High Performance Computing with Python

Final Report

Jonas Manser

4953222

jonas.burster@gmail.com

July 12, 2022

Contents

1 Introduction

The Lattice Boltzmann Method (LBM) is a numerical solution of (nonlinear) partial differential equations of the original BLT introduced in 1988 by McNamara and Zanetti [1]. It is used to simulate flows in a closed system and is based on the core assumption that flows can be approximated to particles on a lattice. This assumption has been shown to be true for incompressible subsonic flows of fluids and gases. Today, the LBM is used in a wide variety of fields from car aerodynamics to ocean current flows.

The LBM originates from the lattice gas automata (LGA) pioneered by Hardy, Pomeau and de Pazzis in the 1970s with the HPP-model [2]. This model could be used to simluate both gas and fluid flows, but did not not, as initially hoped by the authors, lead to the Navier-Stokes equation in the macroscopic limit. Later lattice gas automata models like the FPH-model [3] were able to satisfy the Navier-Stokes equation but were still plagued by many problens, like the lack of Galilean invariance [4] or the strong assumption that each node is surrounded by discrete particle cells, which resulted in massive computing requirements. It also assumed that streaming and collision happened synchronously for all nodes and thus the collision was non-deterministic.

In 1988 McNamara and Zanetti introduced the LBM as a direct alternative to the LGA [1]. Their new method "is based on the simulation of a very simple microscopic system, rather than on the direct integration of partial differential equations" [1]. Because of their close similarity the LBM shares many features with the LGA, like the lack of Galilean invariance but it also satisfies the Navier-Stokes equation in the macroscopic limit. It, crucially, "directly stud[ies] the time evolution of the mean values" [1] and thus does not need statistical averaging to compute the velocity as in LGA leading to lower computing requirements.

The key points of the LBM success is it's simplicity, relatively low consumption of computing requirements and easy parallelization of the algorithm. This is achieved by approximating the fluid to particles on a grid and using a separate streaming and collision step to simulate the particles behaviour over time. This is unlike other computational fluid dynamics (CFD) methods which directly solve the numerically macroscopic properties of a fluied, i.e. the mass, momentum, and energy. Using particles also makes incorperating boundries and microscopic interactions easier than in most other CFD models.

The reminder of the report will first introduce the theory behind the LBM and later present the results for each milestone.

2 Methods

2.1 Probability Density Function

The probability density function (PDF) describes the statistical probability of particles in a closed system not in equilibrium and is denoted by f. In this case, the PDF is given by $f(r_i, v_i, t)$ where r are the positions and v the velocities. The probability for finding a particle in a certain part of the phase space is then given by equation 1.

$$dP = f(\vec{r}, \vec{v}, t) d^3\vec{r} d^3\vec{v} \tag{1}$$

The phase space for equation is given by $[\vec{r}, \vec{r} + d\vec{r}, \vec{v}, \vec{v} + d\vec{v}]$. Thus, the probability for finding a particle in the phase space at position r_i is only depended on the velocity v_i and time t.

The probability of finding a single particle with an arbitraty place r and an arbitrary velocity v in the entire phase space is given by equation 2.

$$P = \int_{\Omega_{\vec{r}}} \int_{\Omega_{\vec{v}}} f(\vec{r}, \vec{v}, t) d^3\vec{r} d^3\vec{v} \quad \stackrel{!}{=} 1 \tag{2}$$

As we are in a closed system where no particles are added or destroyed we can assume that it must hold that equation is equal to one, i.e. normalized to unity.

This then leads us to the general form of the i-th moments of $f(\vec{r}, \vec{v}, t)$ w.r.t. $\vec{v}$ shown in equation 3.

$$\mu_i(\vec{r}) = \int_{\Omega_{\vec{v}}} \vec{v}^i f(\vec{r}, \vec{v}, t) d^3\vec{v} \tag{3}$$

The first and second order moments are the main subjects of this project and can be readily interpreted. The first order moment is the velocity in our system, which for liquids is the flow field. The second order moment is the kenetic energy density in our system which can be readily interpreted as the temperature of a fluid. Different fluids can have different translation ratios between velocities and temperature and thus an increase in temperature in a closed system would lead to higher velocities but the exact increase is depended on the fluid.

2.2 Boltzmann Transport Equation

The Boltzmann transport equation (BTE) tracks the time evolution of the probability distribution function and was published by Ludwig Boltzmann in 1872. To derive it we take the first order derivative of the PDF in respect to time as shown in equation 4.

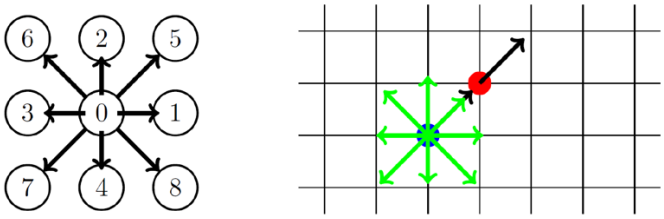
$$\frac{df}{dt} = \frac{\partial f}{\partial t} + \frac{dr(t)}{dt} \nabla_r f + \frac{dv(t)}{dt} \nabla_v f = \left(\frac{\partial f}{\partial t} \right)_{col} \tag{4}$$

Where the term f is the PDF, $\frac{dr(t)}{dt}$ is called the velocity and $\frac{dv(t)}{dt}$ is called the acceleration and by Newtons law is the force acting on the particles devivded by their respective mass. The *l.h.s.* of equation 4 is called the streaming term and the *r.h.s* is the collision term. To show an implementation of the streaming and collision terms the BTE first needs to be discretized.

2.3 Lattice-Boltzmann Method

The BTE is defined in a continous phase space which is not readily implementable in computer code. This can be overcome by approximating the continous phase space to a discrete phase space, called the Lattice-Boltzmann method (LBM). The idea is to discretize over a lattice and use the partial derivatives to solve the BTE. In this project the the spatial dimension is discretized to a 2D lattice and the velocities are descritized to 9 directions also called a D2Q9-model. This is illustarate in where a shows the discretazation of the velocity space and b the discretized of the physical space on a 2D grid with the velocity layered on top.

Figure 1: Discretization of the BTE.



(a) Discretization on the velocity space according to D2Q9.
(b) Uniform 2D grid for the discretization in the physical space.
(Material from lecture)

Streaming As shown in equation 1 streaming and collision are two separate terms. The collision can be ignored by setting it to zero, i.e. $\left(\frac{\partial f}{\partial t}\right)_{col} \stackrel{!}{=} 0$. This simplifies the BTE and implies the movement of particles in the vacuum with no mutual interaction between particles, see equation 5.

$$f_i(r + c_i \nabla t, t + \nabla t) = f_i(r, t) \tag{5}$$

The code wise implementation of a streaming function on a 2D lattice is shown in listing 1 [1].

[1] Throughout I will use the notation $_ixy$ where i is the rolling dimension and x and y are the deminsions of the physical space.

Listing 1: Implementation of the streaming term

```
def stream(f_cxy: np.array) -> np.array:
  for i in range(1, 9):
    f_cxy[i, :, :] = np.roll(f_cxy[i, :, :],
                             shift=C_CA[i],
                             axis=(0, 1))
  return f_cxy
```

Where f_{cxy} is the PDF and C_CA are the discretized velocity directions.

Collision The collision term is the *r.h.s* of the BTE 4 and represents the interaction between particles. It is a complicated two particle scattering integral but can be approximated by a relaxation time approximation τ so that the PDF locally relaxes to an equilibrium distribution $f_{eq}(r, v, t)$. The resulting discrete form of the BTE is shown in equation

$$f_i(r + \nabla t c_i, t + \nabla t) = f_i(r, t) + \omega \left(f_i^{eq}(r, t) - f_i(r, t)\right) \tag{6}$$

Because of the relaxation the PDF equilibrium is a local one and therefor depends on the local denisty ρ and the local average velocity u.

The local denisty is just a summation over the velocities of the PDF, i.e. $\rho(r) = \sum_i f_i$. An example implementation is shown in listing 2.

Listing 2: Implementation of the local density

```
def local_density(f_cxy: np.array) -> np.array:
  r_xy = np.einsum("cxy -> xy", f_cxy)
  return r_xy
```

The local average velocity is more complicated but effectively it is the summation over the velocity dimensions for each physical dimension divided by the local density. It is calculated via $u(r) = \frac{1}{\rho(r)} \sum_i c_i f_i(r)$ and respective code is shown in listing 3.

Listing 3: Implementation of the local average velocity.

```
def local_avg_velocity(f_cxy: np.array, r_xy: np.array):
  u_aij = np.einsum("ac, cxy->axy", C_CA.T, f_cxy) / r_xy
  return u_aij
```

With this we can define the PDF equilibrium function as in equation 7,

$$f_i^{eq}(\rho, u) = w_i \rho \left[1 + 3c_i * u + \frac{9}{2}(c_i * u)^2 - \frac{3}{2}|u|^2\right] \tag{7}$$

where w_i for a D2Q9 lattice can be defined as follow: $w_i = \left(\frac{4}{9}, \frac{1}{9}, \frac{1}{9}, \frac{1}{9}, \frac{1}{9}, \frac{1}{36}, \frac{1}{36}, \frac{1}{36}, \frac{1}{36}\right)$. It is important to note that the approximation in the collision term is not always accurate. But under the assumption that all transport processes occur on a longer time scale it is appropriate to do.

3 Shear Wave Decay and Kinematic Viscosity

Shear wave decay describes the amplitude decay of shear waves, elastic waves travelling through the body of an object.

are a type of elastic wave and are one of the two main types of elastic body waves, so named because they move through the body of an object, unlike surface waves.[1]

In this milestone we will take a look at a method commonly used in computational physics to measure the kinematic viscosity of a fluid. The shear wave method consists in setting a sinusoidal velocity profile in our box and then measuring how fast it decay

.

References

[1] Guy R McNamara and Gianluigi Zanetti. Use of the boltzmann equation to simulate lattice-gas automata. *Physical review letters*, 61(20):2332, 1988.

[2] J Hardy, Yves Pomeau, and O De Pazzis. Time evolution of a two-dimensional model system. i. invariant states and time correlation functions. *Journal of Mathematical Physics*, 14(12):1746–1759, 1973.

[3] U. Frisch, B. Hasslacher, and Y. Pomeau. Lattice-gas automata for the navier-stokes equation. *Phys. Rev. Lett.*, 56:1505–1508, Apr 1986.

[4] XB Nie, Xiaowen Shan, and H Chen. Galilean invariance of lattice boltzmann models. *EPL (Europhysics Letters)*, 81(3):34005, 2008.