q5 model extends D1Q3 by moving particles to the next-nearest neighbor sites in addition.

Model	c^2	Energy	Weight
D1Q3	1/3	0	4/6
		1/2	1/6
D1Q5	1	0	6/12
		1/2	2/12
		2	1/12
D2Q9	1/3	0	16/36
		1/2	4/36
		1	1/36
D3Q15	1/3	0	16/72
		1/2	8/72
		3/2	1/72
D3Q19	1/3	0	12/36
		1/2	2/36
		1	1/36

The D2Q9 model is a 2-D lattice (D2) with 9 discrete velocities: 0, N, S, E, W, NE, NW, SE, SW.

The D3Q15 model is a 3-D lattice with 15 discrete velocities: 0, 6 towards face centers, and 8 towards vertices of a cube.

The D3Q19 model is a 3-D lattice with 19 discrete velocities: 0, 6 velocities to the face centers, and 12 towards edge centers of a cube.

Two-dimensional flow using the D2Q9 model

Dr. Sauro Succi's program `lbe.f` simulates 2-D flow in a rectangular region. This program is translated in `lbe.cpp`.

The discrete velocities are chosen according to the D2Q9 model is a 2-D lattice (D2) with 9 discrete velocities: 0, N, S, E, W, NE, NW, SE, SW. In the program, `npop = 9` is this number of velocities.

This velocities have

$$c_i = \begin{cases} 0 & \text{for } c = 0 \\ 1 & \text{for N, S, E, W} \\ \sqrt{2} & \text{for NE, NW, SE, SW} \end{cases} .$$

and weights

$$w_i = \begin{cases} \frac{4}{9} & \text{for } c = 0 \\ \frac{1}{9} & \text{for N, S, E, W} \\ \frac{1}{36} & \text{for NE, NW, SE, SW} \end{cases} .$$

The speed of sound is given by

$$2c_s^2 = 1 \times 0^2 \times \frac{4}{9} + 4 \times 1^2 \times \frac{1}{9} + 4 \times (\sqrt{2})^2 \times \frac{1}{36} = \frac{2}{3} .$$

We need to discuss some additional features of the Lattice Boltzmann Equation and its numerical implementation.

Boundary Conditions

Various types of boundary conditions are possible:

- Periodic boundary conditions are useful for modeling bulk systems because they tend to minimize finite size edge effects.
- No-slip boundary conditions are appropriate for most fluids in contact with a wall.
- Frictional slip (or the limiting case of free-slip) boundary conditions may be appropriate for smooth boundaries with small (or negligible) friction exerted on the flowing gas or liquid.
- Open inlets and outlets.

Periodic Boundary Conditions

Boundary conditions are straightforward to derive once the model is specified. Consider the D2Q9 model with a rectangular region. The discrete velocities are numbered as follows:

6	2	5
3	0	1
7	4	8

The boundary values at the West end of the region ($x = 0, y$) are implemented by transferring the densities with positive x component of velocity from the East boundary ($x = n_x, y$):

```
void pbc() {
    // East
    for (int j = 1; j <= ny; j++) {
        n[1][0][j] = n[1][nx][j];
        n[5][0][j] = n[5][nx][j];
        n[8][0][j] = n[8][nx][j];
    }
}
```

Note that it is only necessary to transfer three of the 9 densities that will then flow into the region.

No-slip Boundary Conditions

Let's consider the North wall with lattice sites ($x, y = n_y + 1$). The appropriate boundary conditions, which will ensure that the fluid velocity at the wall is zero, are implemented as follows:

```
void mbc() {  
  
    // North  
    for (int i = 1; i <= nx; i++) {  
        n[4][i][ny+1] = n[2][i][ny];  
        n[8][i][ny+1] = n[6][i][ny];  
        n[7][i][ny+1] = n[5][i][ny];  
    }  
}
```

We only need to set the densities with negative y component of velocity, namely (4,7,8). The boundary conditions are implemented by simply reversing these velocities. This fluid velocity normal to the wall is proportional to

$$(n_6 + n_2 + n_5) - (n_7 + n_4 + n_8) = 0 .$$

The tangential fluid velocity component is proportional to

$$(n_5 + n_1 + n_8) - (n_6 + n_3 + n_7) = n_1 - n_3 .$$

Since the components $n_{1,3}$ parallel to the wall do not change during the simulation, we can set $n_1 = n_3$ at the wall initially so the parallel velocity component will remain zero.

Obstacle

Succi's program also allows for a thin vertical obstacle with no-slip boundary conditions centered at $(x = n_x/4, y = n_y/2)$.

Poiseuille Flow Problem

This is viscous flow through a channel under the action of a pressure gradient. With no-slip boundary conditions at the wall of the channel the flow develops a parabolic velocity profile which is stable up to Reynolds numbers of about 2000.

There is a problem with simulating Poiseuille flow using the Lattice Boltzmann Equation because the system behaves like an ideal gas with equation of state

$$P = \rho c_s^2,$$

where c_s is the speed of sound. For the D2Q9 model $c_s^2 = 1/3$. In addition, the flow is incompressible with constant ρ . Thus in equilibrium, the pressure P is constant and there cannot be a pressure gradient to drive the flow!

In a real incompressible fluid, the speed of sound is very large compared with the fluid velocity, and small pressures gradients are consistent with almost constant density. But in the lattice model, the speed of sound is comparable to the fluid velocity! A trick to simulate a constant pressure gradient is to introduce a *body force* which transfers the same momentum to the fluid to overcome viscosity as would a pressure gradient. This is done in the program as follows:

```
cs2 = 1.0 / 3.0;                // speed of sound squared
visc = (1.0 / omega - 0.5) * cs2; // kinematic viscosity
fpois = 8.0 * visc * uf / ny / double(ny); // Poiseuille force
fpois = rho * fpois / 6;
for (int i = 1; i <= nx; i++)
for (int j = 1; j <= ny; j++) {
    n[1][i][j] += fpois;
    n[5][i][j] += fpois;
    n[8][i][j] += fpois;

    n[3][i][j] -= fpois;
    n[6][i][j] -= fpois;
    n[7][i][j] -= fpois;
}
```

Here $\text{visc} = \nu$ is the kinematic viscosity, and u_f is the desired final fluid velocity in the $+x$ direction. Note that if $u_f > 0$ then the densities $n_{1,5,8}$ of particles moving in the $+x$ direction are increased by a constant amount

at each lattice site, and the densities $n_{3,6,7}$ in the opposite direction are decreased correspondingly: thus momentum is continually injected into the fluid which preserving constant density.