Master Thesis Intermediary Report I

Jérémie Guy

June 11, 2024

Abstract

First report about the progress and scope of the Master Thesis on fluid dynamics using the Lattice-Boltzmann method.

1 Introduction

Computer simulations are an indispensable tool to better understand the complexity of the real world. They are mandatory in a wide range of real-life applications where simple calculations would not be enough to predict the outcome of a particular behavior. An example of such behavior is fluid dynamics: with the ever-changing flow and complex interactions, one can only expect an outcome using precise simulations that reflect reality. For this Master Thesis we will concentrate on a particular example of fluid dynamics: the cerebral arterial blood flow, specifically around a branching tube containing a clot.
In this first part, we will implement a basic flowing tube using the Lattice-Boltzmann method and check that everything is in order before making the system more complex.

2 Methodology

The method we used to implement such fluid dynamics, as stated above, will be the Lattice-Boltzmann (LB) method. LB is the successor of the Navier-Stokes equations : while Navier-Stokes non-linear equations allowed for a simulation of an incompressible fluid using partial derivatives, it only allowed to approximate the behavior of macroscopic fluid masses such as ocean currents or atmospheric air mass flow. [2] Navier-Stokes does not allow to take into account for instance gas compressibility, thermic effects, chemical reactions and many more. LB is a different, more modern approach at the mesoscopic level, directly inspired by the gas cellular automata : instead of using a set of discrete variables (representing fluid particles) evolving on a lattice, LB uses the population density of fluid particles on each site of the lattice. This allows for saving much computational space and obtaining a continuous model using mechanical statistics method to compute the final continuous model from the discrete one. [4]

2.1 LB : Formalism

To implement a simulation using LB, we first need to define a lattice. Each of the lattice sites has a population of fluid particles as well as directional discrete velocity. Each of the velocities will point to a neighboring site : for example, on a 2 dimensional lattice, each cell has 9 neighbors, including itself. This example is called a D2Q9 as illustrated in Figure 1. Each of the directions is assigned a weight w_i to incite each population to flow more or less in any given direction. The population on any given lattice point is called the Particle Density Function (PDF).

The PDFs will then evolve following the Lattice- Boltzmann equation [1] :

$$f_i(\mathbf{x} + \mathbf{e}_i \Delta t, t + \Delta t) - f_i(\mathbf{x}, t) = -\frac{\Delta t}{\tau} \left[f_i(\mathbf{x}, t) - f_i^{\text{eq}}(\mathbf{x}, t) \right] \tag{1}$$

where $f_i(\mathbf{x}, t)$ the particle distribution function in the i-th direction at position x and time t, $\mathbf{e}_i$ the velocity vector in the i-th direction, $\Delta t[s]$ the time step, $\tau[s^2/m]$ the relaxation parameter (we will

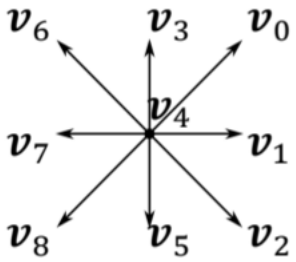

Figure 1: Directional velocity in a D2Q9 lattice, **Source :** [4]

come back to this) and $f_i^{\text{eq}}(\mathbf{x}, t)$ the equilibrium distribution function. We can define the equilibrium with the following equation [1] :

$$f_i^{\text{eq}} = w_i \rho \left[1 + \frac{\mathbf{e}_i \cdot \mathbf{u}}{c_s^2} + \frac{(\mathbf{e}_i \cdot \mathbf{u})^2}{2c_s^4} - \frac{\mathbf{u} \cdot \mathbf{u}}{2c_s^2} \right] \tag{2}$$

where w_i are the weights associated with the discrete velocities, $\rho[kg/m^3]$ the fluid density (covered in the next section), $\mathbf{u}[m/s]$ the fluid velocity and $c_s[m/s]$ the speed of sound in the lattice model ($c_s = \sqrt{\frac{1}{3}}$).

With the equations defined, we initialize the lattice with a set of PDFs and initial velocity. The algorithm will then follow at each iteration 3 main steps : collision, streaming and boundary conditions.

1. collision : at each lattice point, we update the PDFs according to equations 1 and 2.
2. streaming : each of the PDFs streams to the neighboring points, according to $\mathbf{e}_i$, w_i and $\mathbf{u}$.
3. boundary : each boundary of the lattice is defined as either cyclic (the fluid comes back around the opposite side), bounceback (the fluid bounces on a boundary, can be used to add obstacles), inlet (inflow of fluid, requires to impose a constant velocity and fluid density to simulate an input of flow) or outlet (outflow of fluid). We apply the boundary effect at each iteration.

2.2 LB : Parameters

The behavior of the fluid in the LB method is defined by a set of parameters :

1. initial velocity $\mathbf{u}_{ini}[m/s]$: used to first initialize the velocity of the lattice and set a constant inflow velocity if needed.
2. viscosity $\nu[m/s^2]$: it defines the fluidity of the fluid and the behavior of the fluid when computing the collision step based on the equation : $\nu = \frac{\mathbf{u}_{ini} * L}{Re}$ with $L[m]$ the characteristic length scale and Re the Reynolds number. Re will govern the type of flow (laminar, transitional, turbulent).
3. relaxation parameter $\tau[s^2/m]$: it governs how the fluid goes back to its equilibrium state at each collision step of each iteration. It is defined using the following the equation : $\tau = \frac{1}{3 * \nu + \frac{1}{2}}$.

Additionally, we can define the macroscopic variable of the system computed at each iteration and used during the collision step :

1. density $\rho[kg/m^3]$: derived directly from the PDFs using the formula : $\rho(\mathbf{x}, t) = \sum_i f_i(\mathbf{x}, t)$
2. velocity $\mathbf{u}(\mathbf{x}, t)$: obtained from the previously computed density : $\mathbf{u}(\mathbf{x}, t) = \frac{\sum_i f_i(\mathbf{x}, t) \mathbf{e}_i}{\rho(\mathbf{x}, t)}$

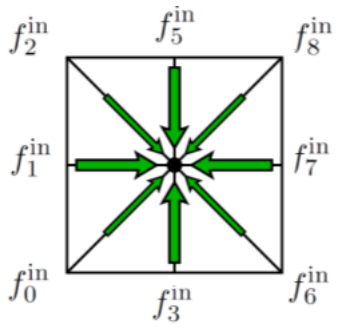

Figure 2: Directional input of a lattice unit, **Source :** [4]

3 Implementation

We implemented a full Lattice-Boltzmann simulation using Python coding language. The library implemented were **Numpy** for the calculation of the PDFs and **Matplotlib** for the renderings.

We defined an initial D2Q9 lattice of size 301x201. The weights ω_i have been initialized as standard. The initial velocity is $\mathbf{u}_i = 0.04[m/s]$ and the Reynolds number is $Re = 10.0$. The top and bottom of the lattice are defined as node-based bounceback, the right side is a simple outflow and the left side is an inflow based on the Zu-He method [5].

The Zu-He method is used to impose an input density and velocity for the unknown incoming PDFs on an inflow. In our case, we set the leftmost side of the lattice as the inflow. Referring to Figure 2, we need to compute the velocity and density of the sites noted as f_0^{in}, f_1^{in}andf_2^{in} (in this case, f_i^{in} being the PDFs before collision in the i-th direction, noted as f_i from now on). We define : $\rho_1 = f_0 + f_1 + f_2$, $\rho_2 = f_3 + f_4 + f_5$ and $\rho_3 = f_6 + f_7 + f_8$ (with f_4 being the center in Figure 2). Zu-He will then give us $\rho(\mathbf{x}, t) = \rho_1(\mathbf{x}, t) + \rho_2(\mathbf{x}, t) + \rho_3(\mathbf{x}, t)$ and $\rho(\mathbf{x}, t) * \mathbf{u}(\mathbf{x}, t) = \rho_1(\mathbf{x}, t) - \rho_3(\mathbf{x}, t)$. With $x = 0$ in our case, using Zu-He, we can compute : $\rho(t) = \frac{\rho_2(t)+2*\rho_3(t)}{1-\mathbf{u}(t)}$ and directly deduce the missing ρ_1.

The unknown velocities can be directly obtained from the initially defined velocity : $\mathbf{u}(\mathbf{x}, t) = \mathbf{u}_{ini}(\mathbf{x})$ where $\mathbf{u}_{ini}(\mathbf{x})$ is the initial velocity lattice.

4 System control

Once the system is fully defined, we executed a set of tests to monitor its behavior. All of the following tests have been made on the system described in section 3 over 30'000 iterations. Additionally, the system has been made cyclic at iteration 15'000 by removing the inflow and outflow from the lattice and freely letting the fluid flow to simulate a simple tube section. This transition (cut) is shown as a vertical dotted red line in the following Figures.

We started by monitoring the total particle population of the system as illustrated in Figure 3.
After the initial perturbation (up to iteration 5'000) we can see a first initial increase of population. This is a direct consequence of the Zu-He method. After the system is made cyclic, we can see that the particle population becomes constant so as expected there is no leak in the system.

Next, we monitored the input amount of the population compared to the output, illustrated in Figure 4. As for the previous test, there is an initial fluctuation before the system becomes stable. We can see before the cut the outflow being slightly shifted as compared to the inflow : this comes from the fact that the fluid has to travel through the tube before reaching its end, thus creating a gap between them. But as expected, the amount of fluid that comes in is the same as the amount that comes out proportional to the global increase of population induced by Zu-He. After the system is made cyclic, we can observe the inflow and outflow becoming the same, once again signifying no leak in the system. The tiny fluctuations appearing after the cut reflect the ripples of density that travel through

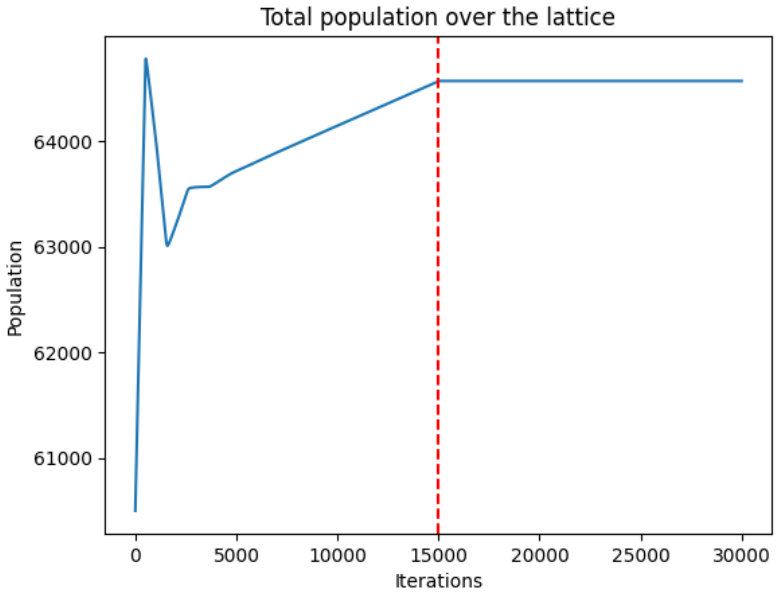

Figure 3: Population sum inside the lattice

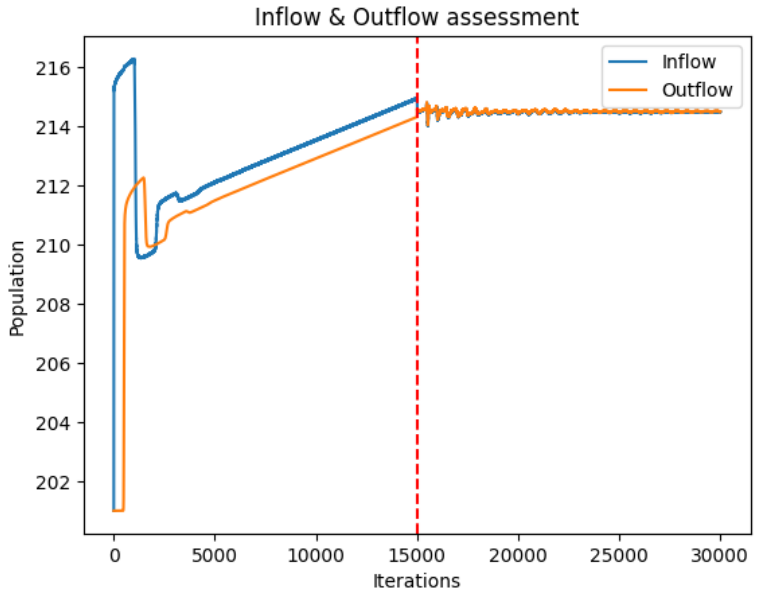

Figure 4: Inflow and Outflow populations

the system after is has been made recursive, before its complete stabilisation.

Another check we can perform is monitoring the momentum density $\mathbf{M}[kg * s/m^2] = \rho * \mathbf{u}$. To reflect a stable system with a steady-state flow, the momentum must be equal at each vertical slice of the lattice (the slices must be perpendicular to the fluid flow direction). We cut the lattice into three slices at three different x coordinates ($x = 50$, $x = 150$ and $x = 250$), computed the momentum and drawn the results on a graph represented in Figure 5. Once more, after the initial perturbations, we can see as expected a constant momentum across the lattice. After the recursivity is added in the system, there is once again some small fluctuations due to the ripple effect. We can additionally see a decrease in the momentum in the fluid : after we stop the external force (inflow and outflow), the fluid will naturally tend to go back by itself to its equilibrium state. The decrease of momentum reflects

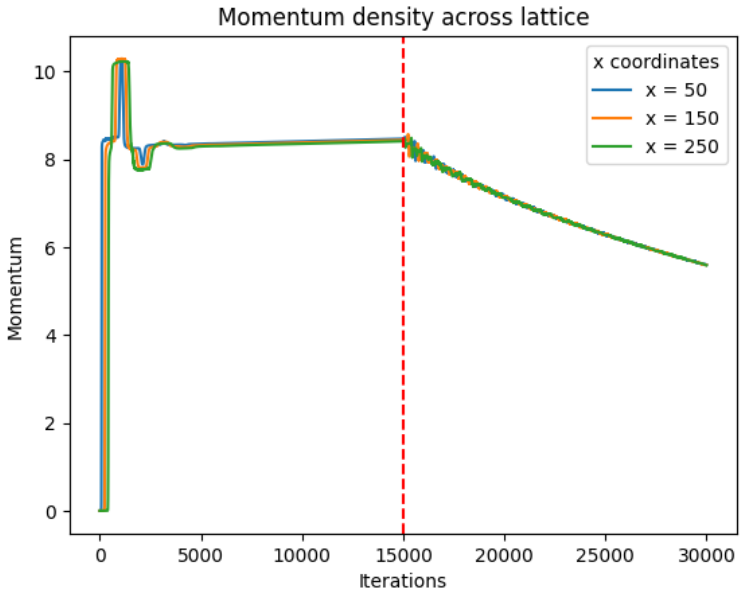

Figure 5: Fluid Momentum

this behavior. If left to run for enough time, we can expect to see the momentum reach 0.

Finally the last check we can do on our system is to look at the velocity profile. In a flowing tube, the velocity profile must be a parabola as defined with the following equation [3]:

$$u(r) = \frac{\Delta P}{4\nu L}(R^2 - r^2) \tag{3}$$

where $u(r)[m/s]$ the velocity with $r[m]$ the distance to the center of the tube radius-wise, $R[m/s]$ the radius of the tube, $\Delta P[Pa]$ the pressure gradient along the tube and ν, L defined as previously. The pressure gradient can be written for our needs as $\Delta P = c_s^2 * \Delta\rho$.

We drew the velocity profile using three values : the velocity profile of the simulation at the end of the 30'000 iterations, the theoretical velocity profile given with equation 3 and the expected velocity profile defined by the following equation :

$$u(r) = u_{max}(1 - (\frac{r}{R})^2) \tag{4}$$

with u_{max} the maximum velocity value taken directly from the simulation at the center of the tube at the last iteration. The resulting curves are shown on Figure 6. We can see that the computed and expected profiles are aligned as expected. We can see a slight deviation from the theoretical profile : this could be induced by the fact that the simulation may be stopped too soon and the system has yet to stabilize a bit more before reaching the values the theory dictates. Overall, we can see that the profile is a parabola, which confirms that the fluid flows correctly.

5 Next steps

Now That the system is functioning properly, the next is to make a more complex geometry : branching tubes with obstacles.

6 References

[1] Cyrus K. Aidun and Jonathan R. Clausen. "Lattice-Boltzmann Method for Complex Flows". In: *Annual Review of Fluid Mechanics* 42.1 (2010), pp. 439–472. DOI: 10.1146/annurev-fluid-121108-145519.

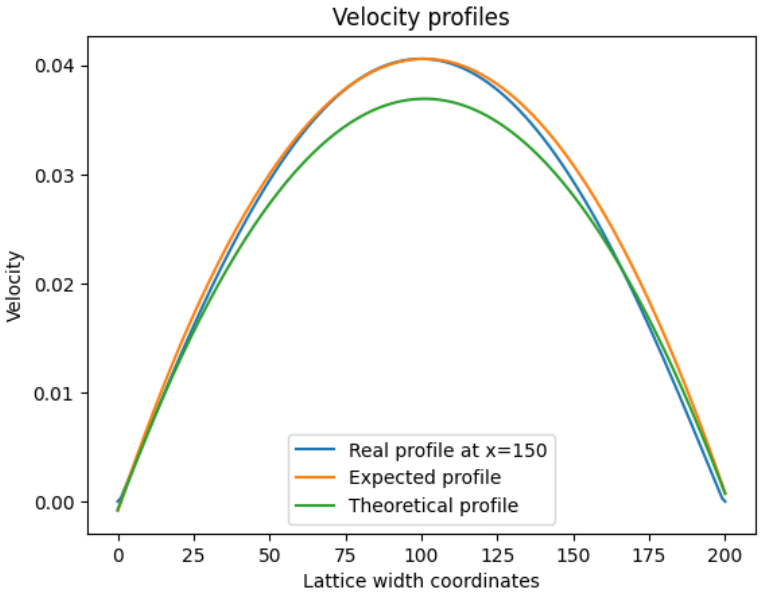

Figure 6: Computed, expected and theoretical velocity profiles

[2] S. Friedlander. *Encyclopedia of Mathematical Physics*. Jean-Pierre Françoise, Gregory L. Naber, Tsou Sheung Tsun, 2006.

[3] G. Hagen and J. L. M. Poiseuille. "Mémoire sur la viscosité des liquides". In: *Annales de chimie et de physique* 3.3 (1854), pp. 151–166.

[4] Jonas Lätt. "La méthode de Boltzmann sur réseau pour la simulation des fluides". In: ().

[5] L. Zu and Y. He. "A New ZU-HE Method for Fluid Inflow". In: *Journal of Fluid Dynamics* 30.2 (20XX), pp. 201–215.