Simulating suspensions in confined geometries with hydrodynamic interaction in Espresso

Marcello Sega

September 11, 2008

This document describes four basic checks performed in order to attest the reliability/accuracy of Espresso in simulating particles in interaction with Lattice-Boltzmann fluid and confining walls under the effect of external forces. These requirements are usually mandatory to be able to simulate a wide class of problems like electro-osmotic flow and electrophoresis. No electrostatics is indeed used in the following tests, though the basic ingredients needed for introducing 2D electrostatics with the ELC algorithm are accounted for.

The simplest test case that needs all of these requirements, and that is at the same time suitable for a validation against analytical results, is the Poiseuille flow between parallel plates.

1 The system

Consider a fluid of density ρ and shear viscosity μ, confined between two plates (with normal along the z axis, and distance $2L$), with a pressure gradient ∇p acting along the x direction and other generic forces per unit volume $\mathbf{g}$ acting on the fluid elements. The Navier-Stokes equations

$$\rho\left[\partial_t \mathbf{v} + (\mathbf{v}\cdot\nabla)\mathbf{v}\right] = -\nabla p + \mathbf{g} + \mu\nabla^2\mathbf{v}$$

reduce, due to the simmetries of the problem and to the fact that we are looking to the stationary solution, to

$$\frac{\partial p}{\partial x} = \mu\frac{\partial^2}{\partial z^2}v_x(z) + \mathbf{g}.$$

The pressure gradient can be expressed in terms of the force per fluid volume $\frac{\partial p}{\partial x} = \frac{dF}{Sdx} = f$ where dF is the total force acting on the xy surface of area S, and dx is the infinitesimal width of the slab considered, so that f is the force per fluid volume, or force density. Suppose now that $\mathbf{g} = 0$; integrating the equation for $v_x(z)$, imposing as a boundary condition that $v_x(0) = 0$, leads to the velocity profile

$$v_x(z) = \frac{f}{2\mu}\left[z^2 - L^2\right].$$

If, instead, no pressure is acting on the fluid, but some particles are dispersed in it, and some force is acting on them, rather than on the fluid, the pressure gradient has to be set to zero, and the force density $g(z)$ is

actually the force acting on the particles (say, G) divided by the volume of fluid occupied by them. In other words, $g(z) = G(z)\rho_p(z)$, where $\rho_p(z)$ is the particle number density profile. For example, if only a position independent external force G — and no internal one — is acting on the particles, the profile will be flat, and $g(x) = G\rho_p$. This case is rather unphysical (e.g. gravitation would act on the particles as well as on the fluid) but it is interesting for testing purposes, because it will lead to a solution analogous to the pure fluid case, allowing to test the particle-LB interaction part of the code.

The general solution is actually given by

$$v_x(z) = \frac{1}{\mu}\int_0^z \int_0^{z'} g(z'')dz'dz'' + cz + d.$$

If there are no inter-particle forces, but just an external force G, the solution is then

$$v_x(z) = \frac{G\rho_p}{2\mu}\left[z^2 - L^2\right].$$

The general soution can be useful in presence of internal forces, like electrostatic interaction between the particles. In this case the Poission-Boltzmann solution for the density profile $\rho_p(z)$ can then be used to solve for the velocity profile of, say, electro-osmotic flow, in case an external homogeneous field is also present.

2 Espresso parameters

Even if the formal expression for the two cases is basically the same, from the simulation pof view the situation is completely different. Namely, in the first case, the system can be simulated employing only a pure LB simulation, without introducing thermal fluctuations, and applying a constant force to the nodes, that shifts the momentum distributions. This is accomplished by something like

```
thermostat off
lbfluid density $lb_dens viscosity $lb_visc agrid $lb_grid
          tau $lb_tau ext_force 1.0 0.0 0.0
```

where the `ext_force` parameter sets the force density acting on the fluid, and is exactly f in the above formulas.

In the case of beads dispersed in the fluid, the coupling between MD and LB has to be switched on, therefore thermal fluctuations are introduced into the system. The constant force is applied on the particles using a line like

```
for { set i 0 } { $i <  [setmd n_part] } { incr i } {
   part $i ext 1.0 0.0 0.0
}
```

where the parameter `ext` is exactly the force G mentioned above. Obviously, the pressure on the fluid has to be switched off:

```
lbfluid ext_force 0.0 0.0 0.0
```

The thermal fluctuations and the coupliong with the particles are switched on by the command

```
thermostat lb $temperature
lbfluid friction  $lb_gamma
```

The other parameters involved in the simulation are the fluid density, viscosity, grid spacing, LB relaxation time and integration time. The tests have been performed for non-unit parameters in order to check for correct factors in the unit conversion throughout the code. The only case not taken into account is that of a ratio between the LB relaxation time `$lb_tau` and the integration timestetp, different from one. It is advisable to run some checks like the actual ones if one wants to use a different ratio. Note also that the effective friction coefficient for the particles is given by a combination of the `$lb_gamma` parameter and the fluid viscosity `$lb_fluid` and it is lattice-dependent [See Eq. 20 in Ahlrichs and Dünweg, J. Chem. Phys. **111** 8225 (1999) - ahlrichs99a.pdf]

3 Notes on Espresso commands

- When using LB in the Lederer version in presence of confining surfaces, the kind of boundary condition has always to be specified by the command `lbboundaries`. In case some MD particles are also present in the system, the only suitable condition is bounce-back, which is switched on by calling

  ```
  lbboundaries bounce_back.
  ```

  This is because of the algorithm which interpolates the velocities of the fluid in the neighborhood of an MD particle (see function `lb_assign_boundary_population()` in `lb.c`) only works for bounce-back boundary conditions.

- The viscosity parameter `$lb_visc` in the `lbfluid` command is the kinematic viscosity $\nu = \frac{\mu}{\rho}$, so one should multiply it by `$lb_dens` in order to obtain the shear viscosity μ

- In the actual Espresso implementation you can access one of the three components of the velocity at a specific (x,y) location, for every value along the z axis (not along the z or y, though) with the following command

  ```
  analyze fluid velprof 0 2 $ix $iy
  ```

  where the first argument identifies the velocity component (x in this case), the second specifies the direction along which one want to compute the profile, and `$ix` and `$iy` are integer numbers specifing the location on the $x-y$ plane of the mesh where to sample the velocities.

- Plane walls have been set with the following commands

  ```
  constraint wall normal 0.0 0.0  1.0
              dist [ expr $wallposlow ] type $walls
  constraint wall normal 0.0 0.0  -1.0
              dist [ expr -1.*($wallposhigh) ] type $walls
  ```

  Notice that the normal of the second wall is negative, and therefore the wall has to be moved by a negative amount in order to be placed at `$wallposhigh`. See also the note on walls in Section 6.

4 Results

Four system have been simulated, namely (a) the pure LB fluid, (b) fluctuating LB fluid with no particles, (c) fluctuating LB with particles and pressure applied on the fluid, (d) fluctuating LB with particles and force applied to the particles. In each case the simulation box has been set to $9 \times 9 \times 9$, using a mesh size of 0.5, an integration timestep and LB relaxation time of 0.01. In cases (c) and (d) 40 particles have been added to the system, with a truncated LJ potential of $\epsilon = 1.0$ and $\sigma = 1.0$. In cases (b),(c) and (d) the temperature has been fixed to 1.0. The LB fluid has a shear viscosity of 1.36 and a density of 3.75, i.e. a kinematic viscosity (Espresso input parameter) of 1.36/3.75. The coupling constant between MD particles and LB fluid has been set to 10.98. Smooth truncated LJ walls has been placed at 0.5 and at `$box_z` - 0.5. (See Section 6)

In figures 1 and 2 the time evolution (after a first initial relaxation) of the velocity in the node next to the central one are presented, for cases (c) and (d).

The obtained velocity profiles (TODO: put here the description f either the tcl function or a c implementation) are shown in Figure 3. The results differ slightly from the expected theoretical result: keeping as a fixed parameter the channel halfwidth of 3.75, the results fit better using a viscosity

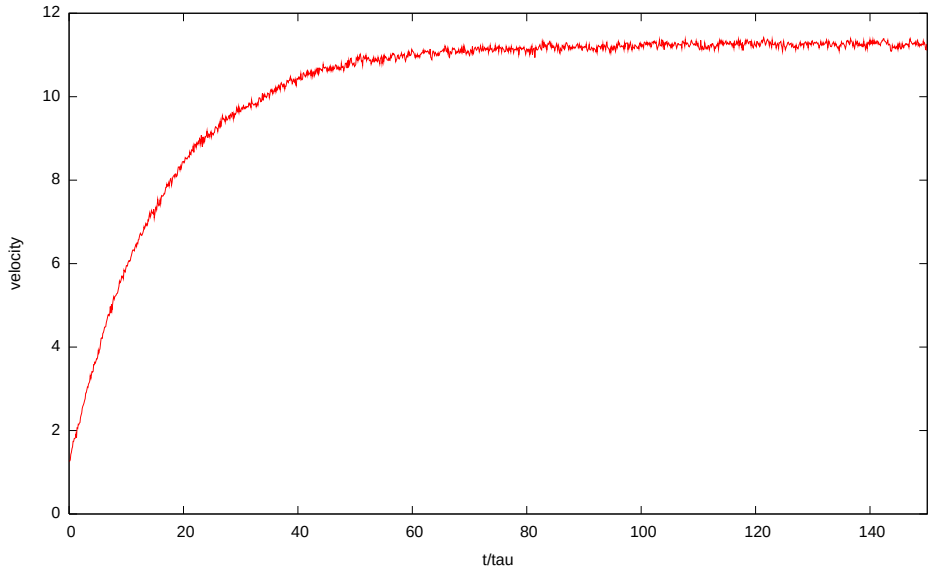

Figure 1: Time evolution of the velocity in the bin next to the center for case (c). (not normalized)

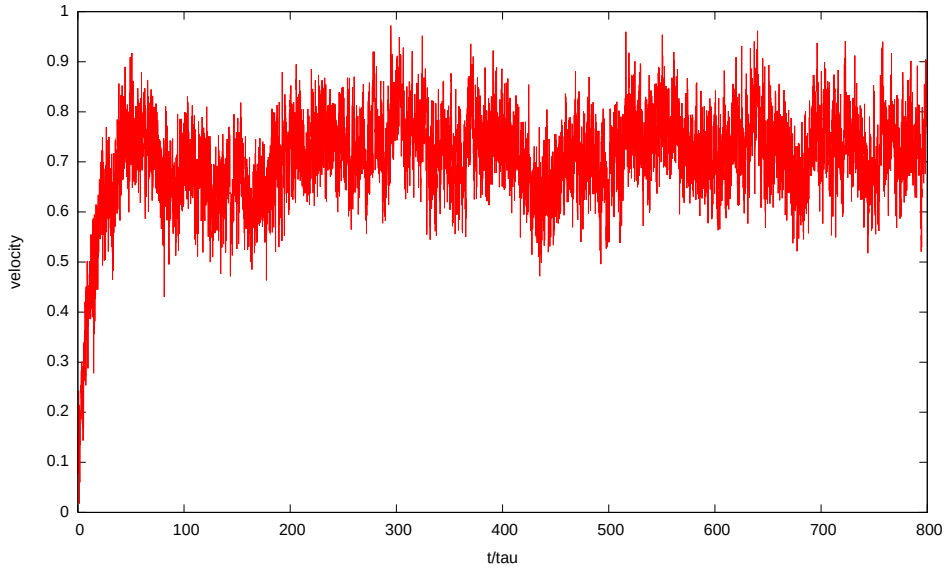

Figure 2: Time evolution of the velocity in the bin next to the center for case (d). (not normalized)

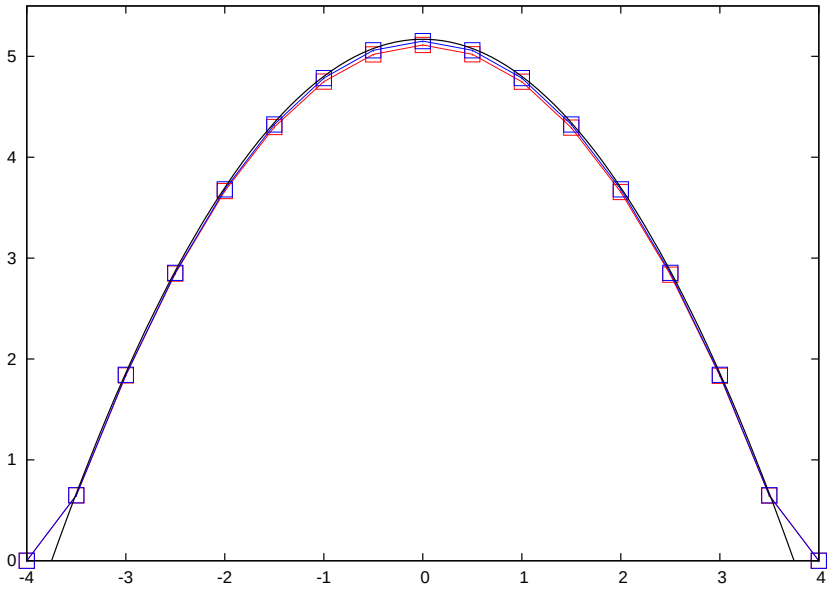

Figure 3: Velocity profile as a function of the position with respect to the channel center. Solid line, theory (no free parameters); blue squares, case (a); red squares, case (b).

of 1.37 and 1.38 for cases (a) and (b), respectively, rather than using the actual viscosity of 1.36. (note: in every plot quantities are in simulation units)

While these differences should be in general acceptable, it is interesting to find which values of the parameters (viscosity μ and hydrodynamic channel half-width L) minimize the difference between the simulation results and the theoretical formula. In Figure4 it is shown in logarithmic scale the difference $\|v_{sim} - v_{theory}\|$ for the "optimal" parameters. For the pure LB case (a) the difference was minimized by $\mu = 1.3601$ and $L = 3.74361$ (the outcome being more sensitive on L than on μ). For the (b) and (c) case the optimal viscosity is 1.37 and 1.375, respectively, and the width L was found to be 3.745 for both cases. In the (b) and (c) cases the presence of fluctuations made the optimization less accurate than the pure LB case (a). In the latter case, the *relative* differences from the analytical expression are always below 2×10^{-5}, while for the fluctuating cases (b) and (c) it is always below 2×10^{-3}.

The fourth system (d) consisted of particles dragged witha constant external force $G = 1.0$, in a fluctuating LB fluid. The effective force acting on

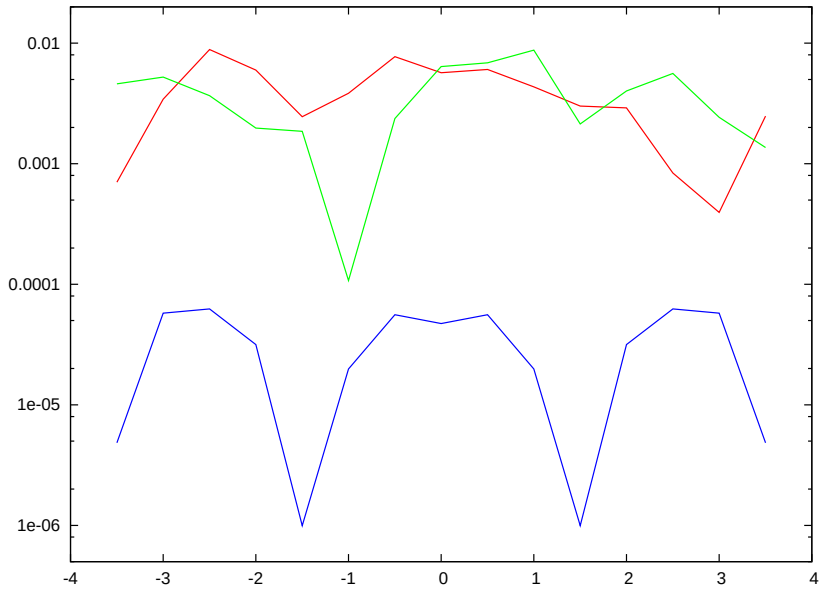

Figure 4: Absolute difference between the LB data and the analytic furmula with optimized parameters. Blue line, case(a); red line, case (b); green line, case (c)

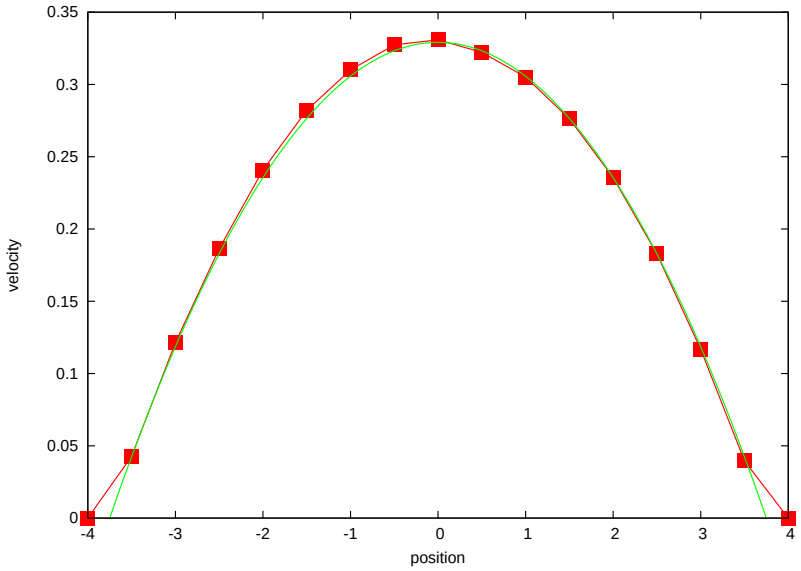

Figure 5: Velocity profile as a function of the position with respect to the channel center. Solid line, theory with optimized parameters; red squares, case (d)

the fluid, ρG cannot actually be computed exactly, because the walls are not hard ones. It is therefore not possible to make an exact comparison with only one simulaation, but just check the fit against the parabolic profile. Keeping as fixed at 1.76 the shear viscosity of the fluid, and optimizing the volume occupied by the particles as $9 \times 9 \times 7.7$ (which is essentially the same as keeping a fixed volume and fitting the viscosity) gives the results shown in Figure 5.

5 Notes on the code version

The code used to perform these tests is derived from the so-called "Lederer" one. LB is implemented in the way of Burkhard Dünweg, Schiller and Ladd (Phys Rev E 76, 036704). The source code is available at

`http://fias.uni-frankfurt.de/~sega/ELS.tgz`

Note that as of today (September 11, 2008), one change to the factors employed is still debated, so if you use this version, remember that the maintainer is not responsible for it. Notice that using the original "Lederer" or `cvs` implementation it is not possible to reproduce the correct velocity profile for gird spacing different from 1, whenever fluctuations are switched

on. Simulations with a grid spacing of 1 should be safe in the "Lederer" and `cvs` versions. The part of debated code is

```
double mu = temperature/lbmodel.c_sound_sq*tau*tau/(agrid*agrid);
```

to which a density factor has been added, becoming

```
double mu = temperature/lbmodel.c_sound_sq*tau*tau*agrid;
```

6 Misc notes

- The main limitation of the actual Espresso implementation is just its inability to deal with LB fluid and non-planar confinement surfaces. If this is the case, please consider to use DPD as a Navier-Stokes solver: the actual Espresso implementation has the option to fully tune the boundary conditions for the fluid (from full slip to stick, while standard DPD can only reproduce full slip on smooth walls)

- The box width has been intentionally not fully exploited in order to check whether positioning walls not at the box boundaries might influence the results. The results with pure fluid with a box of $9 \times 9 \times 9$ and walls located at 0.5 and 8.5 and, with a box of $9 \times 9 \times 8$ and walls located at 0.0 and 8.0 gave indistinguishable results. This point is of interest when using algoruthms like ELC, which require an additional layer in which particles are not entering, or when one could want to match the position of hydrodinamic boundaries (i.e., with bounce-back, half in between the wall nodes and the first fluid node, to first order in viscosity) and particle boundaries (e.g. with truncated LJ interaction, roughtly at σ). In this case one should add an offset `$lj_off` to the LJ interaction between walls and particles to "displace" the two walls (MD and LB). In order to do this, the walls cannot be placed at the box boundary, but at least at `$lj_off` and `$box_z - $lj_off`.