PENNSTATE

1855

Lattice Boltzmann Method

Presented by:
Keith Doyle
Liwei Li
Chaoyi Wang
Cooper Elsworth

Introduction & History

Let's get to know LBM

Introduction

- Two extreme scales for modeling fluid flow (Mele 2013)

  - Macro-Scale
    - Uses PDE Equations such as Navier Stokes equation
    - Normally solved numerically using FDM, FEM, or FVM

  - Micro-Scale
    - Models individual molecules
    - Behavior governed by Hamilton's equation
    - There are too many molecules to practically model virtually anything useful

Introduction

- LBM splits the gap between these two scales
  - Considers a collection of molecules as a unit
  - Able to accurately model macro-scale behavior by considering average behavior of these collections of molecules
  - Behavior governed by Boltzmann equation

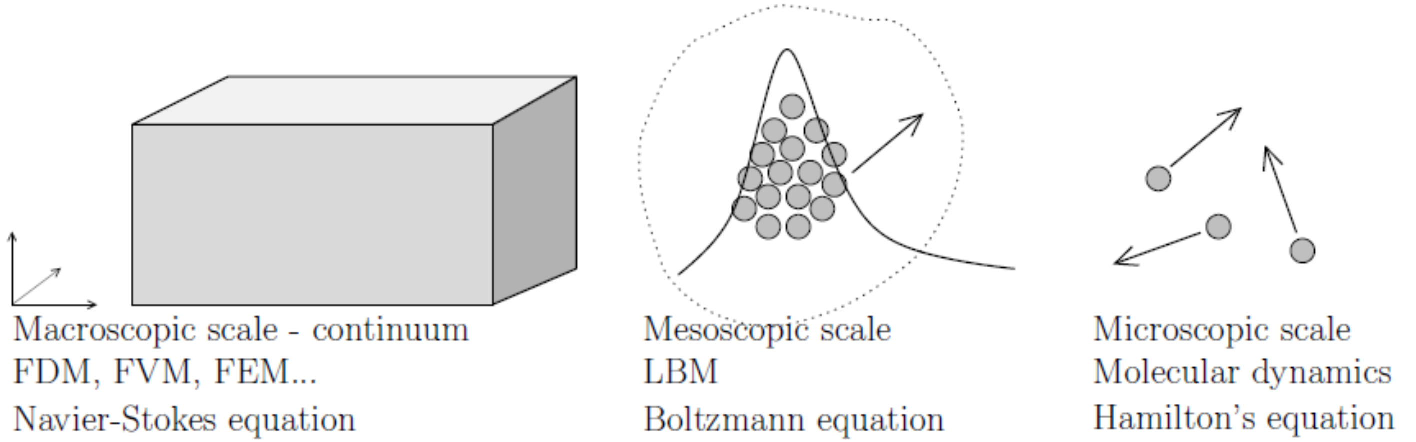


Image : Mele, 2013

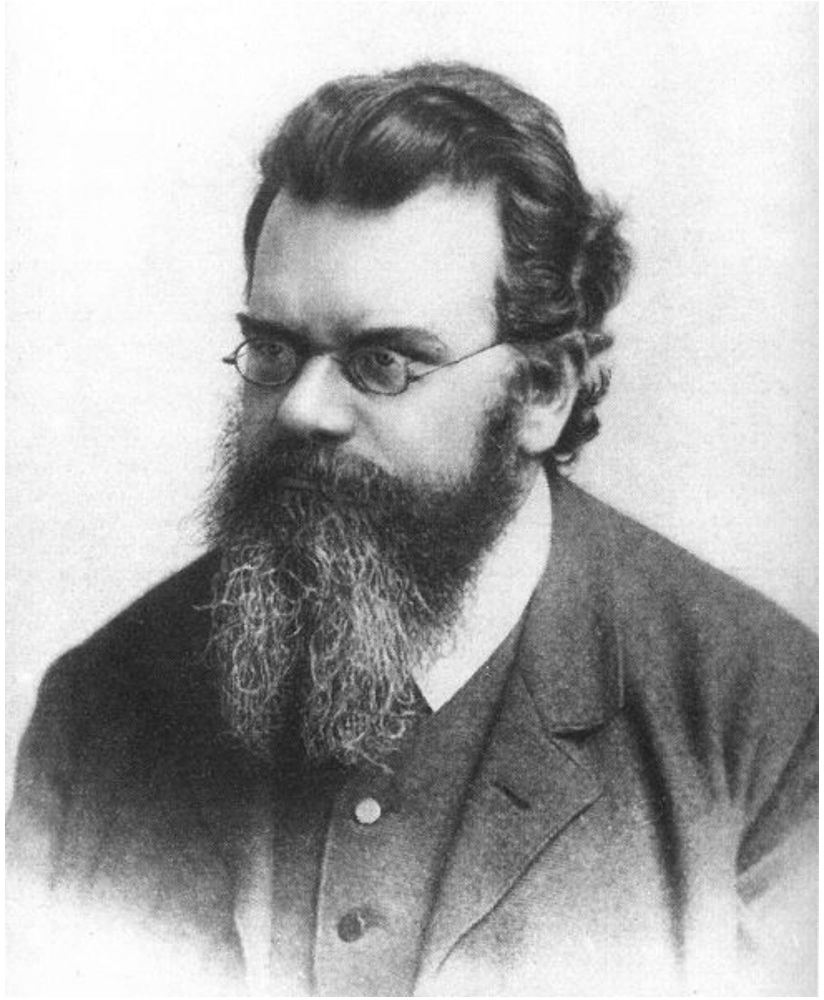
Introduction

- Advantages
  - LBM solved locally so it is easy to break the problem into calculations that can be done in parallel by multiple computer processers (Mele 2013).
  - Meshing is quasi-instantaneous and computationally simple

- Disadvantages
  - Difficult to simulate scenarios with a high Mach numbers
  - Thermo-hydrodynamic scheme is absent

Historical Perspective

- Boltzmann Equation (1800's)
  - Developed by Ludwig Boltzmann
  - Describes the dynamics of an ideal gas
  - The Lattice Boltzmann Equation, which governs behavior in the LBM, is a discretized form of the Boltzmann Equation



Historical Perspective

- Lattice Gas Automata
  - Precursor to LBM
  - Developed by Hardy, Pomeau, and de Pazzis in the 1970's
  - Initially was widely praised as a revolutionary technique.
  - Featured on front page of Washington Post on November 19, 1985
  - Problems with LGA led to the need for the development of LBM

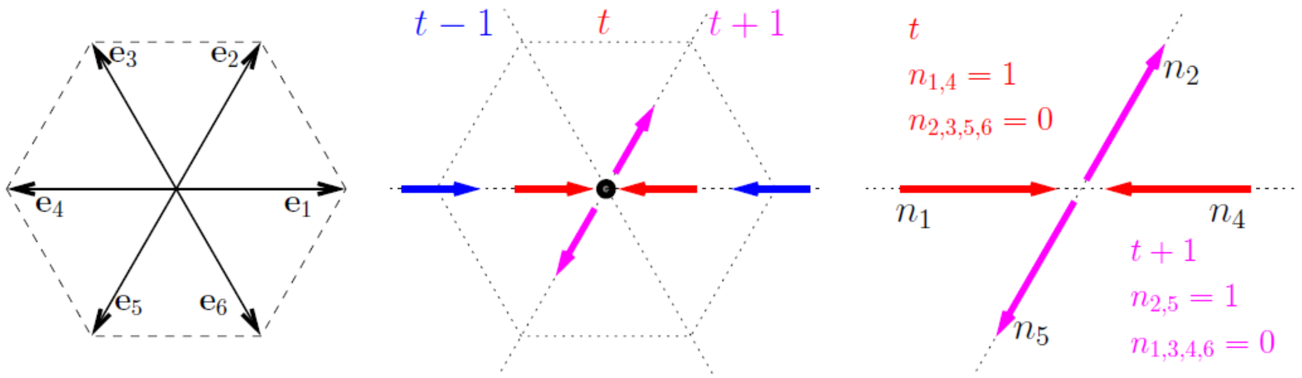


Image : Mele, 2013

Historical Perspective

- Lattice Gas Automata
  - Disadvantages
    - Statistical noise
    - Needs to simulate a large number of particles in order to reach an acceptable solution
    - Computationally inefficient due to its discrete state calculations

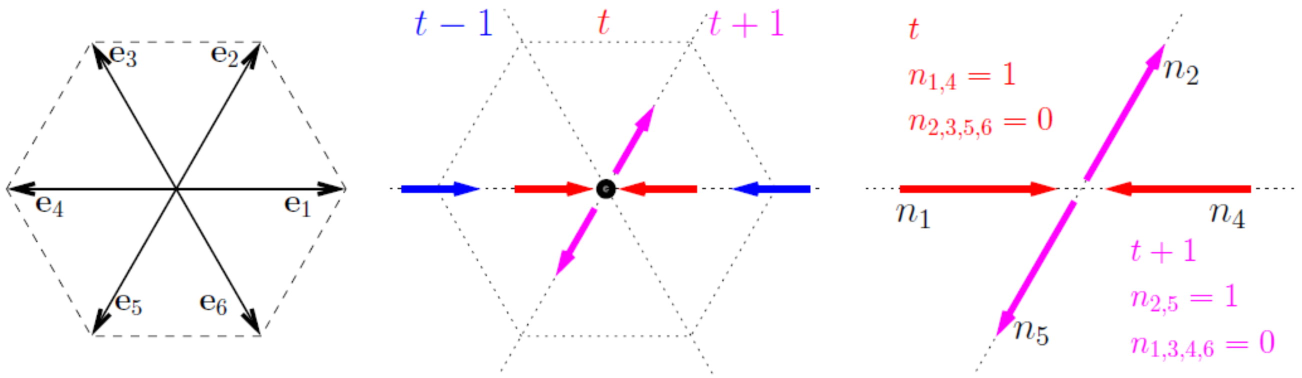


Image : Mele, 2013

Historical Perspective

- Lattice Boltzmann Method
  - Developed incrementally in the 1980's
  - Overcomes statistical noise associated with LGA by replacing boolean particle occupation variables with single particle distribution functions
  - Distribution functions are an averaged quantity, so there is no need to average the state of a large quantity of cells to define macroscopic behavior

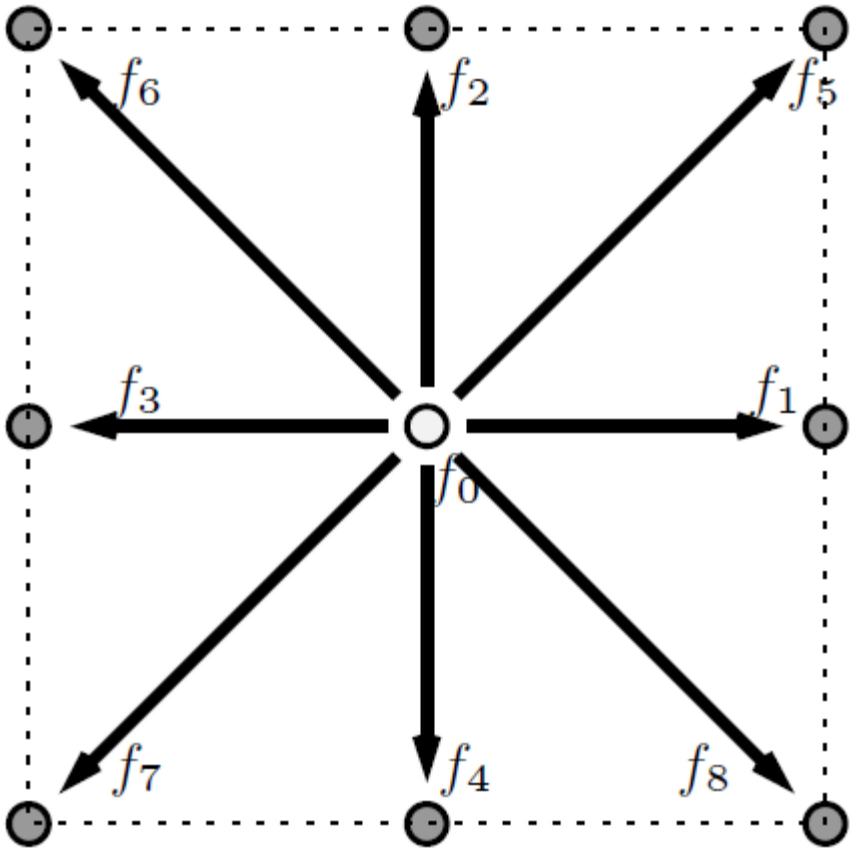


Image : Mele, 2013

General Principle

Essence of Lattice Boltzmann Method

PENNSTATE 1855

General Principles

- LGA (Lattice Gas Automata)
- Lattice Boltzmann Equation
- Collision and Streaming Stages
- LBM vs. CFD (Computational Fluid Dynamics)
- Validation of LBM

Lattice Gas Automata (LGA)

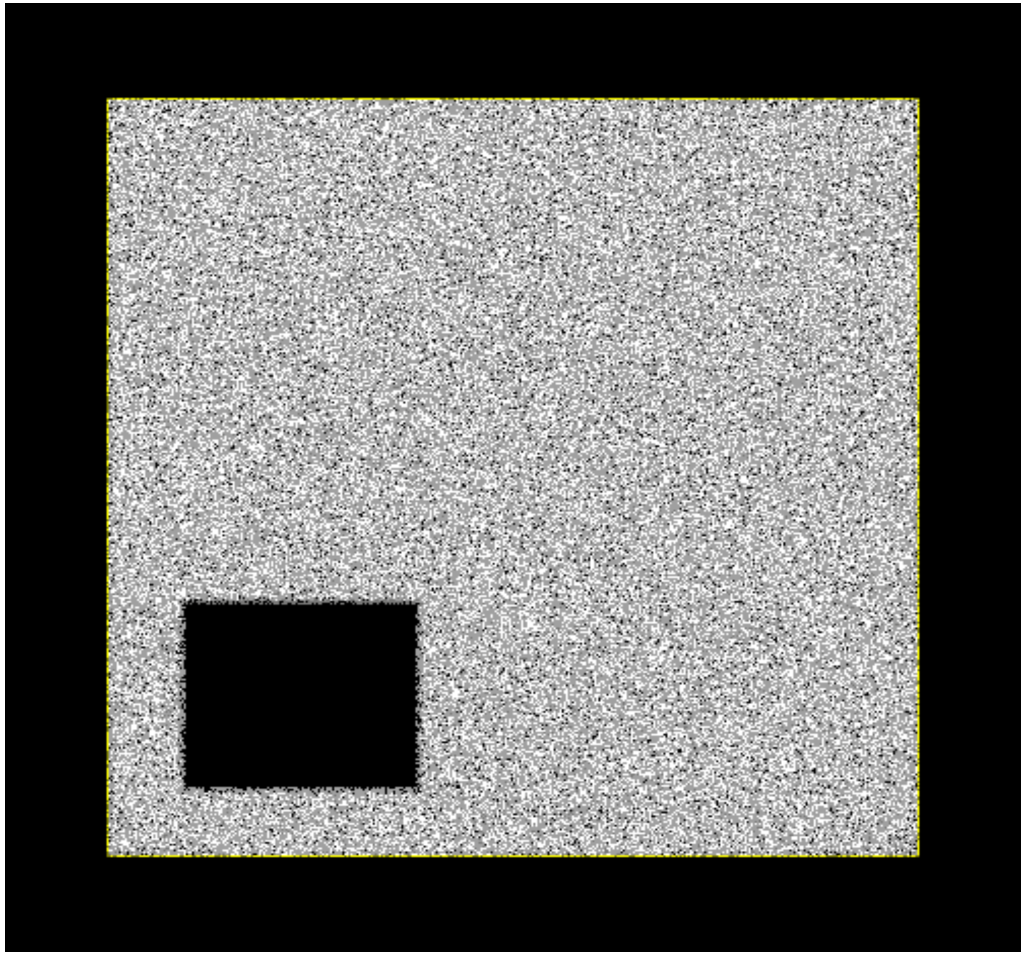
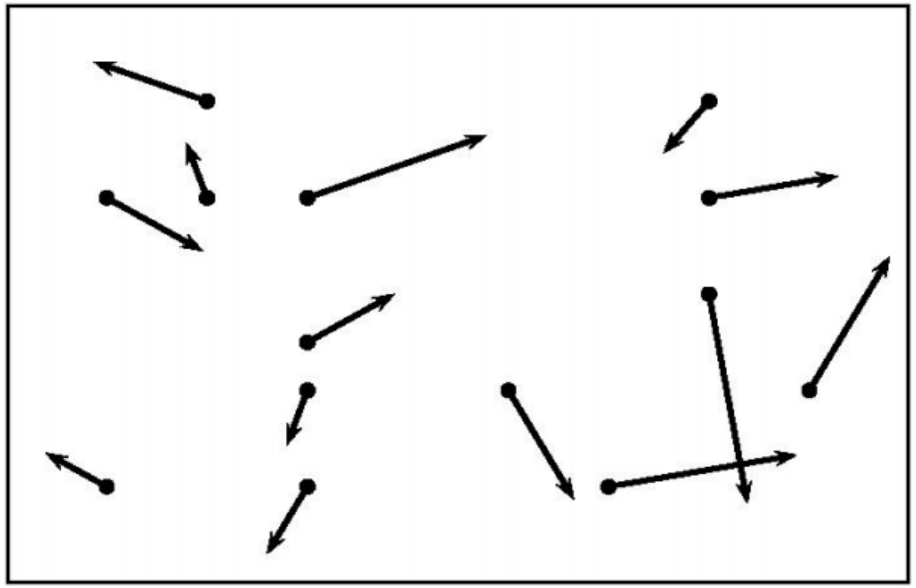


Figure from wiki

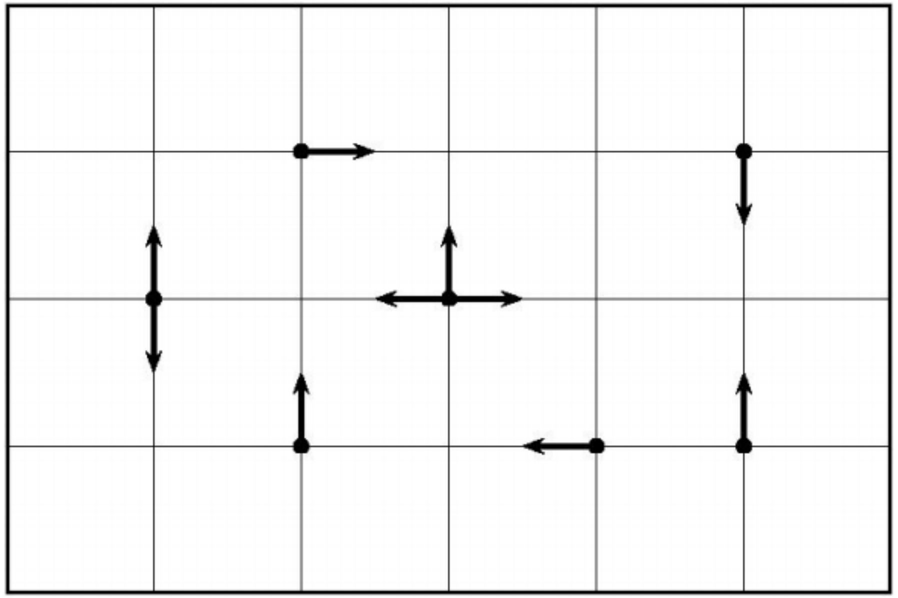
- Originated from early 1990's.
- Lattice Automaton used to simulate fluid flows
- Comprises of a lattice with different states on sites.
- Lattice Gas: states are represented by particles with certain velocities.
- State at each site is purely boolean: there either *is* or *is not* a particle travel in each direction.
- Evolution is done by two steps in each time step: streaming and collision
- Precursor to Lattice Boltzmann Method

Microscopic Dynamics

Microscopic particles inside fluids



Fictitious particles moving along lattice links

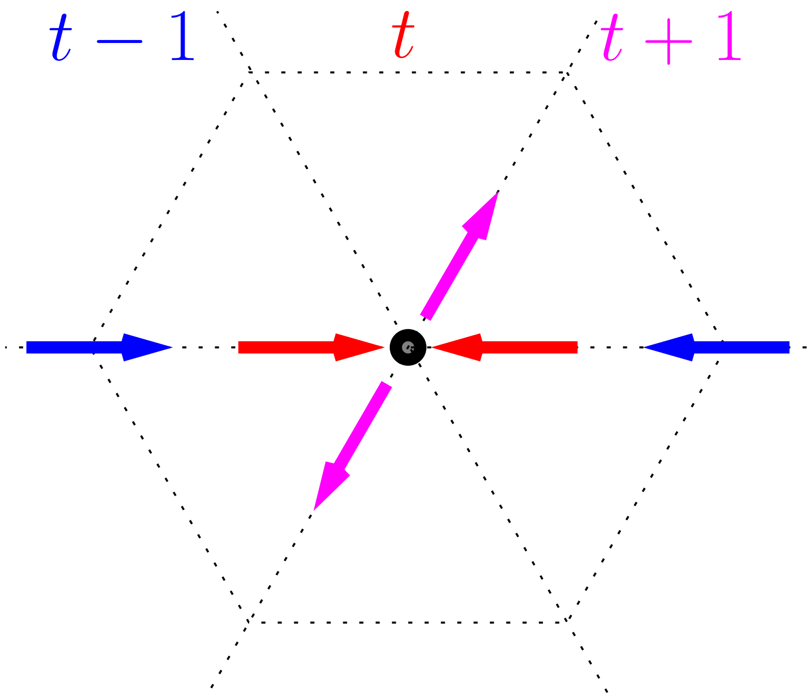


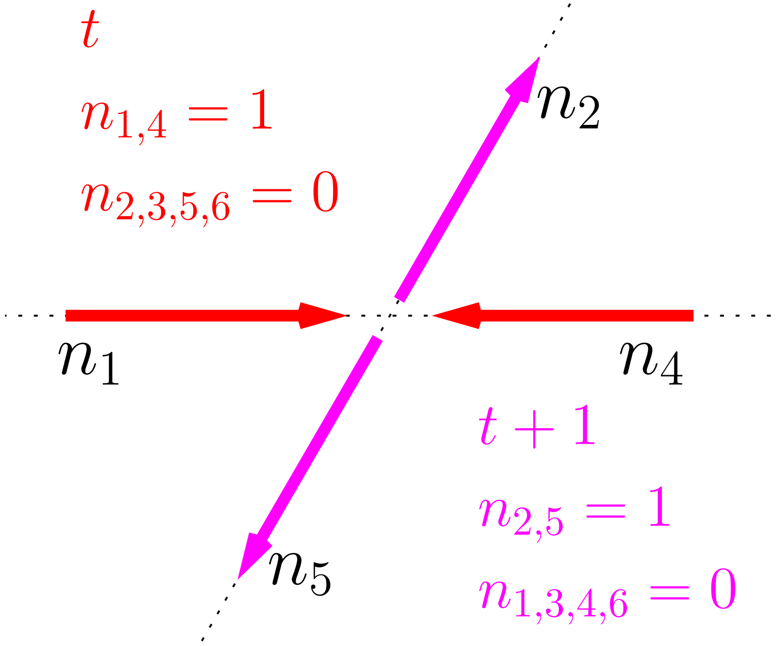
Lattice Gas Automata (LGA)

Evolution Equation

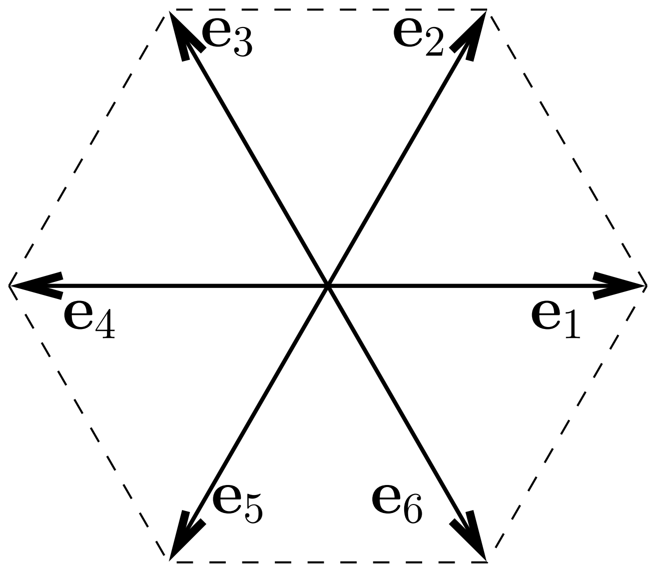
$$n_i\left(x+e_i\delta t, t+\delta t\right)=n_i\left(x,t\right)+\Omega_i\left(n\left(x,t\right)\right)$$

Collision term
-1,0,1





Lattice Gas Automata (LGA)



FHP model
First introduced in 1986 by
Frisch, Hasslacher and Pomeau

- Consider a set of boolean variables:

$$n_i(x,t), \quad i = 0,1,\ldots M$$

- Particle presentation

$$n_i(x,t) = 0$$ No particles at site x and time t

$$n_i(x,t) = 1$$ A particle is present at site x and time t

- Collision rules

$$\Omega_i(n(x,t)) = -1, 0, 1$$

  - Mass conservation
  - Momentum conservation

Lattice Boltzmann Equation

LGA $$n_i\left(x+e_i\delta t, t+\delta t\right)=n_i\left(x,t\right)+\Omega_i\left(n\left(x,t\right)\right)$$

Probability distribution function

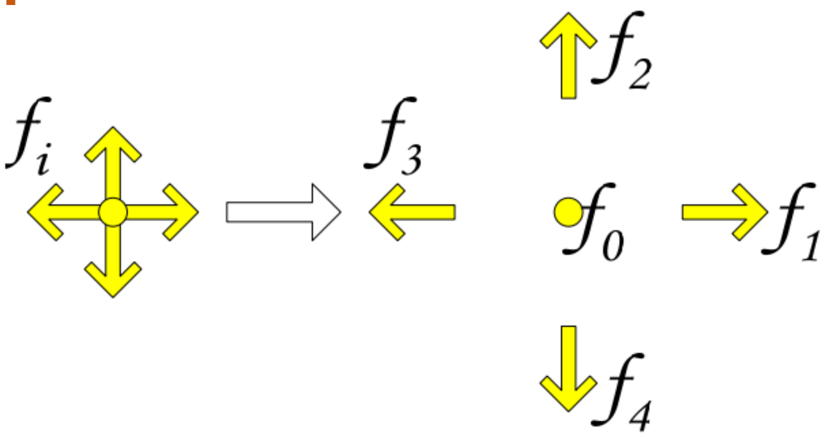
$$f_i=f_i(x,\xi,t)$$

LBM $$f_i\left(x+e_i\delta t, t+\delta t\right)=f_i\left(x,t\right)+\Omega_i\left(f\left(x,t\right)\right)$$

Macroscopic Properties

Probability distribution function

$$f_i = f_i(x, \xi, t)$$

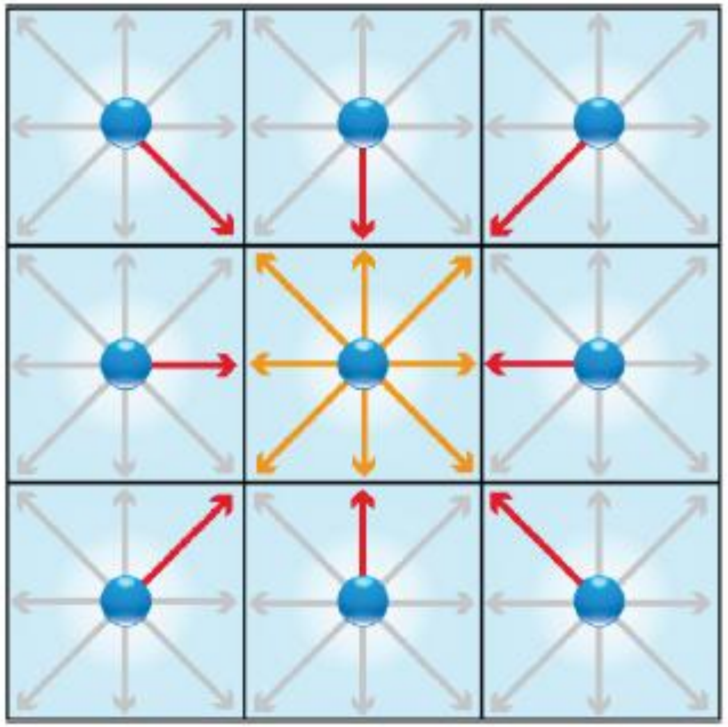


Flow properties easily computed from particle distribution values per time step

$$\rho = \sum_i f_i \qquad u = \frac{\sum_i f_i e_i}{\rho} \qquad \nu = \frac{2\tau - 1}{6}$$

Streaming on Lattice

A D2Q9 Lattice Model



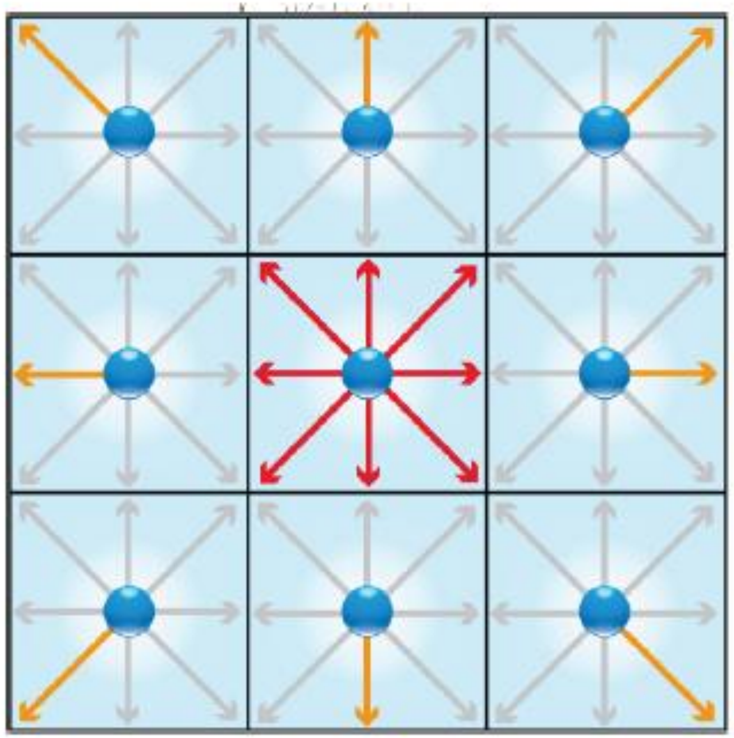


Image from Indo-German winter academy 2011

Collision Computation

Bhatnagar-Gross-Krook (BGK) collision operator for equilibrium

$$\Omega_i = -\tau^{-1}\left(f_i(x,t) - f_i^{EQ}(\rho,u)\right)$$

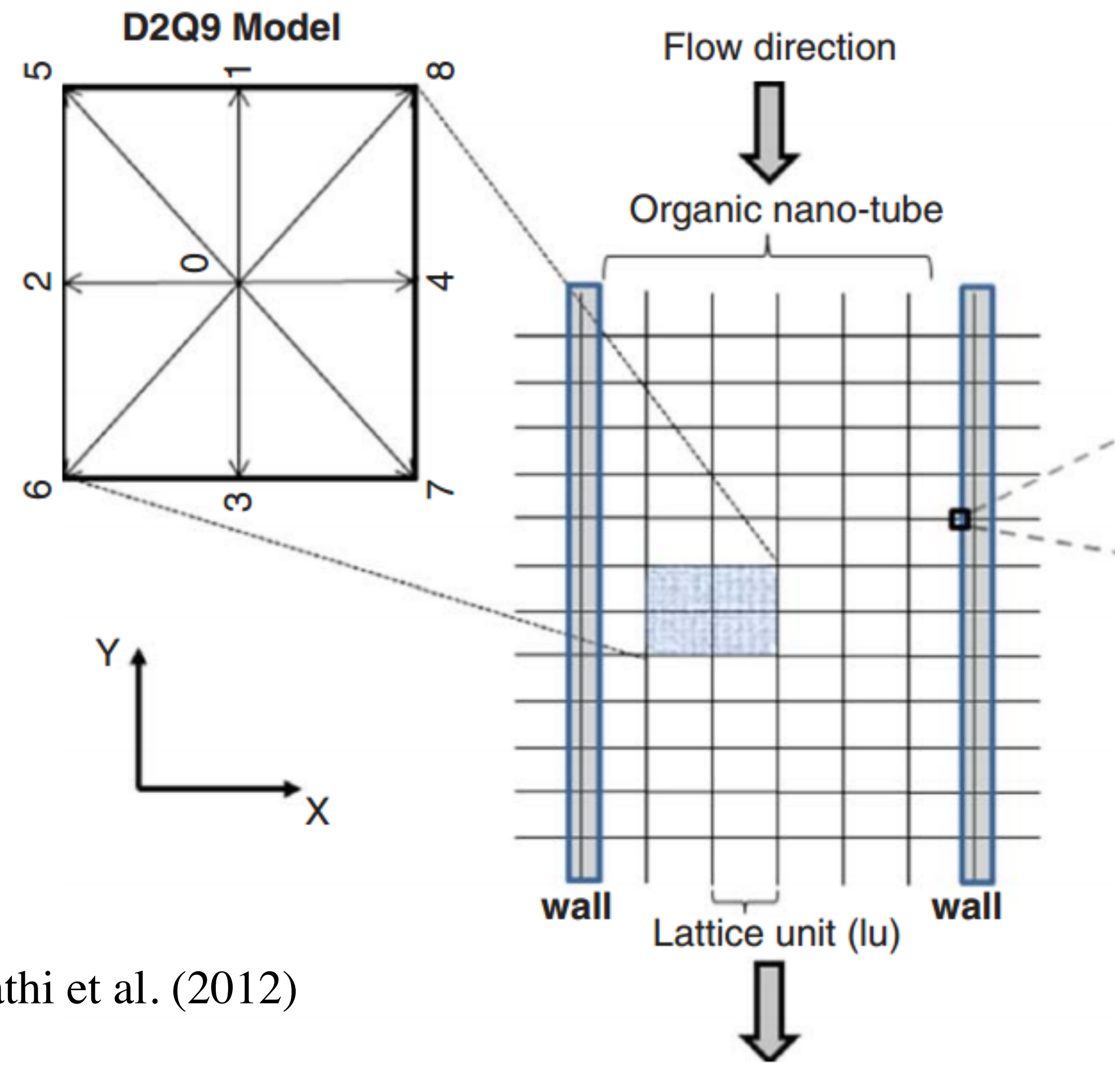
where, τ is the relaxation time

$$f_i^{EQ}(\rho,u) = \rho(A + B(e_i \cdot u) + Cu^2 + D(e_i \cdot u)^2)$$

where, - A, B, C, D are constants defined by lattice geometry

Boundary Handling

Microscopic Numerical Fluid Solver

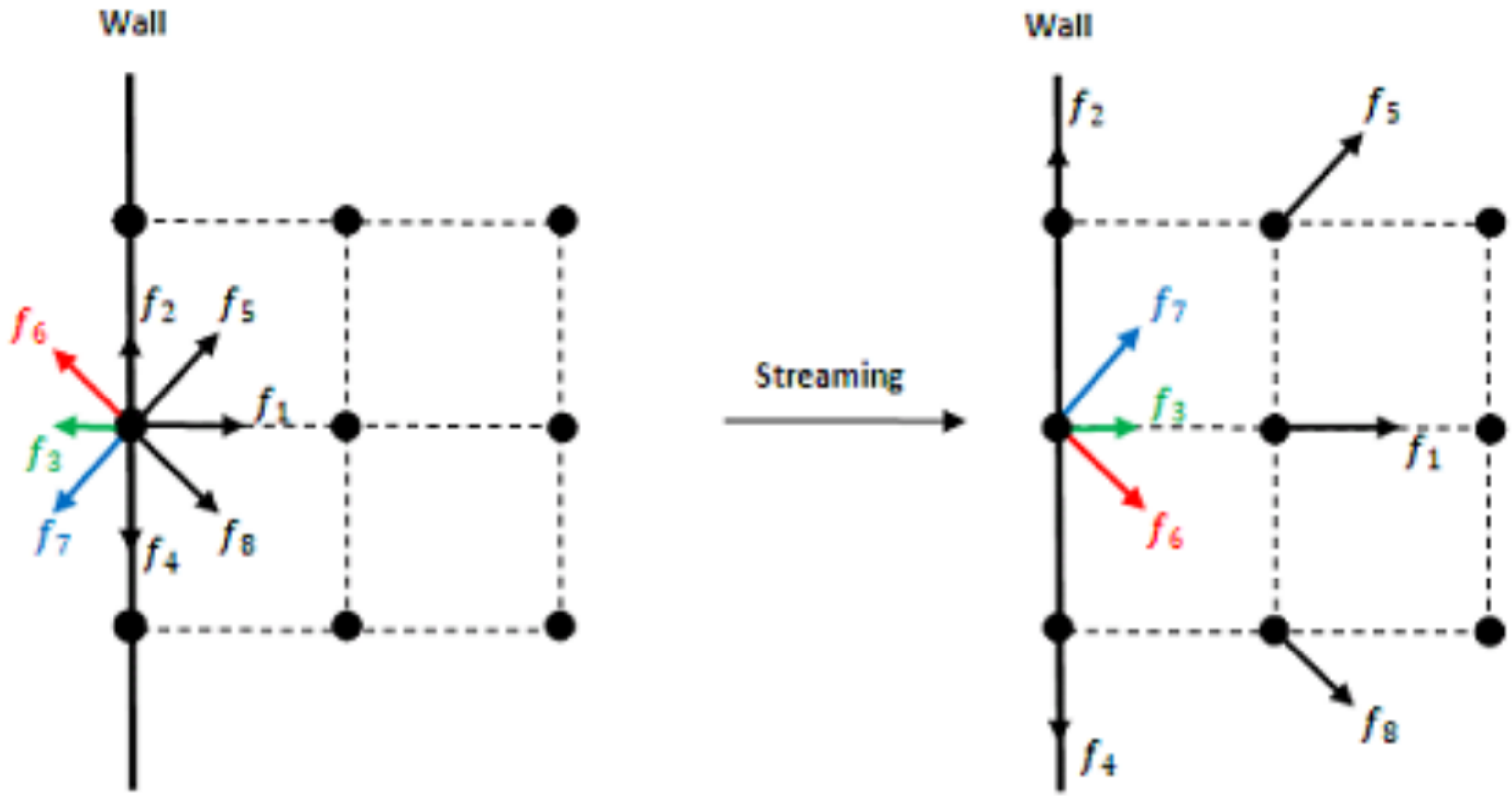


Undefined?

Figure from Fathi et al. (2012)

Bounce-Back Method

$$f_{-i}(x,t+1) = f_i(x,t)$$



Algorithm of LBM

Initialize ϱ,u f_i and f_i^{eq}

$$\rho = \sum_{\alpha} f_{\alpha}$$

$$u = \frac{1}{\rho}\sum_{\alpha} \xi_{\alpha} f_{\alpha}$$

Streaming step: move f_i to f_i^* in the direction of ξ_i

$$f\left(x+\xi\delta_t,\xi,t+\delta_t\right)=f\left(x,\xi,t\right)$$

Compute **macroscopic** ϱ and u

$$\rho = \sum_{\alpha} f_{\alpha}$$

$$u = \frac{1}{\rho}\sum_{\alpha} \xi_{\alpha} f_{\alpha}$$

Compute $\mathbf{f_i^{eq}}$

$$f^{EQ} = \rho w_{\alpha}\left[1+\frac{\xi\cdot u}{RT}-\frac{u\cdot u}{2RT}+\frac{(\xi\cdot u)^2}{2(RT)^2}\right]$$

Collision step: update distribution function f_i

$$f\left(x+\xi\delta_t,\xi,t+\delta_t\right)-f\left(x,\xi,t\right)=-\frac{1}{\tau}\left[f\left(x,\xi,t\right)-f^{EQ}\left(x,\xi,t\right)\right]$$

LBM vs. CFD

Conventional CFD Method

Construction of fluid equations
Navier-Stokes equation
2nd-order PDE, nonlinear convective term

Discrete approximation of PDE
Finite difference, finite element, etc

Numerical integration
Solve the equations on a given mesh and apply PDE boundary conditions

Lattice Boltzmann Method

Discrete formulation of kinetic theory
Lattice Boltzmann equation
1st-order PDE, simple advection

No further approximation
The equations are already in discrete form

Numerical integration
Solve on lattices and apply kinetic based BC

Simple conversion to fluid variables
These are theoretically shown to obey the required fluid equations

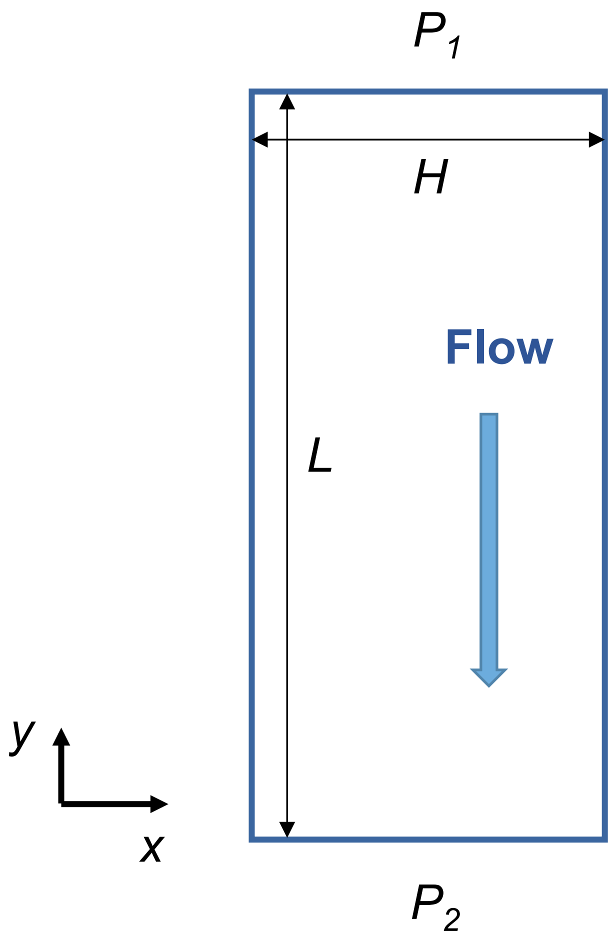
Steady Poiseuille Flow

Navier-Stokes equation for incompressible flow

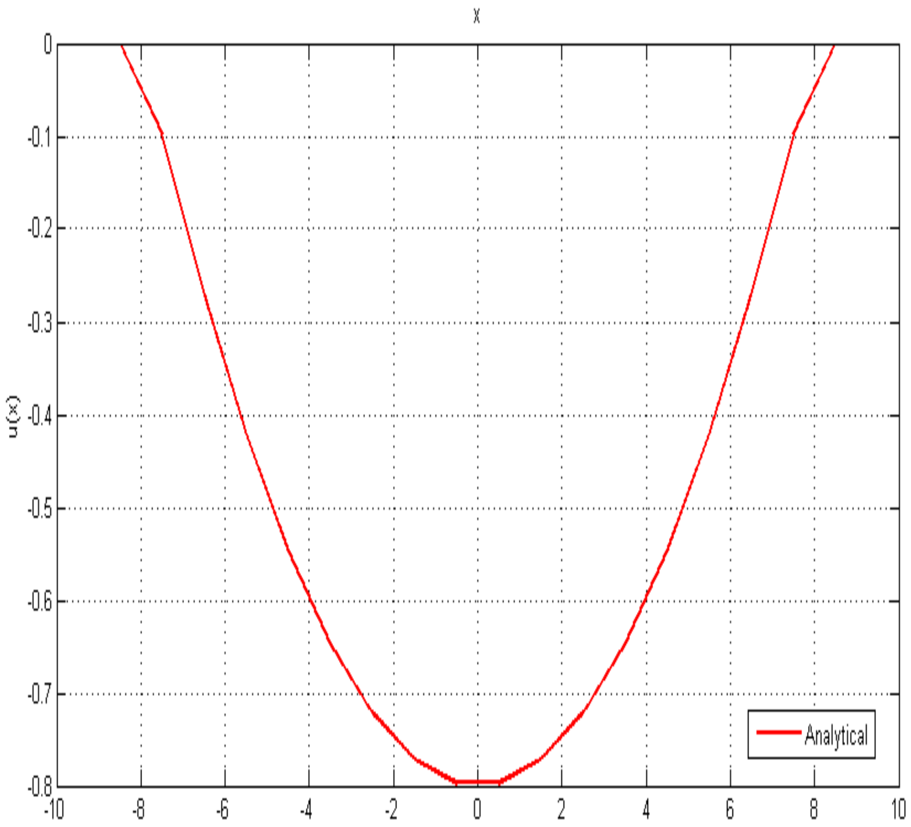
$$\mu \frac{\partial^2 u}{\partial x^2} = \frac{\partial p}{\partial y}$$

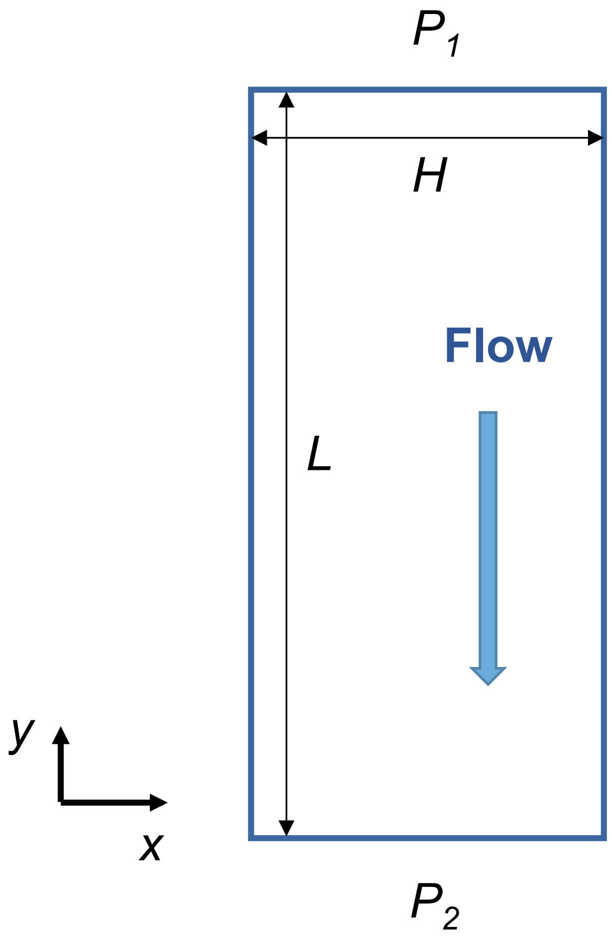
Available exact analytical solution

$$u(x) = \frac{\Delta P}{2\mu L} x(x - H)$$

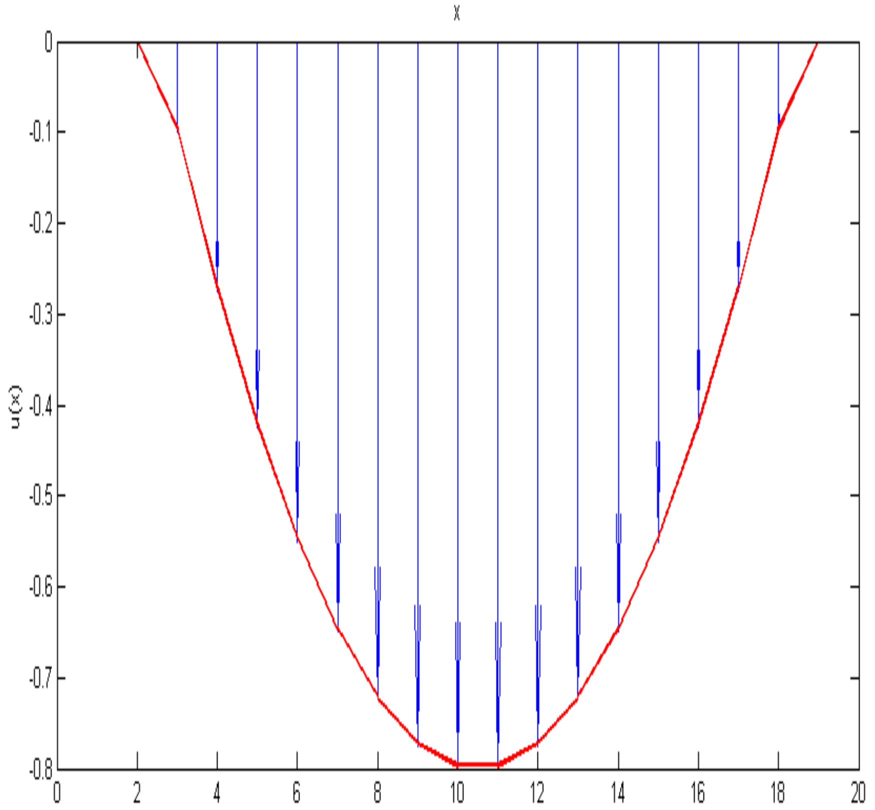


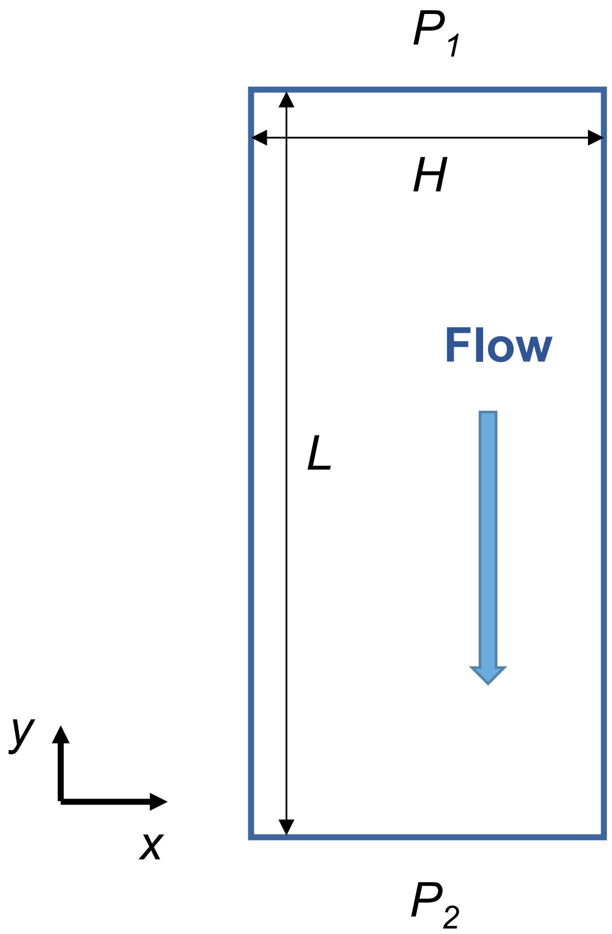
Analytical Fluid Parabolic Velocity Profile





Validation of the LBM





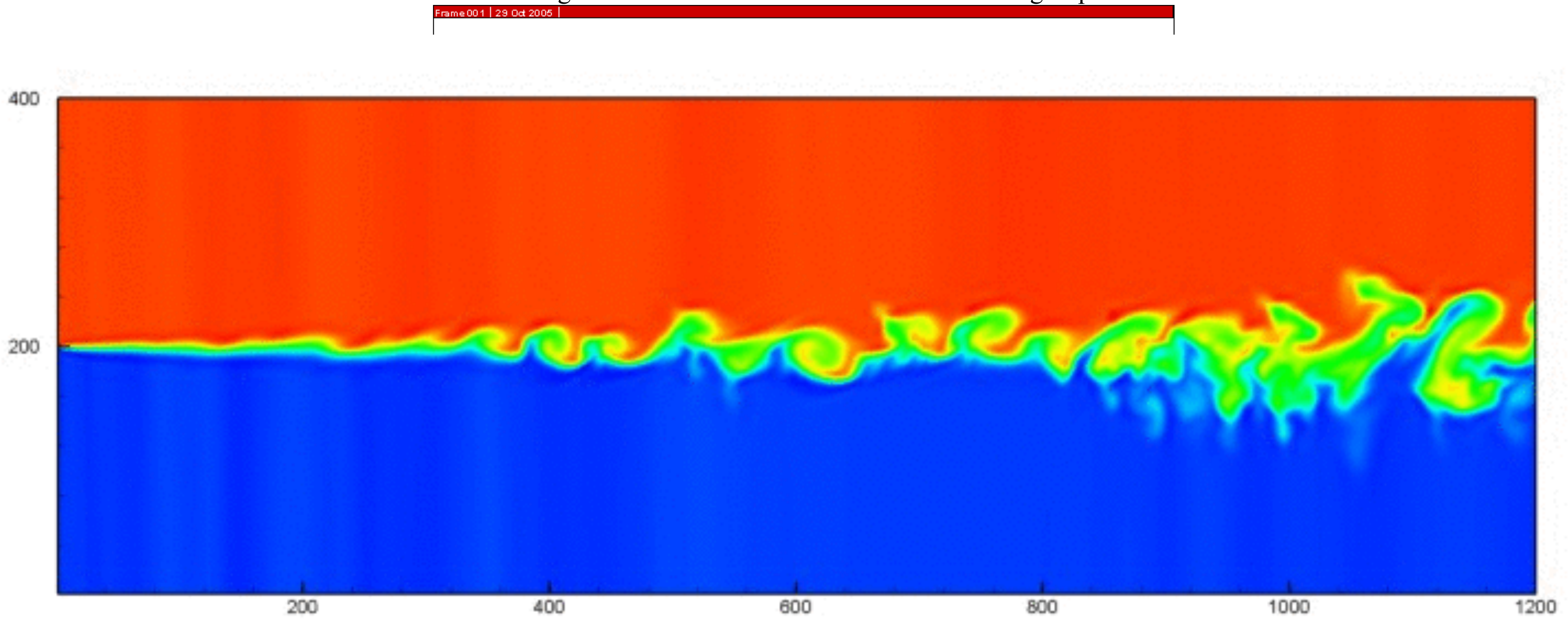
PENNSTATE

1855

Lattice Boltzmann Equation

- We can solve varies Fluid Dynamics problems with LBM

Image from nus.edu & combustion fundamental group



Simulation of turbulent mixing in a binary mixture

Fish motion simulation by LBM

Governing Equation of LBM

Manipulation & Interpretation

Lattice Boltzmann Equation

We start from general Boltzmann Equation

$$\frac{\partial f}{\partial t} + \frac{p}{m} \cdot \nabla f + F \cdot \frac{\partial f}{\partial p} = \left(\frac{\partial f}{\partial t} \right)_{coll} \quad ---(1)$$

In which:

- f is a particle distribution function
- F is external force field acting on the particle
- m is particle mass
- p is particle momentum
- t is time

Lattice Boltzmann Equation

To derive LB equation, assume zero force field
Also note that momentum over mass is particle velocity
Together with (1) yields:

$$\frac{\partial f}{\partial t} + \vec{\xi} \cdot \nabla f = \left(\frac{\partial f}{\partial t}\right)_{coll} \quad ---(2)$$

In which:

- ξ is microscopic velocity $\frac{p}{m} = \xi$

$$\left(\frac{\partial f}{\partial t}\right)_{coll} = ?$$

Collision term is usually complex and hard to interpret

Collision term is usually approximated using Bhatnagar-Gross-Krook (BGK) collision operator

$$\Omega_i = -\tau^{-1}\left(n_i - n_i^{EQ}\right)$$

In which:

- Ω is the collision term
- τ or λ is a relaxation time representing the amount of time it consumed to return to equilibrium state.
- n or f is the particle distribution function
- n^{EQ} or g is the distribution function in equilibrium state.

$$\left(\frac{\partial f}{\partial t}\right)_{coll} = -\frac{1}{\lambda}(f - g)$$

Lattice Boltzmann Equation

Assemble BGK collision term with LHS yields the general Lattice Boltzmann Equation:

$$\frac{\partial f}{\partial t} + \xi \cdot \nabla f = -\frac{1}{\lambda}(f - g)$$

in which:

- f is the single particle distribution function.
- ξ is the microscopic velocity vector
- λ is the relaxation time due to collision
- g is the Boltzmann-Maxwellian distribution function.

$$g \equiv \frac{\rho}{(2\pi RT)^{D/2}} \exp\left(-\frac{(\xi - u)^2}{2RT}\right)$$

in which:

- D is the dimension of space
- R is the ideal gas contant
- ρ, T and u are the macroscopic density of mass, temperature and velocity respectively. They are moments of distribution function f.

Lattice Boltzmann Equation

Compute macroscopic quantities (moments of distribution function f)

$$\rho = \int f\,d\xi = \int g\,d\xi$$

$$\rho u = \int \xi f\,d\xi = \int \xi g\,d\xi$$

$$\rho \varepsilon = \frac{1}{2}\int (\xi - u)^2 f\,d\xi = \frac{1}{2}\int (\xi - u)^2 g\,d\xi$$

Macroscopic quantities can be represented by integrating the distribution function in proper order

That's the beauty of LBM

Discretized LB Equation

Chapman-Enkog assumption

$$\int h(\xi) f(x,\xi,t) d\xi = \int h(\xi) g(x,\xi,t) d\xi$$

$$h(\xi) = A + B \cdot \xi + C\xi \cdot \xi$$

in which:

- A and C are arbitrary constants, B is an arbitrary constant vector

By writing LB equation in an ODE form and implementing Chapman-Enkog assumption

We can discretize LB equation in time

$$\frac{df}{dt} + \frac{1}{\lambda} f = \frac{1}{\lambda} g$$

$$\frac{d}{dt} \equiv \frac{\partial}{\partial t} + \xi \cdot \nabla$$

The Equation can be formally integrated over time step δ_t

Discretized LB Equation

$$f(\boldsymbol{x}+\boldsymbol{\xi}\delta_t,\boldsymbol{\xi},t+\delta_t)=\frac{1}{\lambda}e^{-\delta_t/\lambda}\int_0^{\delta_t}e^{t'/\lambda}g(\boldsymbol{x}+\boldsymbol{\xi}t',\boldsymbol{\xi},t+t')dt'$$
$$+e^{-\delta_t/\lambda}f(\boldsymbol{x},\boldsymbol{\xi},t).$$

Assuming that δ_t is small enough and g is smooth enough locally, the following approximation can be made:

$$g(\boldsymbol{x}+\boldsymbol{\xi}t',\boldsymbol{\xi},t+t')=\left(1-\frac{t'}{\delta_t}\right)g(\boldsymbol{x},\boldsymbol{\xi},t)$$
$$+\frac{t'}{\delta_t}g(\boldsymbol{x}+\boldsymbol{\xi}\delta_t,\boldsymbol{\xi},t+\delta_t)$$
$$+O(\delta_t^2),\quad 0\leqslant t'\leqslant\delta_t\,.$$

The leading terms neglected in the above approximation are of the order of $O(\delta_t^2)$. With this approximation, Eq. (8) becomes

$$f(\boldsymbol{x}+\boldsymbol{\xi}\delta_t,\boldsymbol{\xi},t+\delta_t)-f(\boldsymbol{x},\boldsymbol{\xi},t)$$
$$=(e^{-\delta_t/\lambda}-1)[f(\boldsymbol{x},\boldsymbol{\xi},t)-g(\mathbf{x},\boldsymbol{\xi},t)]$$
$$+\left(1+\frac{\lambda}{\delta_t}(e^{-\delta_t/\lambda}-1)\right)$$
$$\times[g(\boldsymbol{x}+\boldsymbol{\xi}\delta_t,\boldsymbol{\xi},t+\delta_t)-g(\boldsymbol{x},\boldsymbol{\xi},t)].$$

If we expand $e^{-\delta_t/\lambda}$ in its Taylor expansion and, further, neglect the terms of order $O(\delta_t^2)$ or smaller on the right-hand side of Eq. (10), then Eq. (10) becomes

$$f(\boldsymbol{x}+\boldsymbol{\xi}\delta_t,\boldsymbol{\xi},t+\delta_t)-f(\boldsymbol{x},\boldsymbol{\xi},t)=-\frac{1}{\tau}[f(\boldsymbol{x},\boldsymbol{\xi},t)-g(\boldsymbol{x},\boldsymbol{\xi},t)],$$

Discretized LB Equation

$$f\left(x+\xi\delta_t,\xi,t+\delta_t\right)-f\left(x,\xi,t\right)=-\frac{1}{\tau}\left[f\left(x,\xi,t\right)-g\left(x,\xi,t\right)\right]$$

$$\rho=\int f\,d\xi=\int g\,d\xi$$

$$\rho u=\int \xi f\,d\xi=\int \xi g\,d\xi$$

$$\rho\varepsilon=\frac{1}{2}\int(\xi-u)^2 f\,d\xi=\frac{1}{2}\int(\xi-u)^2 g\,d\xi$$

Above is the evo [illegible] th discrete time

In which $\frac{\tau}{\delta_t}$ is the dimensionless relaxation time

Recall: we can calculate macroscopic quantities by integrating in momentum space.

The integration can be approximated by quadrature up to a certain degree of accuracy.

Discretized LB Equation

The approximating quadrature takes the form:

$$\int \psi(\xi) g(x,\xi,t)\, d\xi = \sum_{\alpha} W_\alpha \psi(\xi_\alpha) g(x,\xi_\alpha,t)$$

Where Ψ(ξ) is a polynomial of ξ, Wα is the weight coefficient of the quadrature, and ξα is the discrete velocity set. Accordingly, the hydrodynamic moments can be computed by:

$$\rho = \sum_{\alpha} f_\alpha = \sum_{\alpha} g_\alpha$$

$$\rho u = \sum_{\alpha} \xi_\alpha f_\alpha = \sum_{\alpha} \xi_\alpha g_\alpha$$

$$\rho \varepsilon = \frac{1}{2} \sum_{\alpha} (\xi_\alpha - u)^2 f_\alpha = \frac{1}{2} \sum_{\alpha} (\xi_\alpha - u)^2 g_\alpha$$

Where:

$$f_\alpha \equiv f_\alpha(x,t) \equiv W_\alpha f(x,\xi_\alpha,t)$$

$$g_\alpha \equiv g_\alpha(x,t) \equiv W_\alpha g(x,\xi_\alpha,t)$$

Question becomes finding:

1. **A approximation of distribution function f**
2. **Weight coefficients**

Approximation of Distribution Function

Recall Boltzmann-Maxwellian distribution function: $f \equiv \frac{\rho}{(2\pi RT)^{D/2}} \exp\left(-\frac{(\xi - u)^2}{2RT}\right)$

Assume D=2, which means a 2-D case

$$f = \frac{\rho}{(2\pi RT)^{D/2}} \exp\left(-\frac{(\xi - u)^2}{2RT}\right)$$
$$= \frac{\rho}{(2\pi RT)} \exp\left(-\frac{\xi \cdot \xi}{2RT}\right) \exp\left(\frac{2\xi \cdot u - u \cdot u}{2RT}\right)$$
$$\approx \frac{\rho}{(2\pi RT)} \exp\left(-\frac{\xi \cdot \xi}{2RT}\right) \left[1 + \frac{\xi \cdot u}{RT} - \frac{u \cdot u}{2RT} + \frac{(\xi \cdot u)^2}{2(RT)^2}\right]$$
$$= \rho w_\alpha \left[1 + \frac{\xi \cdot u}{RT} - \frac{u \cdot u}{2RT} + \frac{(\xi \cdot u)^2}{2(RT)^2}\right]$$

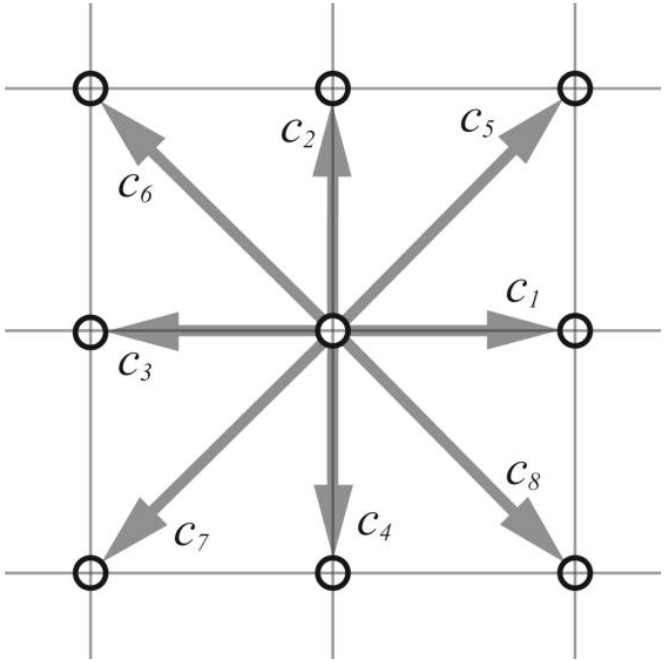
$$f_\alpha = \rho w_\alpha \left[1 + \frac{\xi \cdot u}{RT} - \frac{u \cdot u}{2RT} + \frac{(\xi \cdot u)^2}{2(RT)^2}\right]$$
$$f_\alpha^{EQ} = \rho w_\alpha \left[1 + \frac{\xi \cdot u'}{RT} - \frac{u' \cdot u'}{2RT} + \frac{(\xi \cdot u')^2}{2(RT)^2}\right]$$

$$RT = c_s^2 = c^2/3$$

Cs is the sound speed of the system

Weighting Coefficients

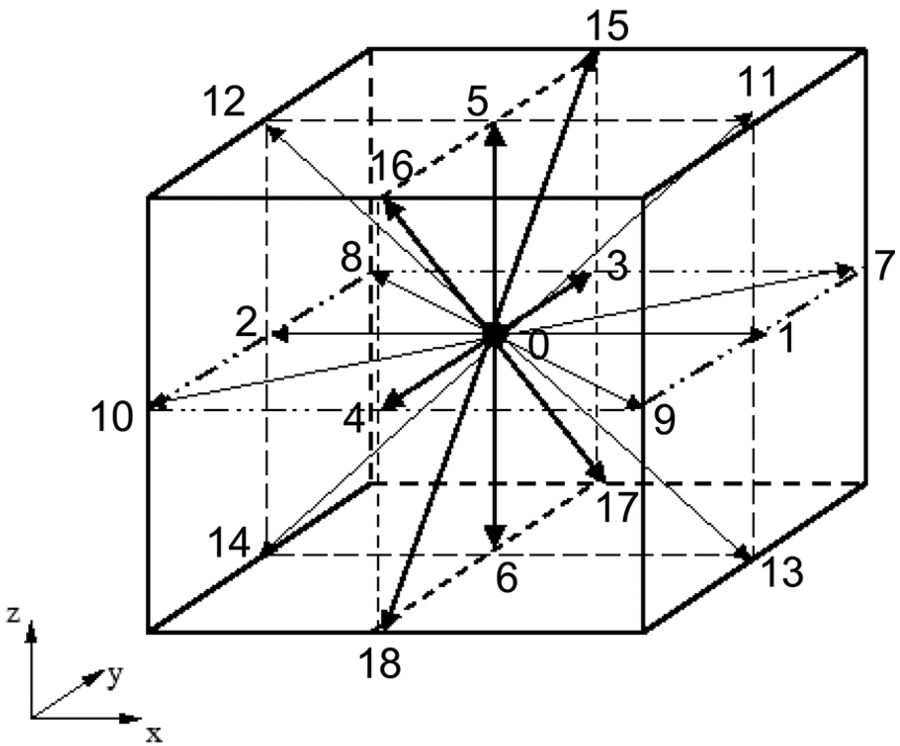
Weight Wα depends on Lattice arrangements



D2Q9 model
Image from nus.edu

$$w_\alpha = \begin{cases} \frac{4}{9}, & \alpha = 9 \\ \frac{1}{9}, & \alpha = 1,2,3,4 \\ \frac{1}{36}, & \alpha = 5,6,7,8 \end{cases}$$

Lattice Boltzmann Method



D3Q19 Lattice model
Image from ASME Digital Collection

$$w_\alpha = \begin{cases} \frac{2}{36} & \alpha = 1 \cdots 6 \\ \frac{1}{36} & \alpha = 7 \cdots 18 \\ \frac{12}{36} & \alpha = 19 \end{cases}$$

Summary

$$f = \rho w_\alpha \left[1 + \frac{\xi \cdot u}{RT} - \frac{u \cdot u}{2RT} + \frac{(\xi \cdot u)^2}{2(RT)^2} \right]$$

$$f^{EQ} = \rho w_\alpha \left[1 + \frac{\xi \cdot u}{RT} - \frac{u \cdot u}{2RT} + \frac{(\xi \cdot u)^2}{2(RT)^2} \right]$$

$$w_\alpha = \begin{cases} \frac{4}{9}, & \alpha = 0 \\ \frac{1}{9}, & \alpha = 1,2,3,4 \\ \frac{1}{36}, & \alpha = 5,6,7,8 \end{cases}$$

$$\rho = \sum_\alpha f_\alpha$$

$$u = \frac{1}{\rho} \sum_\alpha \xi_\alpha f_\alpha$$

$$f(x + \xi\delta_t, \xi, t + \delta_t) - f(x, \xi, t) = -\frac{1}{\tau}\left[f(x, \xi, t) - f^{EQ}(x, \xi, t) \right]$$

PENNSTATE
1855

Algorithm of LBM

Initialize ρ,u f_i and f_i^{eq}

Streaming step & Boundary: move f_i to f_i^* in the direction of ξ_i

Compute macroscopic ρ and u

Compute f_i^{eq}

Collision step: calculate updated distribution function f_i using evolution equation

$$\rho = \sum_\alpha f_\alpha$$

$$u = \frac{1}{\rho}\sum_\alpha \xi_\alpha f_\alpha$$

$$f\left(x+\xi\delta_t,\xi,t+\delta_t\right) = f\left(x,\xi,t\right)$$

$$\rho = \sum_\alpha f_\alpha$$

$$u = \frac{1}{\rho}\sum_\alpha \xi_\alpha f_\alpha$$

$$f^{EQ} = \rho w_\alpha\left[1+\frac{\xi\cdot u}{RT}-\frac{u\cdot u}{2RT}+\frac{\left(\xi\cdot u\right)^2}{2\left(RT\right)^2}\right]$$

$$f\left(x+\xi\delta_t,\xi,t+\delta_t\right)-f\left(x,\xi,t\right) = -\frac{1}{\tau}\left[f\left(x,\xi,t\right)-f^{EQ}\left(x,\xi,t\right)\right]$$

Calculation Example

Steady Channel Flow

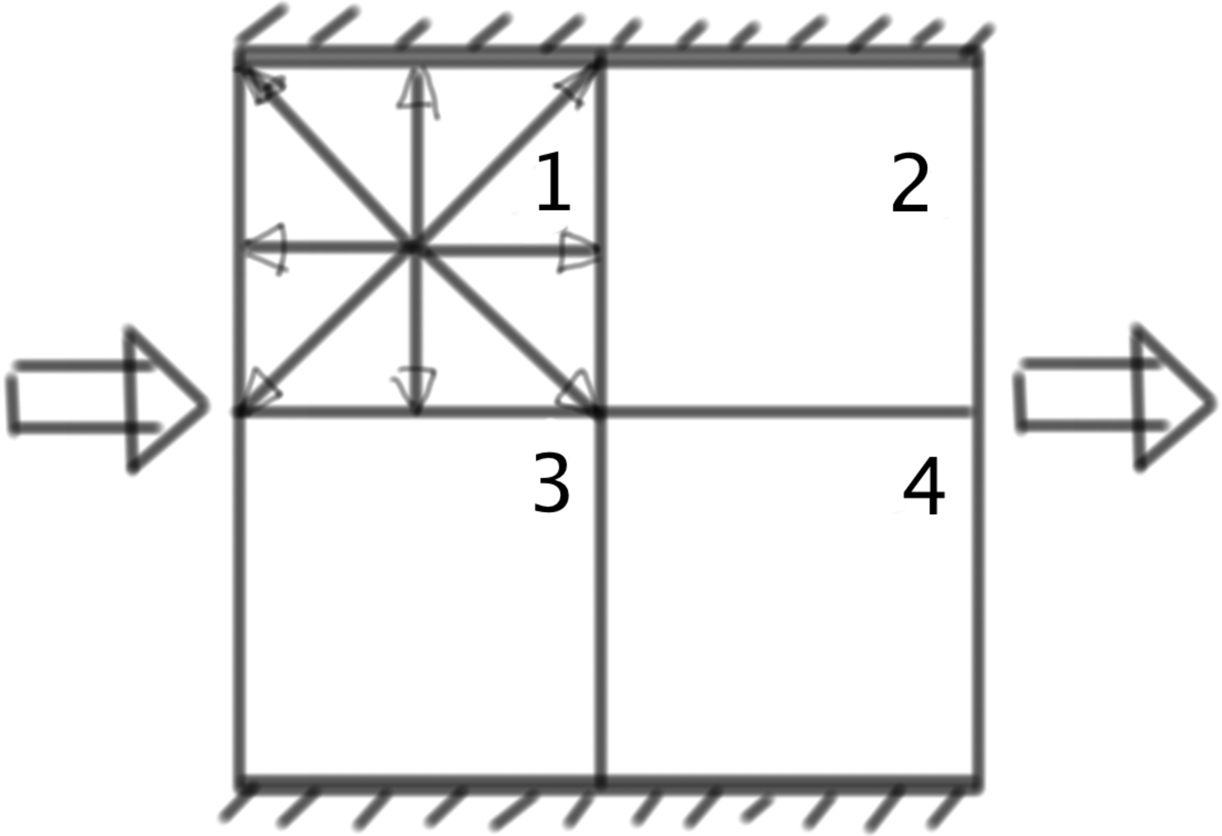
Problem Description

- D2Q9 model
- 2 by 2 system, 4 lattices
- Channel flow from left to right
- Boundary condition--bounce back
- Initial parameter

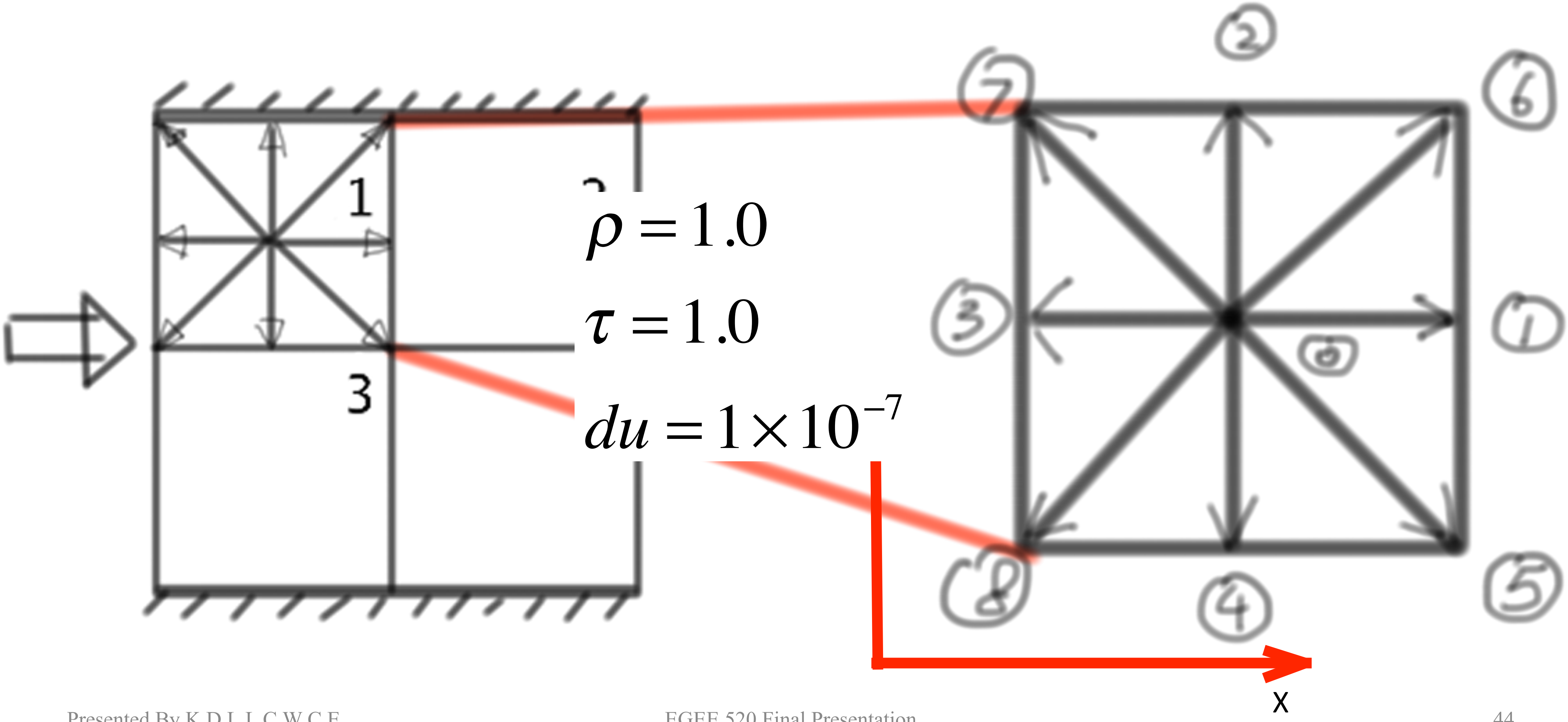
$$\rho = 1.0$$

$$\tau = 1.0$$

$$du = 1 \times 10^{-7}$$



Hand Calculation Example

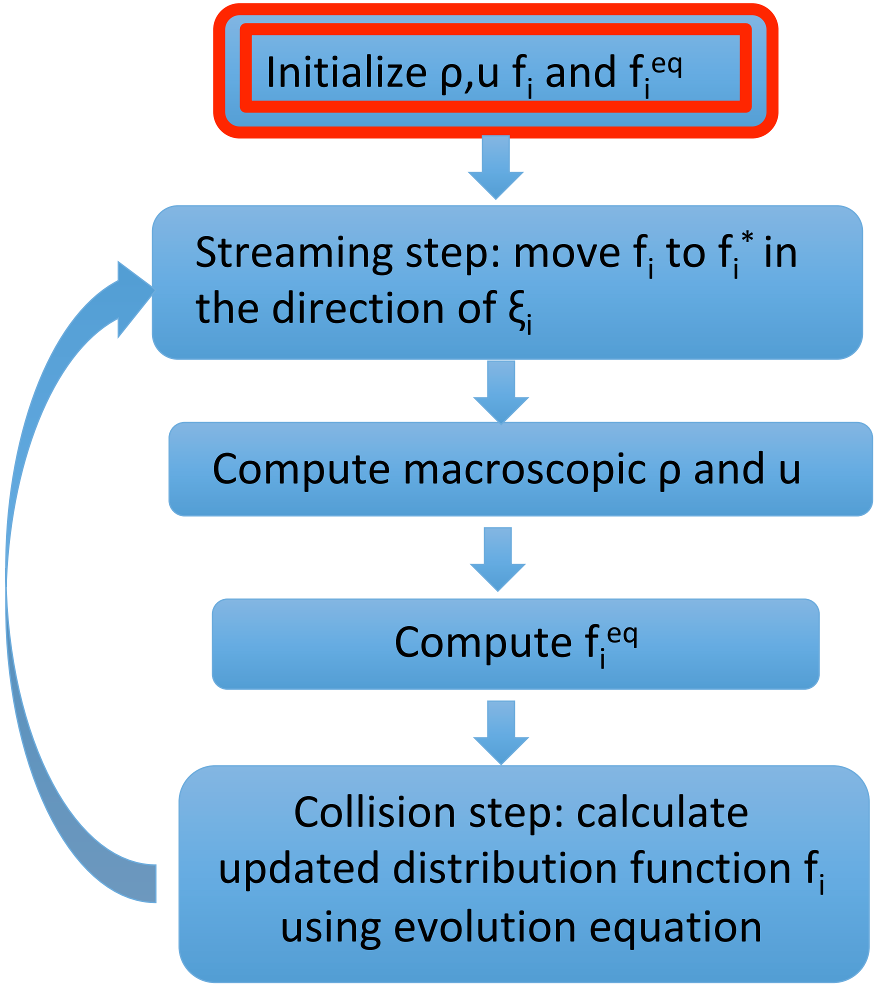


$\rho = 1.0$

$\tau = 1.0$

$du = 1 \times 10^{-7}$

Algorithm of LBM



$$\rho = \sum_\alpha f_\alpha$$
$$u = \frac{1}{\rho}\sum_\alpha \xi_\alpha f_\alpha$$

$$f\left(x+\xi\delta_t,\xi,t+\delta_t\right) = f\left(x,\xi,t\right)$$

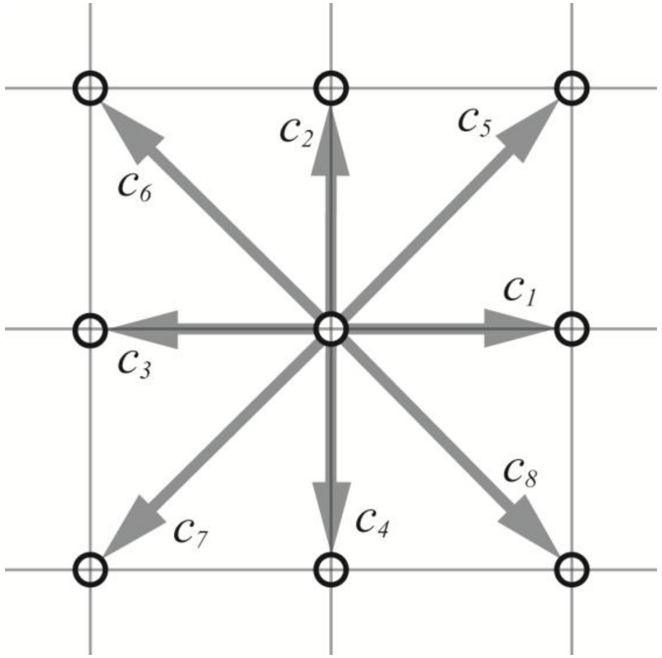
$$\rho = \sum_\alpha f_\alpha$$
$$u = \frac{1}{\rho}\sum_\alpha \xi_\alpha f_\alpha$$

$$f^{EQ} = \rho w_\alpha\left[1+\frac{\xi\cdot u}{RT}-\frac{u\cdot u}{2RT}+\frac{\left(\xi\cdot u\right)^2}{2\left(RT\right)^2}\right]$$

$$f\left(x+\xi\delta_t,\xi,t+\delta_t\right)-f\left(x,\xi,t\right) = -\frac{1}{\tau}\left[f\left(x,\xi,t\right)-f^{EQ}\left(x,\xi,t\right)\right]$$

Hand Calculation Example

For each lattice in D2Q9 model, we present velocity by combination of 9 matrices, each matrix contains distribution function fα, α=1,2,…,9

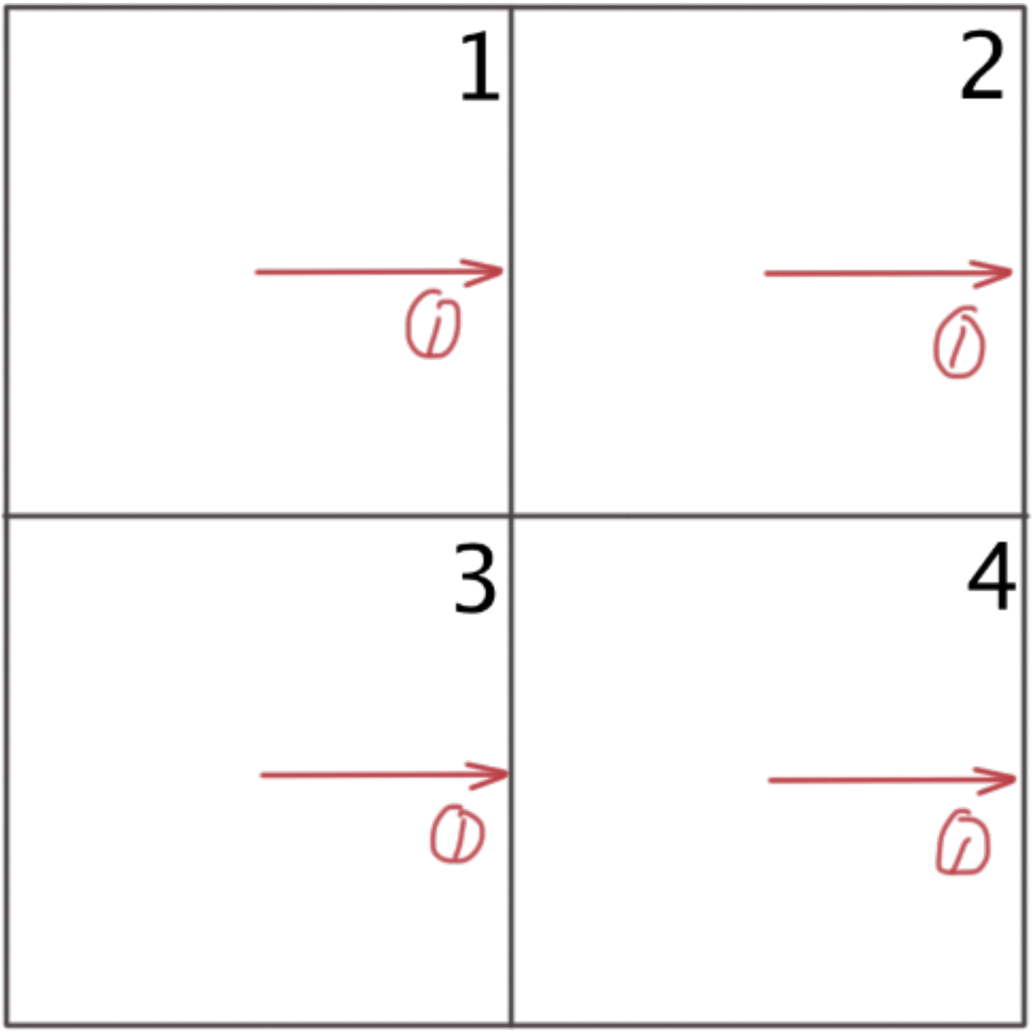


Assume a initial state:

$$f1 = f2 = \ldots = f9$$

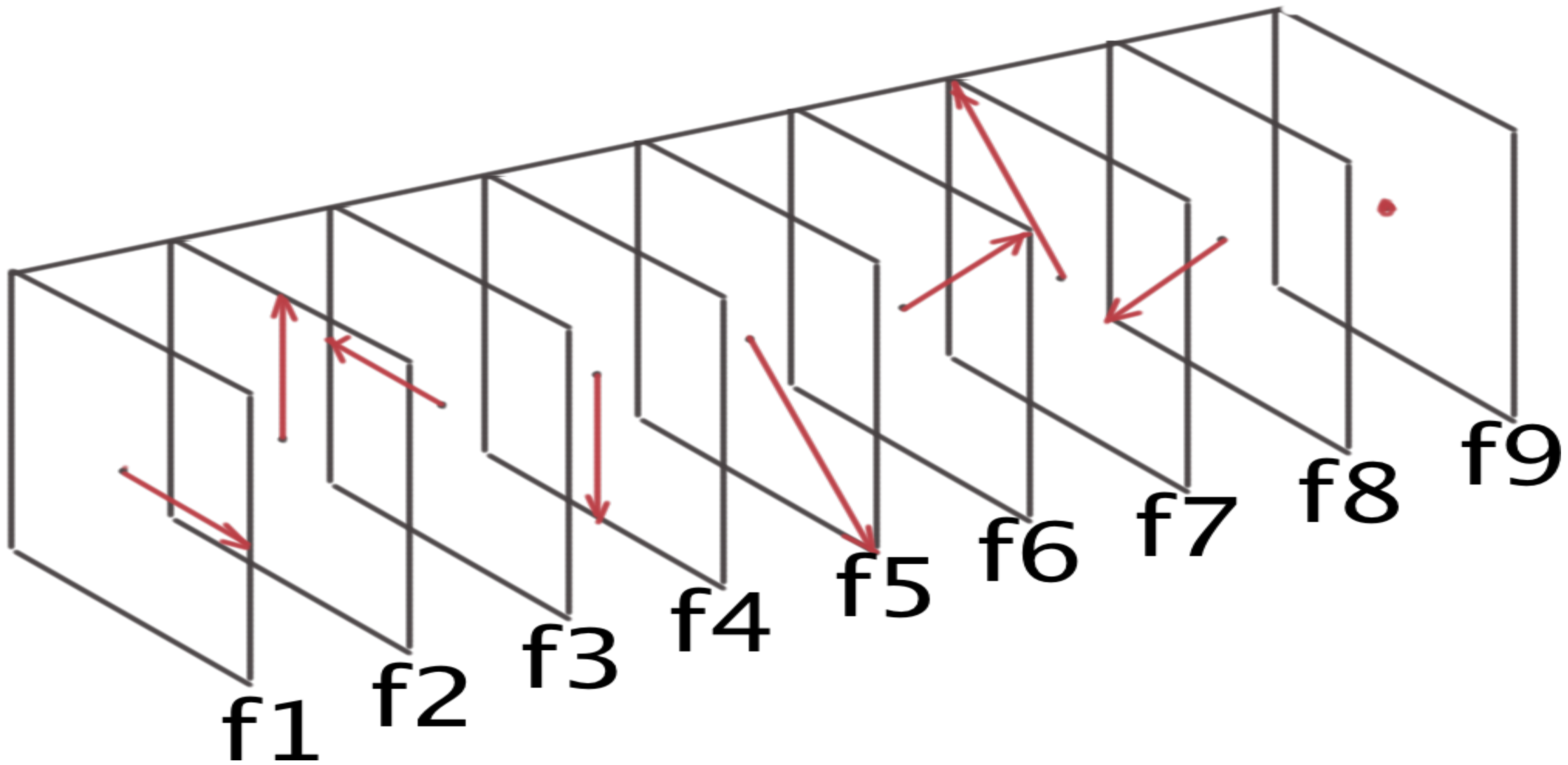
$$f_\alpha = \rho / 9, \alpha = 1,2,3...,9$$

Hand Calculation Example



$$\Rightarrow f_1 = \begin{pmatrix} \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} \end{pmatrix}$$

Hand Calculation Example



Hand Calculation Example

Distribution Function

$$(f1)=\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}(f2)=\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}(f3)=\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}$$

$$(f4)=\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}(f5)=\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}(f6)=\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}$$

$$(f7)=\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}(f8)=\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}(f9)=\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}$$

Algorithm of LBM

Initialize ρ,u f_i and f_i^{eq}

$$\rho = \sum_\alpha f_\alpha \qquad u = \frac{1}{\rho}\sum_\alpha \xi_\alpha f_\alpha$$

Streaming step: move f_i to f_i^* in the direction of ξ_i

$$f\left(x+\xi\delta_t,\xi,t+\delta_t\right) = f\left(x,\xi,t\right)$$

Compute macroscopic ρ and u

$$\rho = \sum_\alpha f_\alpha \qquad u = \frac{1}{\rho}\sum_\alpha \xi_\alpha f_\alpha$$

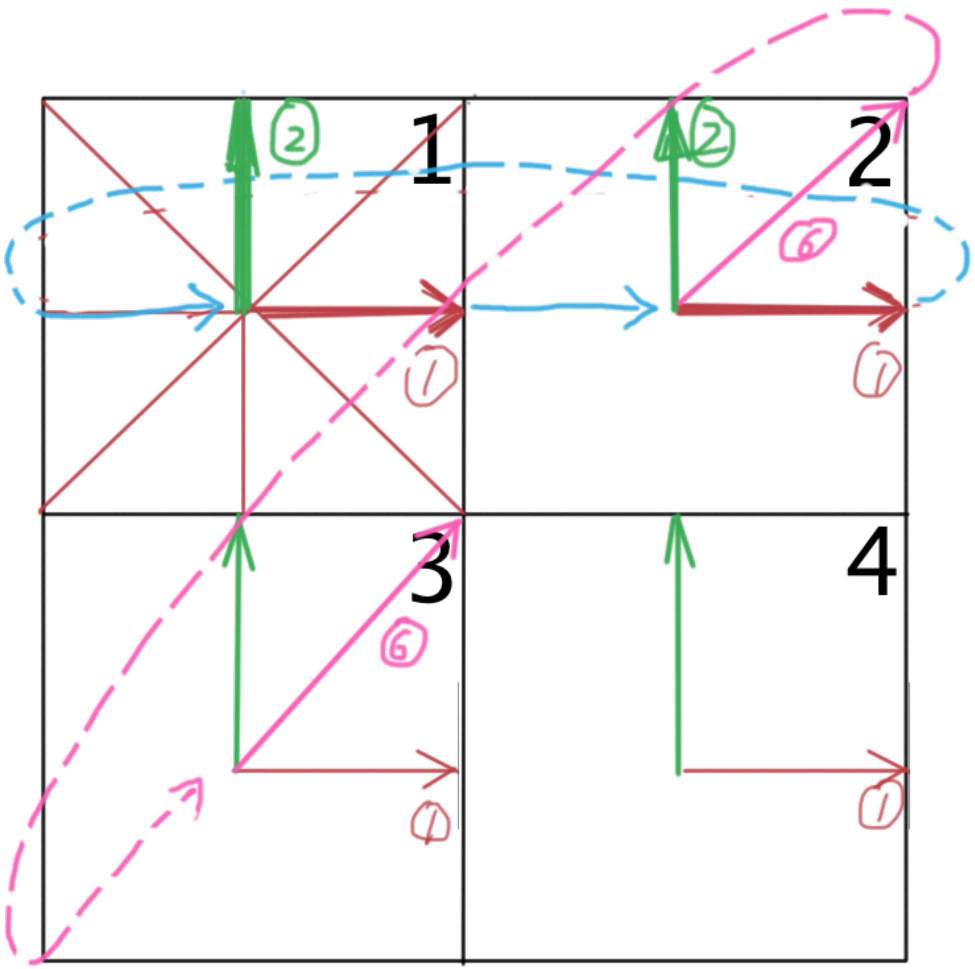
Compute f_i^{eq}

$$f^{EQ} = \rho w_\alpha \left[1 + \frac{\xi\cdot u}{RT} - \frac{u\cdot u}{2RT} + \frac{(\xi\cdot u)^2}{2(RT)^2}\right]$$

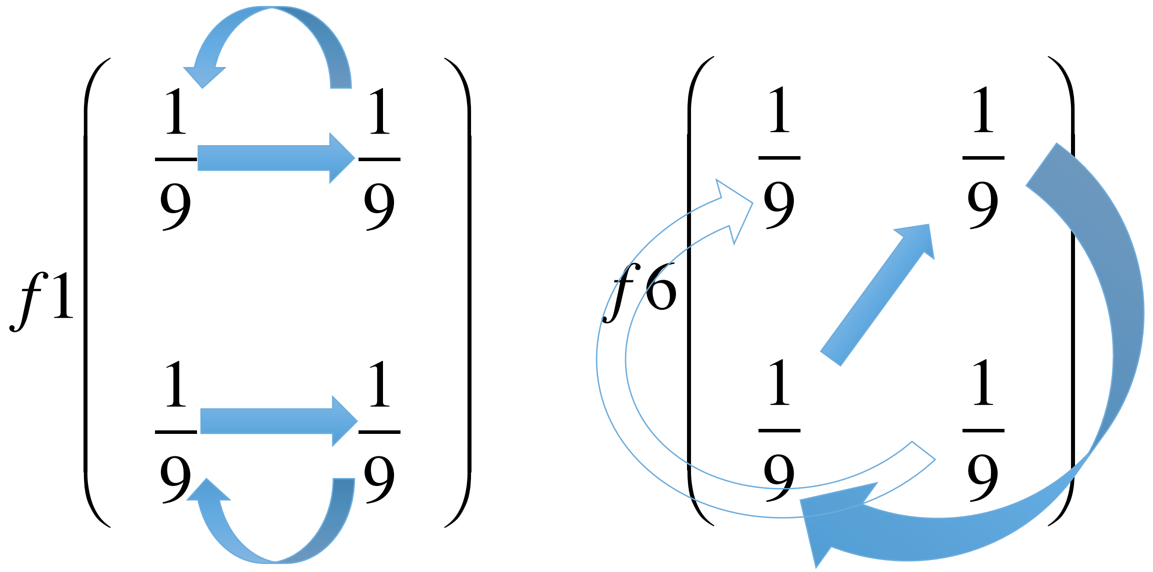
Collision step: calculate updated distribution function f_i using evolution equation

$$f\left(x+\xi\delta_t,\xi,t+\delta_t\right) - f\left(x,\xi,t\right) = -\frac{1}{\tau}\left[f\left(x,\xi,t\right) - f^{EQ}\left(x,\xi,t\right)\right]$$

Hand Calculation Example



Streaming



Hand Calculation Example

$$(f1 \rightarrow)\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix} \Rightarrow \begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix} (f2 \uparrow)\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix} \Rightarrow \begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix} (f3 \leftarrow)\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix} \Rightarrow \begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}$$

$$(f4 \downarrow)\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix} \Rightarrow \begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix} (f5 \searrow)\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix} \Rightarrow \begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix} (f6 \nearrow)\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix} \Rightarrow \begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}$$

$$(f7 \nwarrow)\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix} \Rightarrow \begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix} (f8 \searrow)\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix} \Rightarrow \begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix} (f9 \circ)\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix} \Rightarrow \begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}$$

Algorithm of LBM

Initialize ρ,u f_i and f_i^{eq}

Streaming step: move f_i to f_i^* in the direction of ξ_i

Compute macroscopic ρ and u

Compute f_i^{eq}

Collision step: calculate updated distribution function f_i using evolution equation

$$\rho = \sum_\alpha f_\alpha$$
$$u = \frac{1}{\rho}\sum_\alpha \xi_\alpha f_\alpha$$

$$f(x+\xi\delta_t,\xi,t+\delta_t) = f(x,\xi,t)$$

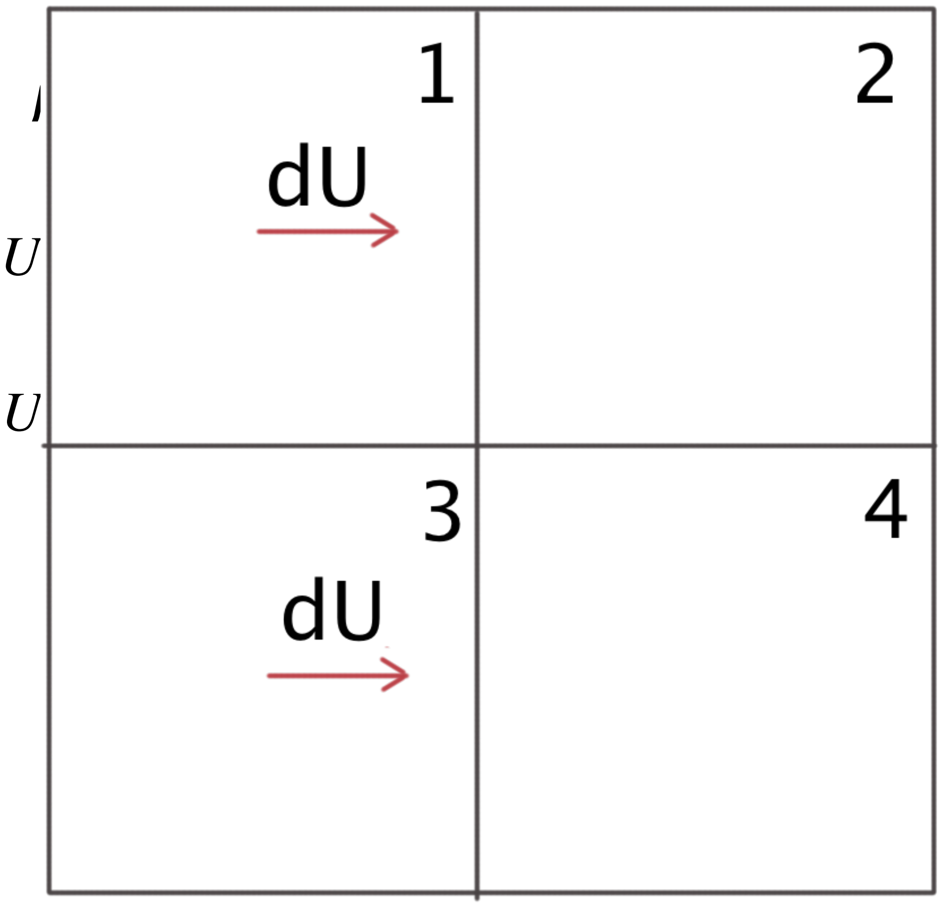
$$\rho = \sum_\alpha f_\alpha$$
$$u = \frac{1}{\rho}\sum_\alpha \xi_\alpha f_\alpha$$

$$f^{EQ} = \rho w_\alpha\left[1+\frac{\xi\cdot u}{RT}-\frac{u\cdot u}{2RT}+\frac{(\xi\cdot u)^2}{2(RT)^2}\right]$$

$$f(x+\xi\delta_t,\xi,t+\delta_t) - f(x,\xi,t) = -\frac{1}{\tau}\left[f(x,\xi,t) - f^{EQ}(x,\xi,t)\right]$$

Hand Calculation Example

Calculate Macroscopic Quantities



$$\rho = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

$$U_x(1) + du$$
$$U_x(3) + du$$

$$U_x = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \qquad U_y = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

$$U \nearrow = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \qquad U \swarrow = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

$$U \nwarrow = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \qquad U \searrow = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

Hand Calculation Example

Calculate Macroscopic Velocities

$$\rho_\alpha = \sum_\alpha f_\alpha = \frac{1}{9} \times 9 = 1$$

$$U_x = \frac{1}{\rho}\left((f1 + f5 + f6) - (f3 + f7 + f8)\right)$$

$$U_y = \frac{1}{\rho}\left((f6 + f2 + f7) - (f5 + f4 + f8)\right)$$

$$U = U_x^2 + U_y^2$$

$$U \nearrow = U_x + U_y$$

$$U \searrow = U_x - U_y$$

$$U \nwarrow = -U \searrow$$

$$U \swarrow = -U \nearrow$$

$$\rho = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

$$U_x(1) + du$$

$$U_x(3) + du$$

$$U_x = \begin{pmatrix} 1e-7 & 0 \\ 1e-7 & 0 \end{pmatrix} \qquad U_y = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

$$U = \begin{pmatrix} 1e-7 & 0 \\ 1e-7 & 0 \end{pmatrix}$$

$$U \nearrow = \begin{pmatrix} 1e-7 & 0 \\ 1e-7 & 0 \end{pmatrix} \qquad U \swarrow = \begin{pmatrix} -1e-7 & 0 \\ -1e-7 & 0 \end{pmatrix}$$

$$U \nwarrow = \begin{pmatrix} -1e-7 & 0 \\ -1e-7 & 0 \end{pmatrix} \qquad U \searrow = \begin{pmatrix} 1e-7 & 0 \\ 1e-7 & 0 \end{pmatrix}$$

PENNSTATE

1855

Algorithm of LBM

1. Initialize ρ,u f_i and f_i^{eq}
2. Streaming step: move f_i to f_i^* in the direction of ξ_i
3. Compute macroscopic ρ and u
4. **Compute f_i^{eq}**
5. Collision step: calculate updated distribution function f_i using evolution equation

$$\rho = \sum_\alpha f_\alpha$$
$$u = \frac{1}{\rho}\sum_\alpha \xi_\alpha f_\alpha$$

$$f\left(x+\xi\delta_t,\xi,t+\delta_t\right)=f\left(x,\xi,t\right)$$

$$\rho = \sum_\alpha f_\alpha$$
$$u = \frac{1}{\rho}\sum_\alpha \xi_\alpha f_\alpha$$

$$f^{EQ}=\rho w_\alpha\left[1+\frac{\xi\cdot u}{RT}-\frac{u\cdot u}{2RT}+\frac{\left(\xi\cdot u\right)^2}{2\left(RT\right)^2}\right]$$

$$f\left(x+\xi\delta_t,\xi,t+\delta_t\right)-f\left(x,\xi,t\right)=-\frac{1}{\tau}\left[f\left(x,\xi,t\right)-f^{EQ}\left(x,\xi,t\right)\right]$$

Hand Calculation Example

Calculate Equilibrium State Distribution Function

$$f_\alpha^{EQ} = \rho w_\alpha \left[1 + \frac{\xi \cdot u'}{RT} - \frac{u' \cdot u'}{2RT} + \frac{(\xi \cdot u')^2}{2(RT)^2}\right]$$

$$\xi \cdot u' = U_x; U_y; U\nearrow; U\swarrow; U\searrow; U\nwarrow .$$

$$u' \cdot u' = U^2$$

$$RT = c_s^2 = c^2/3$$

$$c = 1.0$$

$$f_1^{EQ} = \rho w_1 \left[1 + \frac{U_x}{RT} - \frac{U^2}{2RT} + \frac{(U_x)^2}{2(RT)^2}\right]$$

$$f_2^{EQ} = \rho w_2 \left[1 + \frac{U_y}{RT} - \frac{U^2}{2RT} + \frac{(U_y)^2}{2(RT)^2}\right]$$

$$f_3^{EQ} = \rho w_3 \left[1 + \frac{-U_x}{RT} - \frac{U^2}{2RT} + \frac{(-U_x)^2}{2(RT)^2}\right]$$

$$f_4^{EQ} = \rho w_4 \left[1 + \frac{-U_y}{RT} - \frac{U^2}{2RT} + \frac{(-U_y)^2}{2(RT)^2}\right]$$

$$f_5^{EQ} = \rho w_5 \left[1 + \frac{U\nearrow}{RT} - \frac{U^2}{2RT} + \frac{(U\nearrow)^2}{2(RT)^2}\right]$$

$$f_6^{EQ} = \rho w_6 \left[1 + \frac{U\nwarrow}{RT} - \frac{U^2}{2RT} + \frac{(U\nwarrow)^2}{2(RT)^2}\right]$$

$$f_7^{EQ} = \rho w_7 \left[1 + \frac{U\swarrow}{RT} - \frac{U^2}{2RT} + \frac{(U\swarrow)^2}{2(RT)^2}\right]$$

$$f_8^{EQ} = \rho w_8 \left[1 + \frac{U\searrow}{RT} - \frac{U^2}{2RT} + \frac{(U\searrow)^2}{2(RT)^2}\right]$$

$$f_9^{EQ} = \rho w_9 \left[1 + \frac{-U\circ}{RT} - \frac{U^2}{2RT} + \frac{(-U\circ)^2}{2(RT)^2}\right]$$

PENNSTATE 1855

Hand Calculation Example

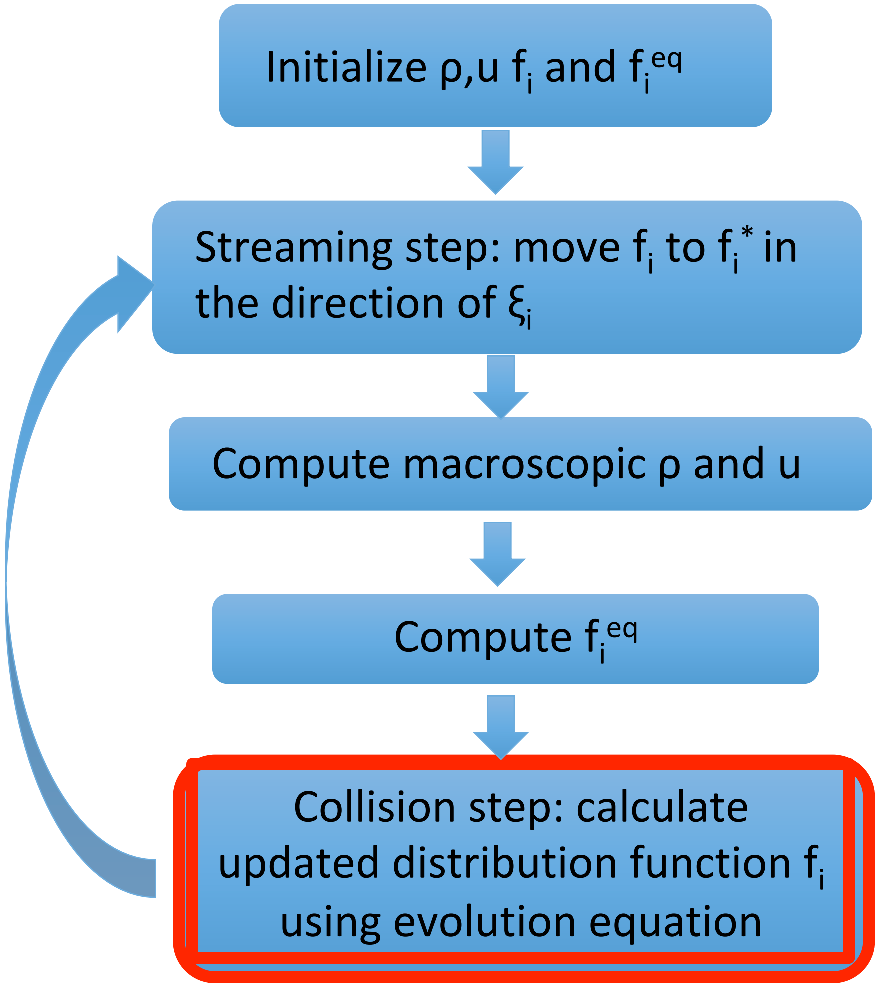
Calculated Equilibrium State Distribution Function

$$\left(f1^{EQ}\rightarrow\right)=\begin{pmatrix}0.01234568 & 0.012345679\\0.01234568 & 0.012345679\end{pmatrix}\left(f2^{EQ}\uparrow\right)=\begin{pmatrix}0.012345679 & 0.012345679\\0.012345679 & 0.012345679\end{pmatrix}\left(f3^{EQ}\leftarrow\right)=\begin{pmatrix}0.01234567 & 0.012345679\\0.01234567 & 0.012345679\end{pmatrix}$$

$$\left(f4^{EQ}\downarrow\right)=\begin{pmatrix}0.012345679 & 0.012345679\\0.012345679 & 0.012345679\end{pmatrix}\left(f5^{EQ}\searrow\right)=\begin{pmatrix}0.003086 & 0.003086\\0.003086 & 0.003086\end{pmatrix}\left(f6^{EQ}\nearrow\right)=\begin{pmatrix}0.003086 & 0.003086\\0.003086 & 0.003086\end{pmatrix}$$

$$\left(f7^{EQ}\nwarrow\right)=\begin{pmatrix}0.003086 & 0.003086\\0.003086 & 0.003086\end{pmatrix}\left(f8^{EQ}\searrow\right)=\begin{pmatrix}0.003086 & 0.003086\\0.003086 & 0.003086\end{pmatrix}\left(f9^{EQ}\circ\right)=\begin{pmatrix}0.0493827 & 0.0493827\\0.0493827 & 0.0493827\end{pmatrix}$$

Algorithm of LBM



$$\rho = \sum_\alpha f_\alpha$$

$$u = \frac{1}{\rho} \sum_\alpha \xi_\alpha f_\alpha$$

$$f\left(x+\xi\delta_t,\xi,t+\delta_t\right) = f\left(x,\xi,t\right)$$

$$\rho = \sum_\alpha f_\alpha$$

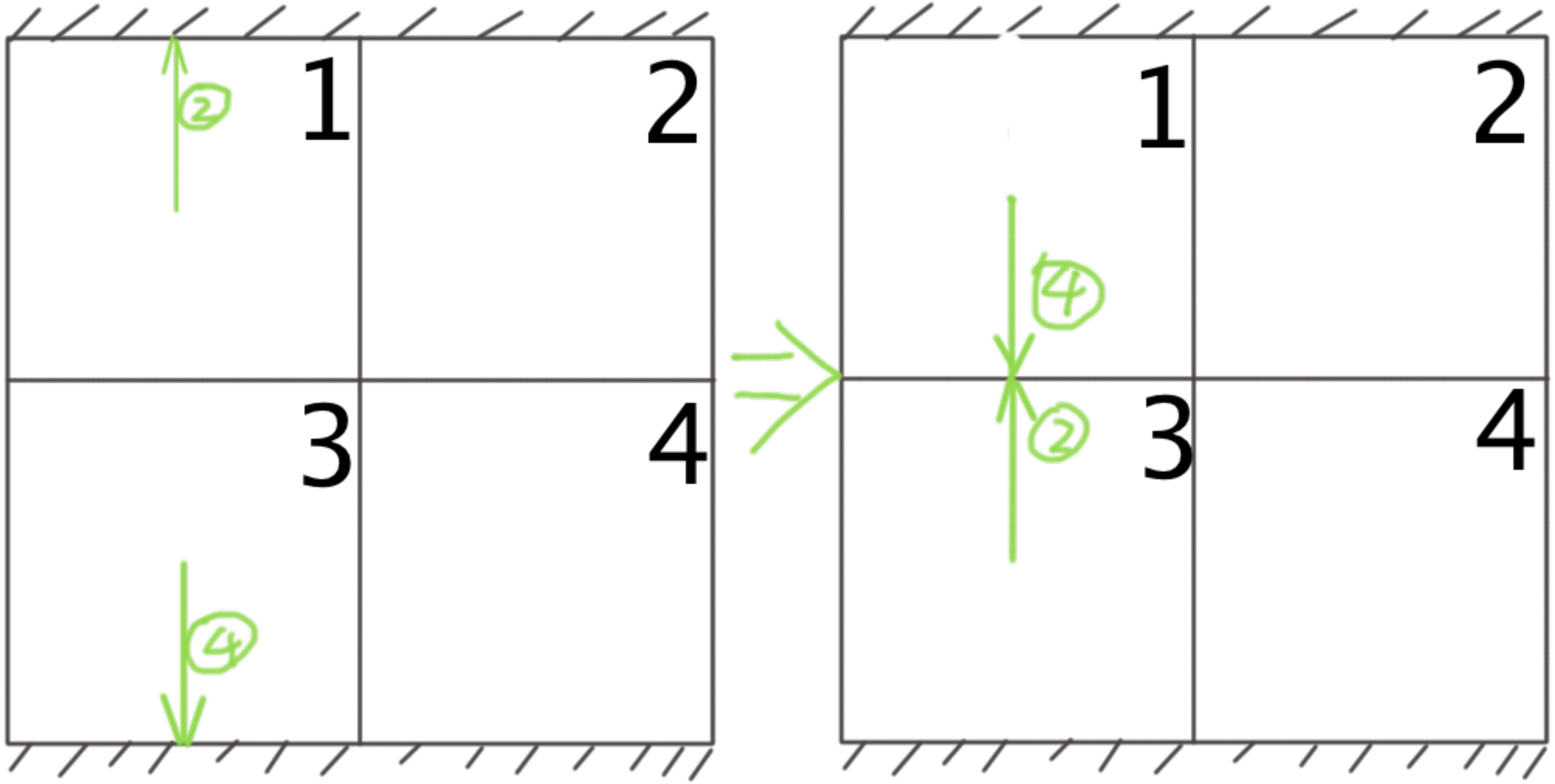
$$u = \frac{1}{\rho} \sum_\alpha \xi_\alpha f_\alpha$$

$$f^{EQ} = \rho w_\alpha \left[1 + \frac{\xi \cdot u}{RT} - \frac{u \cdot u}{2RT} + \frac{(\xi \cdot u)^2}{2(RT)^2}\right]$$

$$f\left(x+\xi\delta_t,\xi,t+\delta_t\right) - f\left(x,\xi,t\right) = -\frac{1}{\tau}\left[f\left(x,\xi,t\right) - f^{EQ}\left(x,\xi,t\right)\right]$$

Hand Calculation Example

Bounce back collision

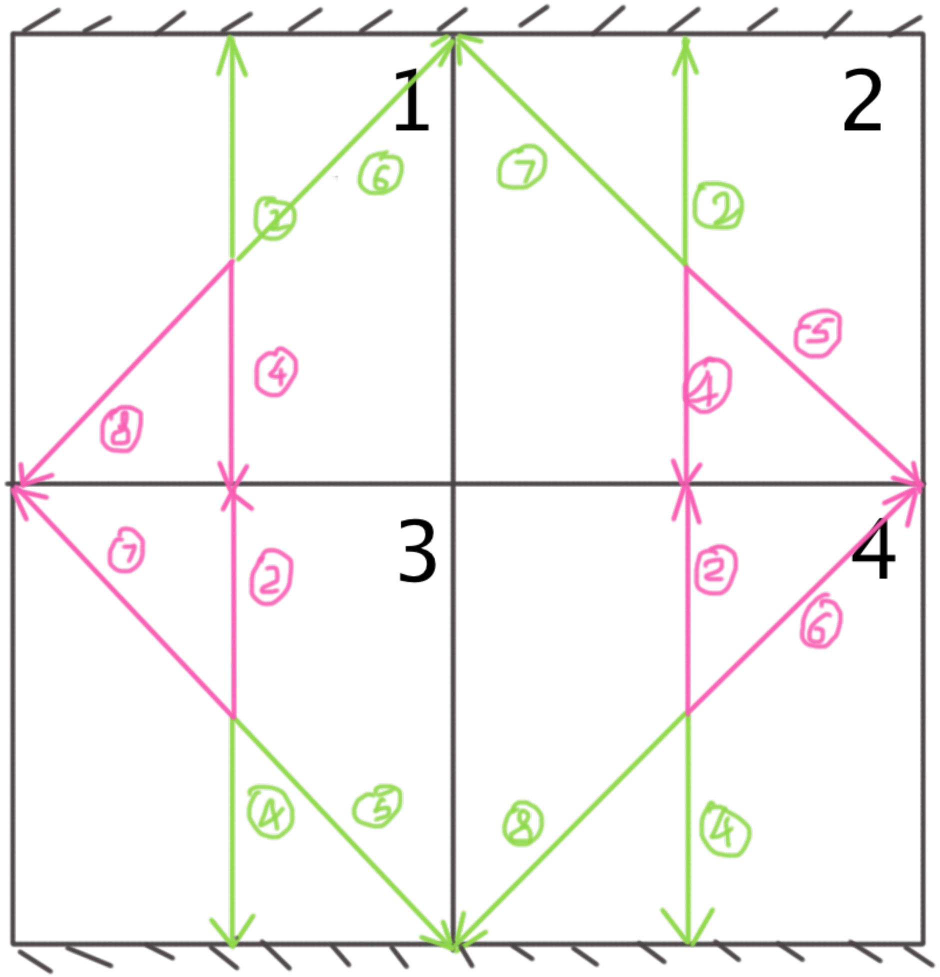


Hand Calculation Example

Bounce back collision

$$f4\begin{pmatrix} \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} \end{pmatrix} \Rightarrow f2\begin{pmatrix} \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} \end{pmatrix} \qquad f2\begin{pmatrix} \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} \end{pmatrix} \Rightarrow f4\begin{pmatrix} \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} \end{pmatrix}$$

Hand Calculation Example



$$\left(f2\uparrow\right)\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}\Rightarrow\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}\left(f4\downarrow\right)\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}\Rightarrow\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}$$

$$\left(f5\searrow\right)\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}\Rightarrow\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}\left(f6\nearrow\right)\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}\Rightarrow\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}$$

$$\left(f7\nwarrow\right)\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}\Rightarrow\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}\left(f8\searrow\right)\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}\Rightarrow\begin{pmatrix}\frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9}\end{pmatrix}$$

Hand Calculation Example

$$f(x+\xi\delta_t,\xi,t+\delta_t)-f(x,\xi,t)=-\frac{1}{\tau}\left[f(x,\xi,t)-f^{EQ}(x,\xi,t)\right]$$

$$\because \tau = 1.0$$

$$f(t+\delta t)-f=-\frac{1}{\tau}\left[f-f^{EQ}\right]$$

$$\Rightarrow f(t+\delta t)=f-\left[f-f^{EQ}\right]$$

$$\Rightarrow f(t+\delta t)=f^{EQ}$$

T=1.0 indicates that distribution function goes to equilibrium within the current time step

Hand Calculation Example

$$\left(f1^{EQ}\rightarrow\right)' = \begin{pmatrix} 0.01234568 & 0.012345679 \\ 0.01234568 & 0.012345679 \end{pmatrix} \left(f2^{EQ}\uparrow\right)' = \begin{pmatrix} 0.012345679 & 0.012345679 \\ 0.012345679 & 0.012345679 \end{pmatrix} \left(f3^{EQ}\leftarrow\right)' = \begin{pmatrix} 0.01234567 & 0.012345679 \\ 0.01234567 & 0.012345679 \end{pmatrix}$$

$$\left(f4^{EQ}\downarrow\right)' = \begin{pmatrix} 0.012345679 & 0.012345679 \\ 0.012345679 & 0.012345679 \end{pmatrix} \left(f5^{EQ}\searrow\right)' = \begin{pmatrix} 0.003086 & 0.003086 \\ 0.003086 & 0.003086 \end{pmatrix} \left(f6^{EQ}\nearrow\right)' = \begin{pmatrix} 0.003086 & 0.003086 \\ 0.003086 & 0.003086 \end{pmatrix}$$

$$\left(f7^{EQ}\nwarrow\right)' = \begin{pmatrix} 0.003086 & 0.003086 \\ 0.003086 & 0.003086 \end{pmatrix} \left(f8^{EQ}\searrow\right)' = \begin{pmatrix} 0.003086 & 0.003086 \\ 0.003086 & 0.003086 \end{pmatrix} \left(f9^{EQ}\circ\right)' = \begin{pmatrix} 0.0493827 & 0.0493827 \\ 0.0493827 & 0.0493827 \end{pmatrix}$$

PENNSTATE

1855

Algorithm of LBM

Initialize ρ,u f_i and f_i^{eq}

Streaming step: move f_i to f_i^* in the direction of ξ_i

Compute macroscopic ρ and u

Compute f_i^{eq}

Collision step: calculate updated distribution function f_i using evolution equation

$$\rho = \sum_\alpha f_\alpha \qquad u = \frac{1}{\rho}\sum_\alpha \xi_\alpha f_\alpha$$

$$f\left(x+\xi\delta_t,\xi,t+\delta_t\right) = f\left(x,\xi,t\right)$$

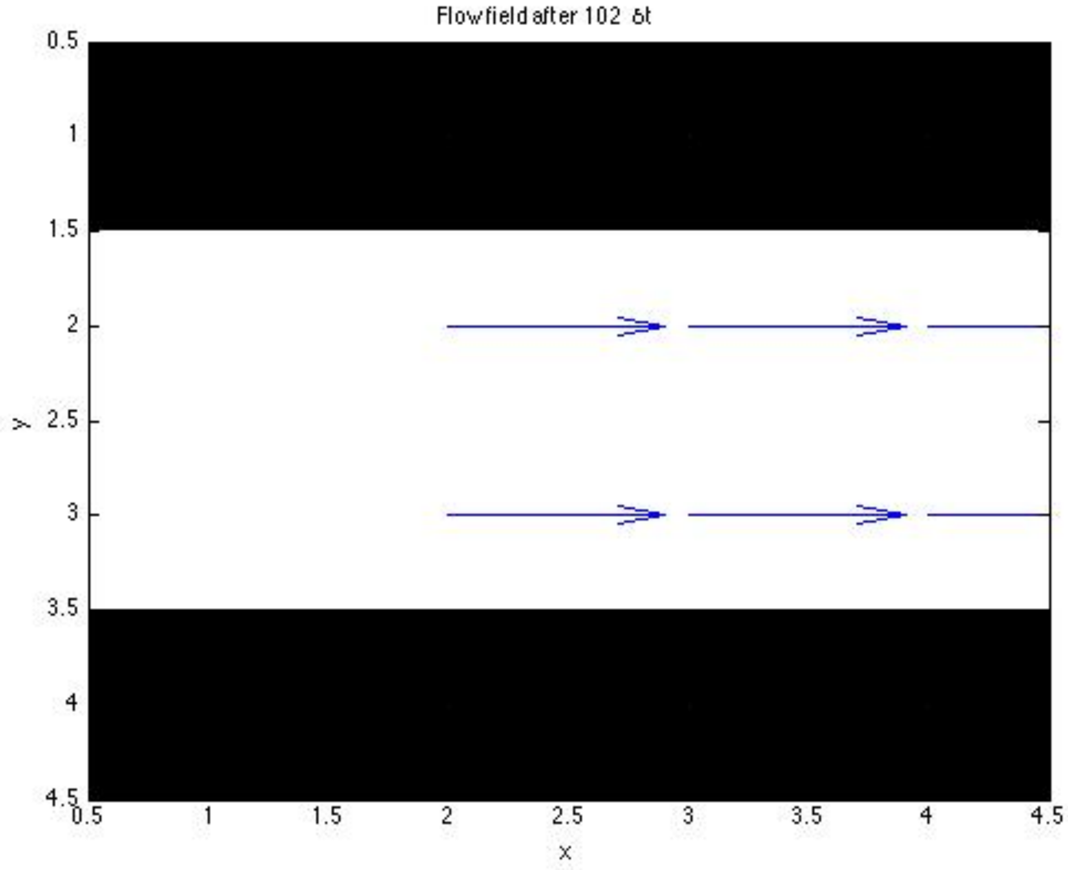
$$\rho = \sum_\alpha f_\alpha \qquad u = \frac{1}{\rho}\sum_\alpha \xi_\alpha f_\alpha$$

$$f^{EQ} = \rho w_\alpha\left[1+\frac{\xi\cdot u}{RT}-\frac{u\cdot u}{2RT}+\frac{\left(\xi\cdot u\right)^2}{2\left(RT\right)^2}\right]$$

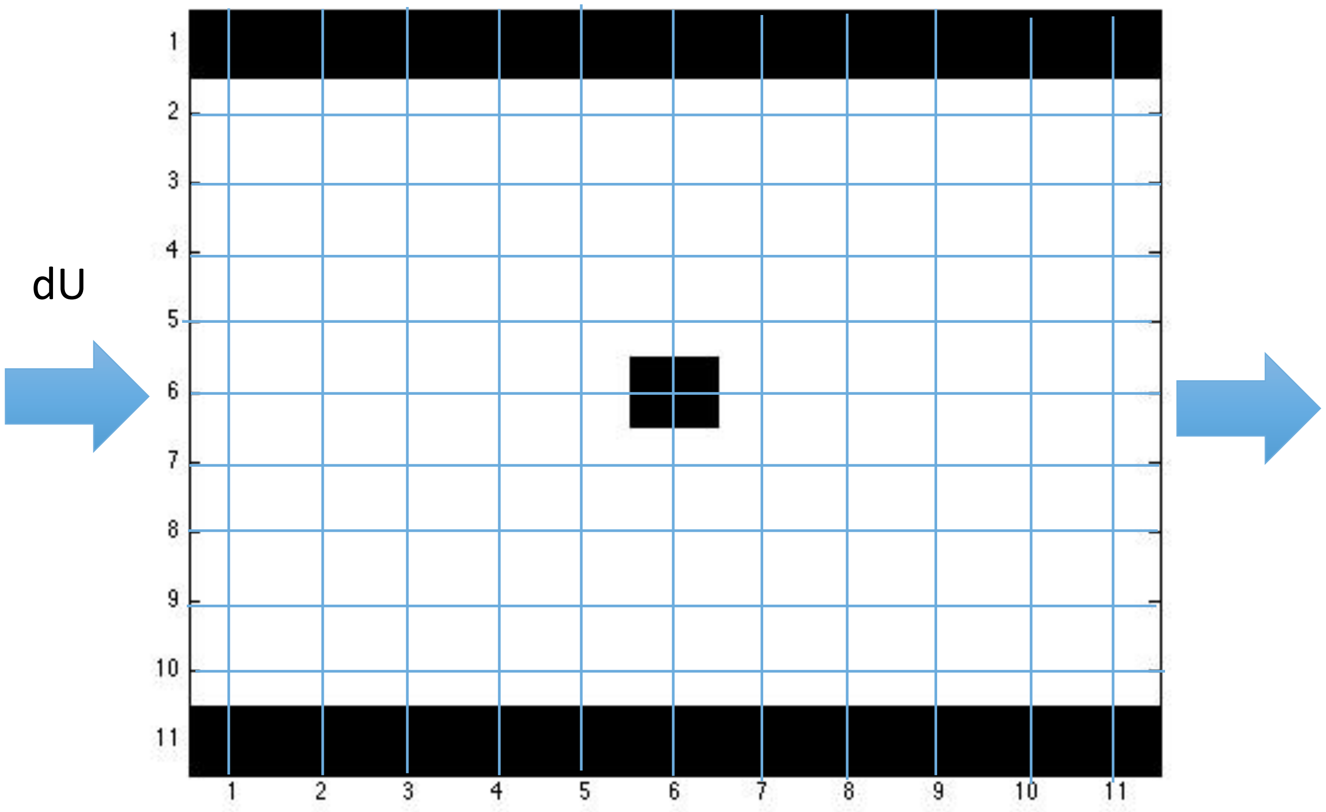
$$f\left(x+\xi\delta_t,\xi,t+\delta_t\right)-f\left(x,\xi,t\right) = -\frac{1}{\tau}\left[f\left(x,\xi,t\right)-f^{EQ}\left(x,\xi,t\right)\right]$$

Hand Calculation Example

Calculation results and visualization



Problem Description



Steady Fluid Flow through a channel with a block in the middle

$$\tau = 1.0$$

$$\rho = 1.0$$

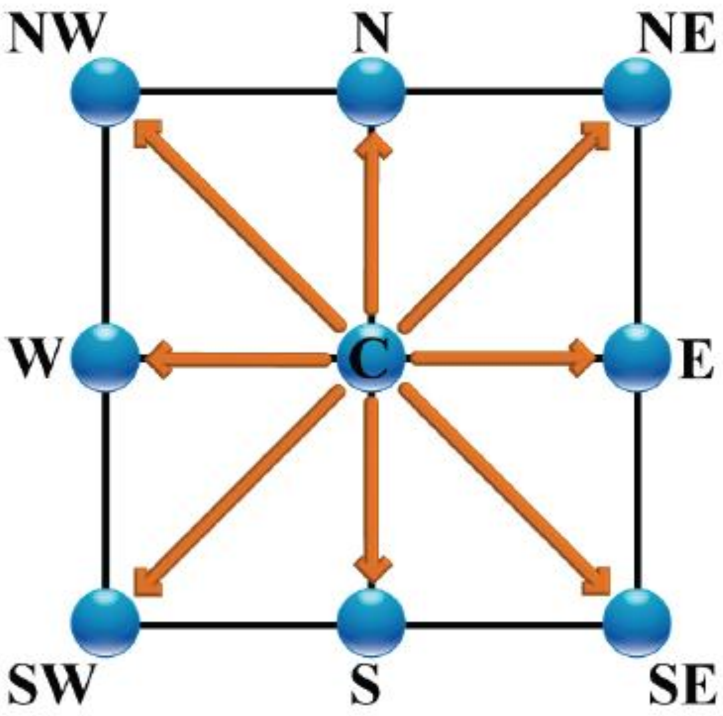
$$RT = 1/3$$

D2Q9 MODEL
11×11 mesh
100 active lattice

MATLAB based calculation

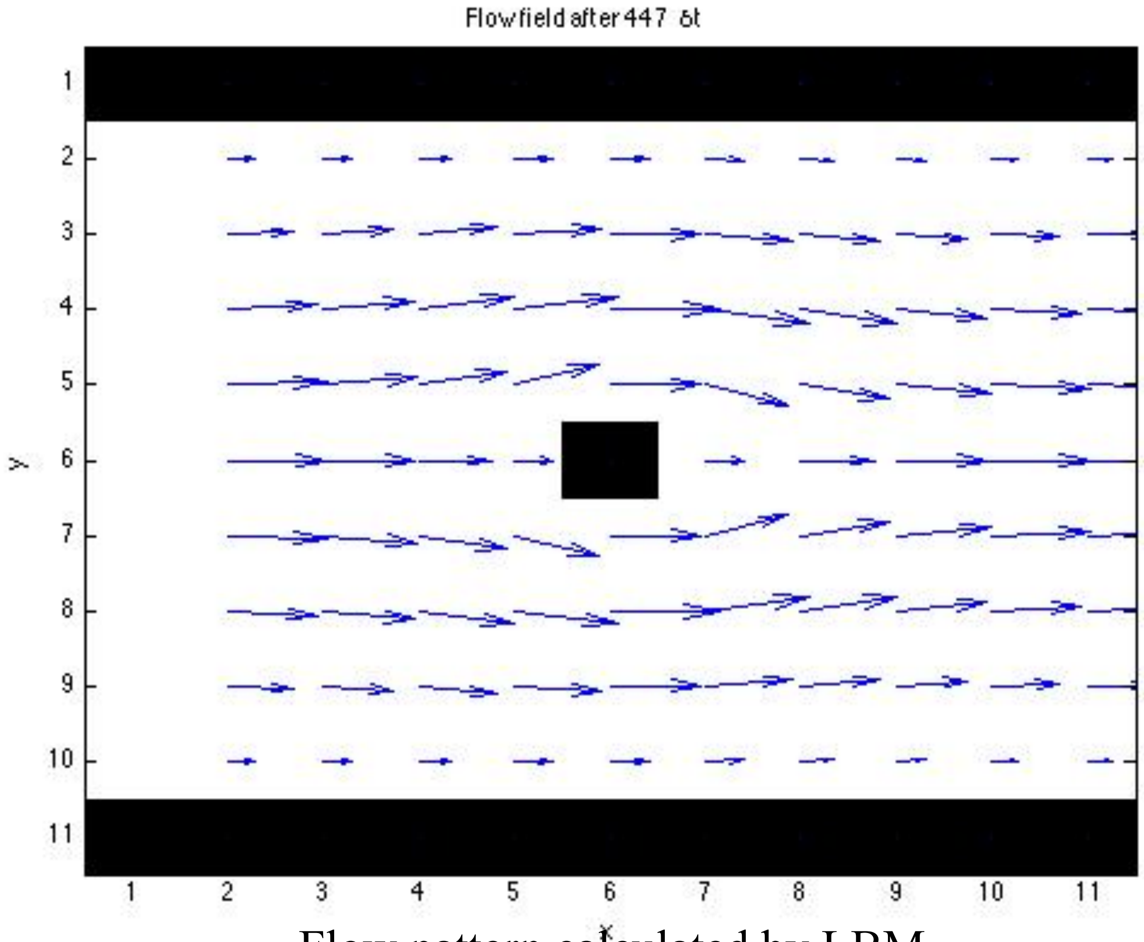
Why?

- Every lattice contains a 9-dimensional matrix
- Lengthy calculation hard to present by hand
- In MATLAB multi-dimensional matrix can be easily presented



D2Q9 model

MATLAB based calculation



Flow pattern calculated by LBM

Numerical Example

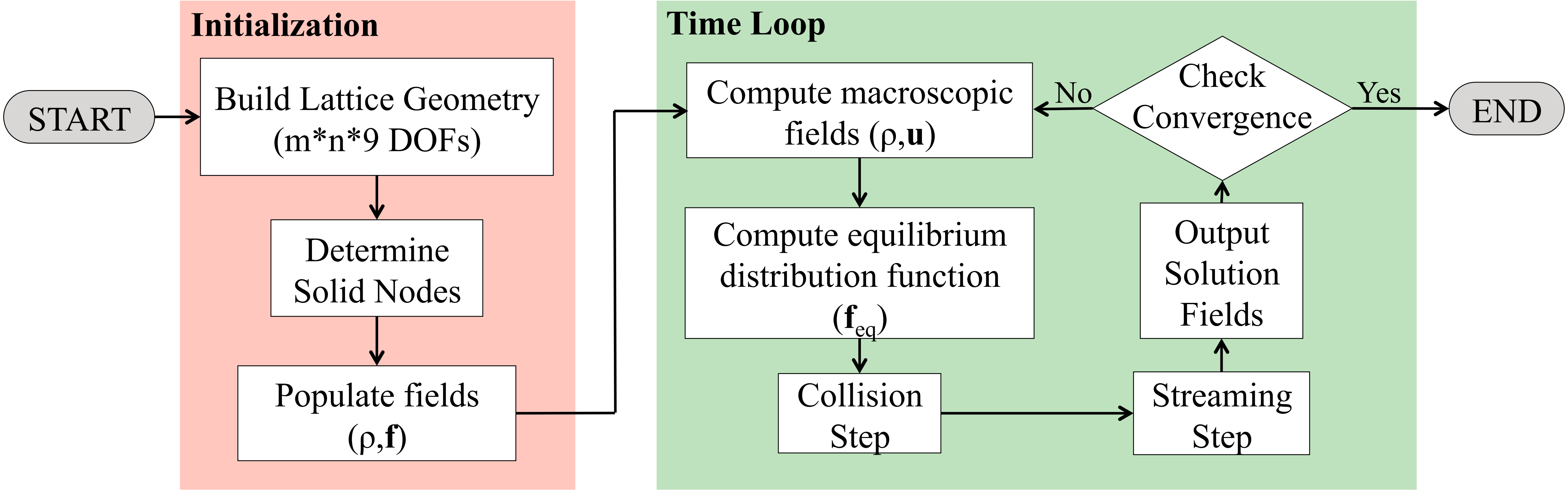
Implementation and Results

Motivation

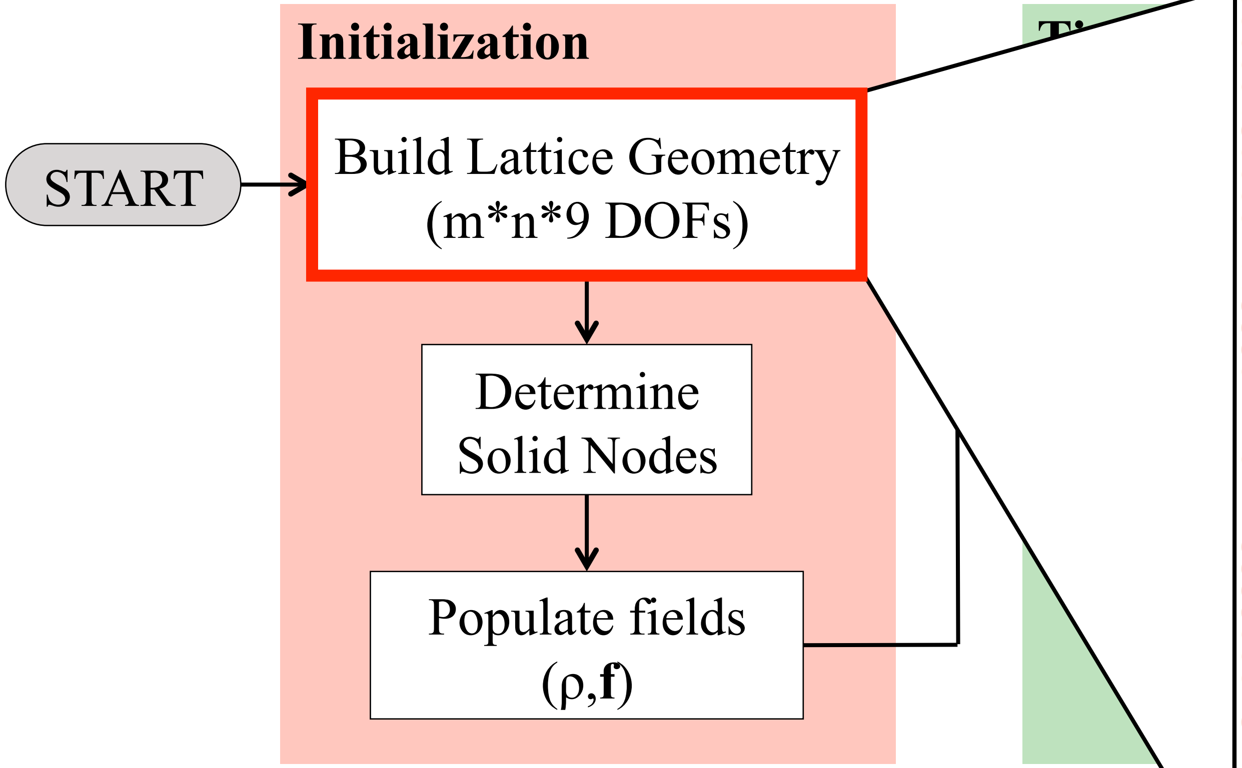
- Porous Media Flow
  - Discrete Simulation (NS) vs. Averaged Flow (Darcy)
  - Couple with transport/heat transfer

- Lattice-Boltzmann Methods
  - Incompressible Navier-Stokes
    - Water/Oil
  - Complex/Stochastic Geometries
    - Simple Meshing
  - Scalable
    - Large simulations

Implementation



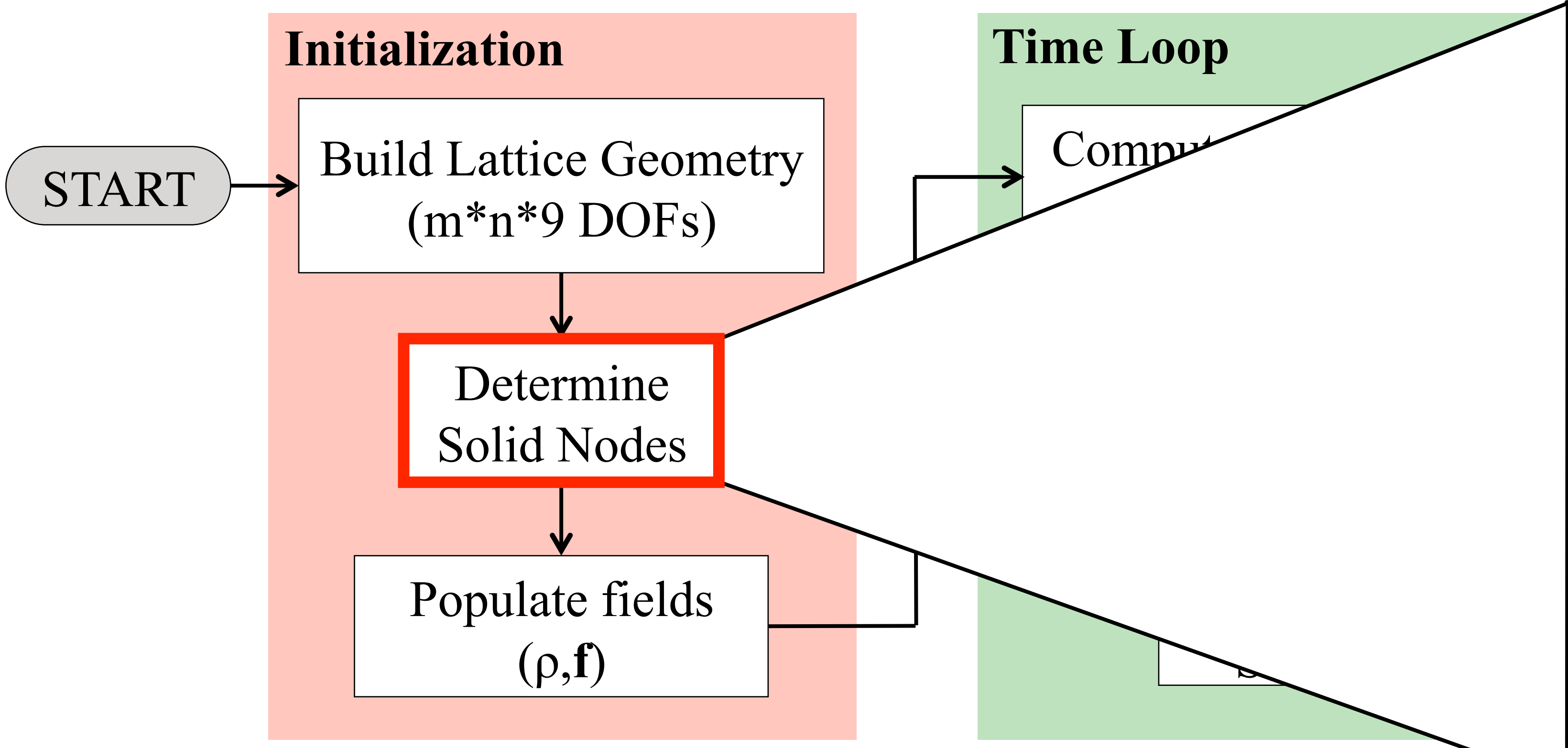
Implementation




```
const int m = 100;
const int n = 400; //m,n: dimensions of triangulation
const double tau = 1.0; //tau: relaxation parameter/kinematic viscosity (nu = (2*tau - 1)/6)
const double g = 0.001; //g: gravity
const double dist = 0.05; //dist: solid percentage
int rad = 6; //rad: radius of solid particle
double distRad = 0.003125; // = dist/pow(2*rad,2); //distRad: solid distribution due to size of particles
const double rhoS = 1.0; //rhoS: fluid density
const int dur = 500; //dur: simulation time
const double ep = 1e-6; //ep: error tolerance
const int minIt = 10; //minIt: minimum number of time steps

int isSolid [m][n]; //isSolid: array of solid node locations
int i_p, j_p; //i_p,j_p: periodic indices
double u [m][n][2]; //u: velocity
double rho [m][n]; //rho: density
double f[m][n][9]; //f: distribution function
const double t [9] = {4./9., 1./9., 1./36.,
                      1./9., 1./36.,
                      1./9., 1./36.,
                      1./9., 1./36.}; //t: lattice weights
const int e [9][2] = {{0,0}, {1,0}, {1,1},
                     {0,1},{-1,1},{-1,0},
                     {-1,-1}, {0,-1}, {1,-1}}; //e: local lattice velocity vectors

double ux_uy2, ux_uy4, ux_uy6, ux_uy8;
double u_dot_u;
double feq [m][n][9]; //feq: equilibrium distribution function
double ueq [2]; //ueq: equilibrium velocity
const double f1 = 3.;
const double f2 = 9./2.;
const double f3 = 3./2.; //f_: eq. dist. constants

double temp; //temp: temp variable for collide

int i_n, j_n;
double ftemp [m][n][9]; //ftemp: temp streaming eq. dist.

double u_old[m][n][2]; //u_old: previous time step velocity
double diff; //diff: temp. normed error
double u_diff; //u_diff: normed error
```

Initialize all fields and values for memory preallocation

Implementation

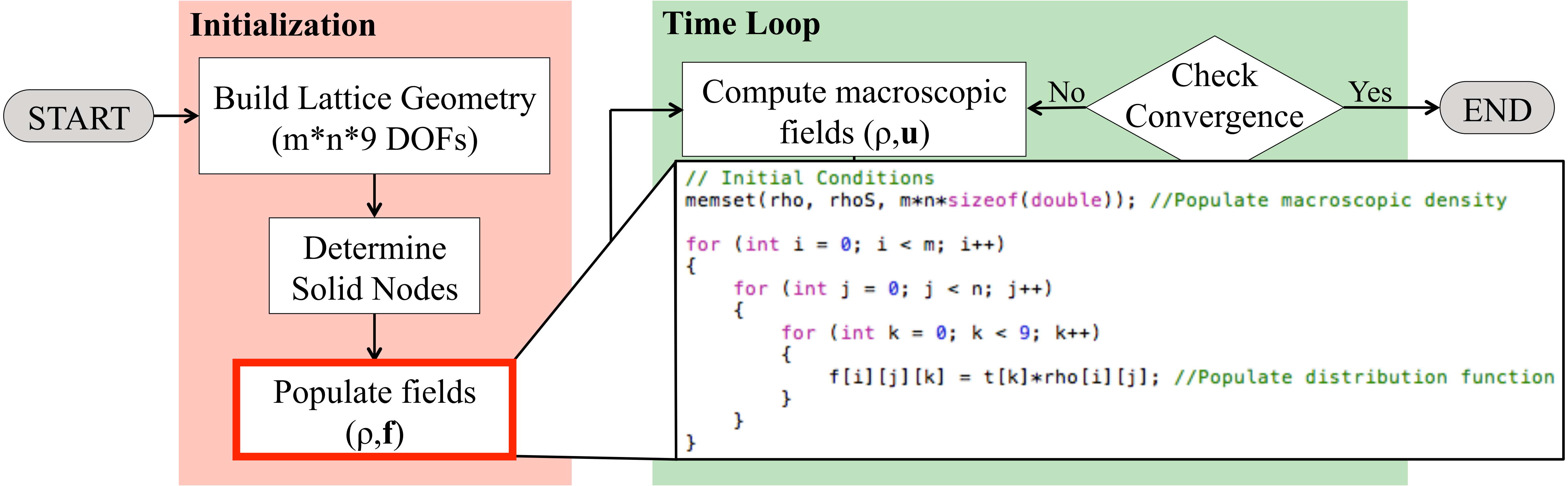



```c
//Build lattice geometry
int count = 0;
memset(isSolid, 0, m*n*sizeof(int));
srand (time(NULL));
for (int j = 0; j < n; j++) //Add solid nodes to geometry
{
    for (int i = 0; i < m; i++)
    {
        if (((double) rand() / (RAND_MAX)) <= distRad)
        {
            for (int ii = -rad; ii <= rad; ii++)
            {
                for (int jj = -rad; jj <= rad; jj++)
                {
                    if (i+ii < 0) //Ensure periodicity of geometry
                        i_p = (m-1)+(1+(i+ii));
                    else if (i+ii > (m-1))
                        i_p = ((i+ii)-(m-1))-1;
                    else
                        i_p = i+ii;

                    if (j+jj < 0) //Ensure periodicity of geometry
                        j_p = (n-1)+(1+(j+jj));
                    else if (j+jj > (n-1))
                        j_p = ((j+jj)-(n-1))-1;
                    else
                        j_p = j+jj;

                    if (sqrt(pow(ii,2)+pow(jj,2)) <= rad + pow(1./rad,2))
                    {
                        isSolid[i_p][j_p] = 1;
                    }
                }
            }
            isSolid[i][j] = 1;
        }
    }
}
```

Ensure periodicity of solid geometry

isSolid[m][n] = Boolean value

Implementation

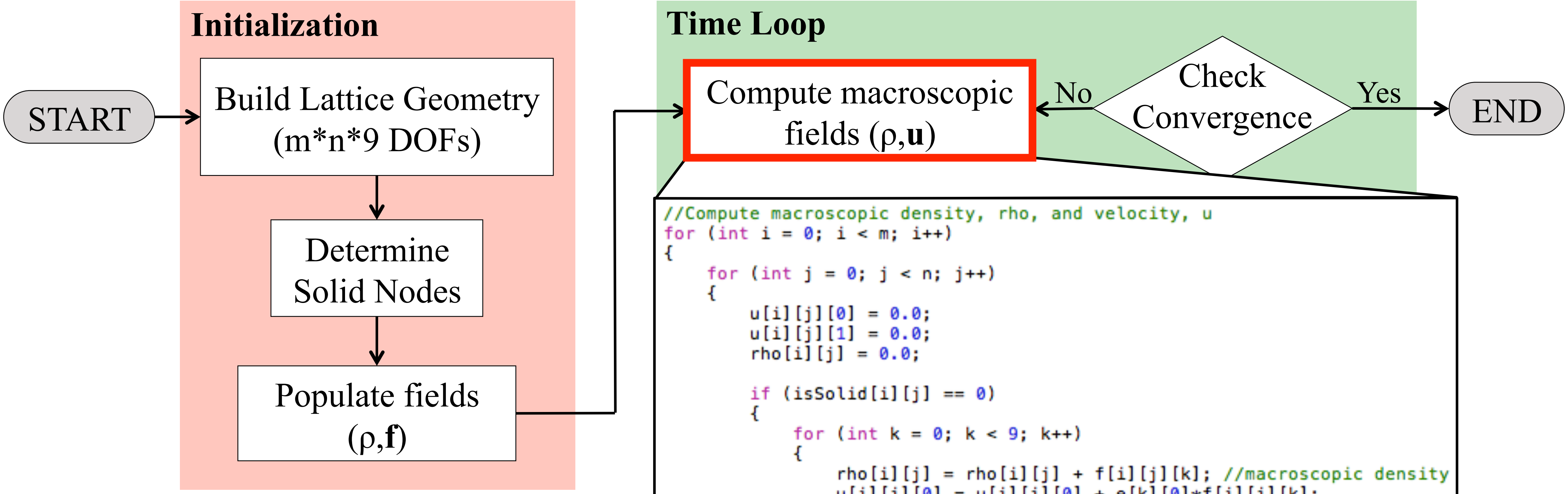



```
// Initial Conditions
memset(rho, rhoS, m*n*sizeof(double)); //Populate macroscopic density

for (int i = 0; i < m; i++)
{
    for (int j = 0; j < n; j++)
    {
        for (int k = 0; k < 9; k++)
        {
            f[i][j][k] = t[k]*rho[i][j]; //Populate distribution function
        }
    }
}
```

rho[m][n] = constant and uniform density

f [m][n][9] = density * lattice weights

Implementation



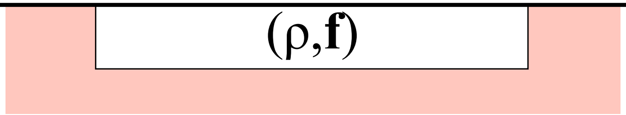

```
//Compute macroscopic density, rho, and velocity, u
for (int i = 0; i < m; i++)
{
    for (int j = 0; j < n; j++)
    {
        u[i][j][0] = 0.0;
        u[i][j][1] = 0.0;
        rho[i][j] = 0.0;

        if (isSolid[i][j] == 0)
        {
            for (int k = 0; k < 9; k++)
            {
                rho[i][j] = rho[i][j] + f[i][j][k]; //macroscopic density
                u[i][j][0] = u[i][j][0] + e[k][0]*f[i][j][k];
                u[i][j][1] = u[i][j][1] + e[k][1]*f[i][j][k];
            }
            u[i][j][0] = u[i][j][0]/rho[i][j]; //macroscopic velocity
            u[i][j][1] = u[i][j][1]/rho[i][j];
        }
    }
}
```

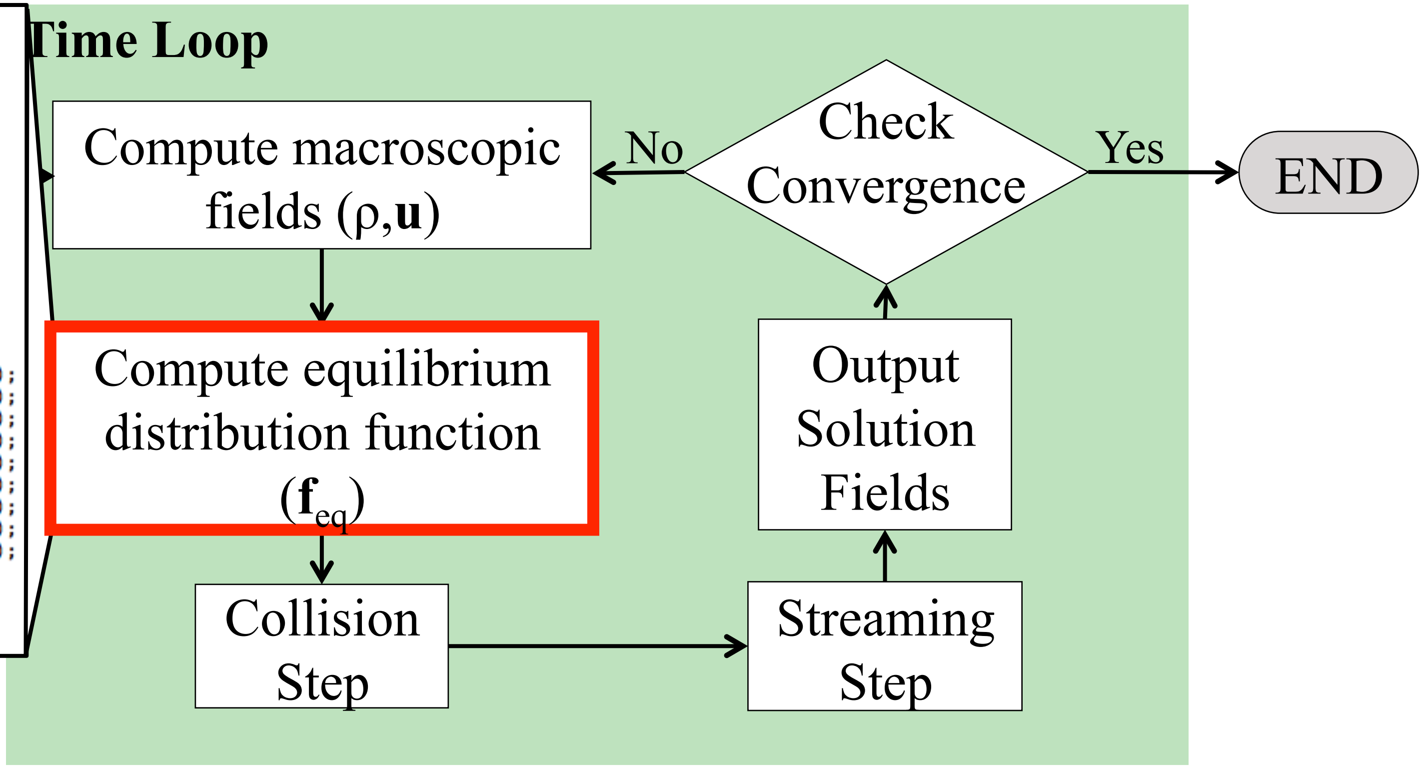
$$\rho = \sum_{\alpha} f_{\alpha}$$

$$\vec{u} = \frac{1}{\rho} \sum_{a} \vec{e}_{\alpha} f_{\alpha}$$

Implementation

```
//Compute equilibrium distribution function
for (int i = 0; i < m; i++)
{
    for (int j = 0; j < n; j++)
    {
        ueq[0] = u[i][j][0];
        ueq[1] = u[i][j][1] - tau*g; //Add forcing term from gravity

        ux_uy2 = ueq[0] + ueq[1];
        ux_uy4 = -ueq[0] + ueq[1];
        ux_uy6 = -ueq[0] - ueq[1];
        ux_uy8 = ueq[0] - ueq[1];
        u_dot_u = pow(ueq[0],2) + pow(ueq[1],2);

        if (isSolid[i][j] == 0)
        {
            feq[i][j][0] = rho[i][j]*t[0]*(1 + 0          + 0                  - f3*u_dot_u);
            feq[i][j][1] = rho[i][j]*t[1]*(1 + f1*ueq[0]  + f2*pow(ueq[0],2)   - f3*u_dot_u);
            feq[i][j][2] = rho[i][j]*t[2]*(1 + f1*ux_uy2  + f2*ux_uy2*ux_uy2   - f3*u_dot_u);
            feq[i][j][3] = rho[i][j]*t[3]*(1 + f1*ueq[1]  + f2*pow(ueq[1],2)   - f3*u_dot_u);
            feq[i][j][4] = rho[i][j]*t[4]*(1 + f1*ux_uy4  + f2*ux_uy4*ux_uy4   - f3*u_dot_u);
            feq[i][j][5] = rho[i][j]*t[5]*(1 - f1*ueq[0]  + f2*pow(ueq[0],2)   - f3*u_dot_u);
            feq[i][j][6] = rho[i][j]*t[6]*(1 + f1*ux_uy6  + f2*ux_uy6*ux_uy6   - f3*u_dot_u);
            feq[i][j][7] = rho[i][j]*t[7]*(1 - f1*ueq[1]  + f2*pow(ueq[1],2)   - f3*u_dot_u);
            feq[i][j][8] = rho[i][j]*t[8]*(1 + f1*ux_uy8  + f2*ux_uy8*ux_uy8   - f3*u_dot_u);
            //Note here c = delta_x/delta_t = 1 and is not included
        }
    }
}
```



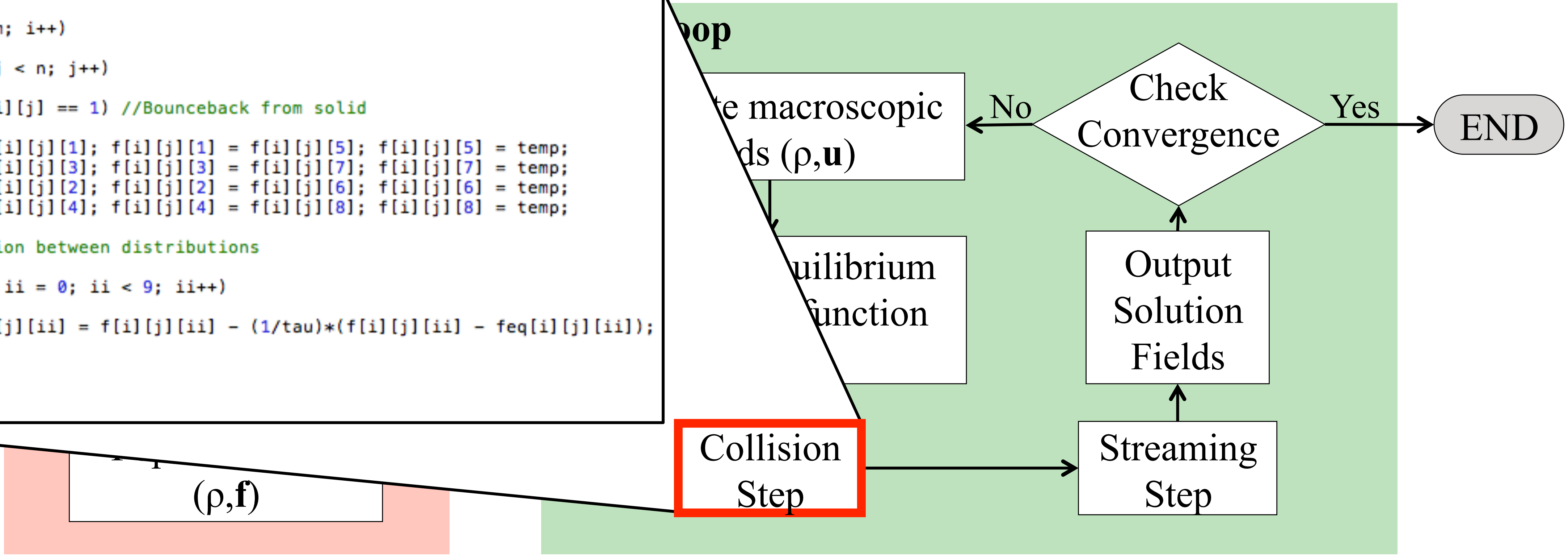


Add gravity: $\vec{u} = \vec{u} + \tau \vec{g}$

$$f_{eq} = w_\alpha \rho \left[1 + 3\frac{\vec{e}_\alpha \cdot \vec{u}}{c^2} + \frac{9}{2}\frac{(\vec{e}_\alpha \cdot \vec{u})^2}{c^4} - \frac{3}{2}\frac{\vec{u}^2}{c^2} \right]$$

Implementation

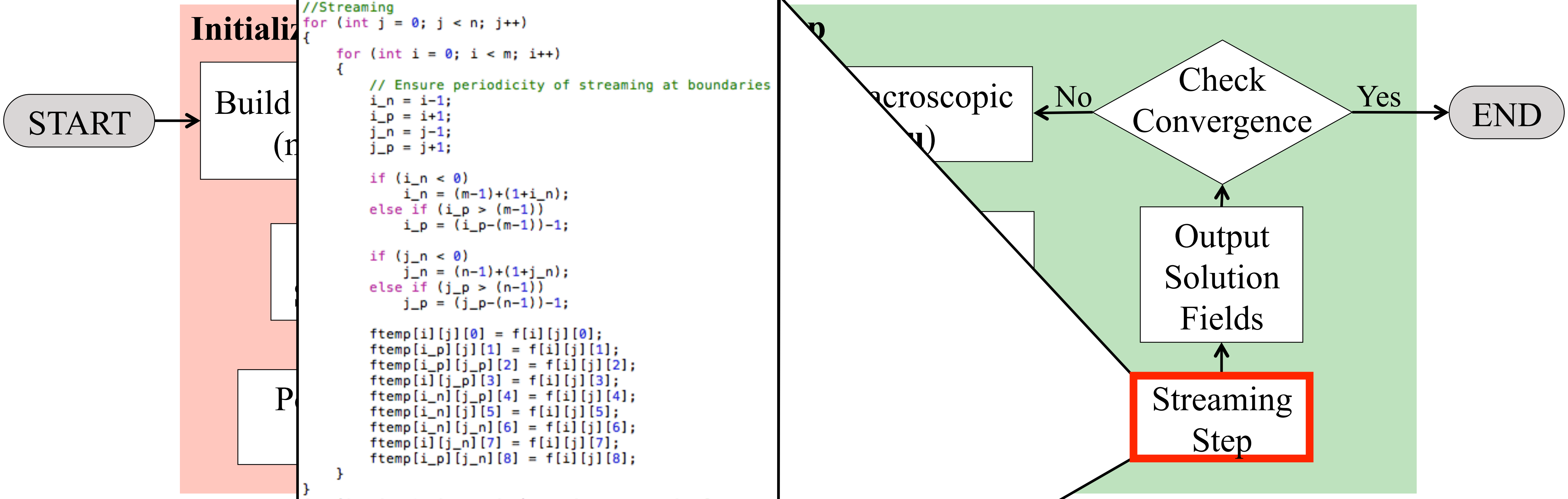
```
//Collision
for (int i = 0; i < m; i++)
{
    for (int j = 0; j < n; j++)
    {
        if (isSolid[i][j] == 1) //Bounceback from solid
        {
            temp = f[i][j][1]; f[i][j][1] = f[i][j][5]; f[i][j][5] = temp;
            temp = f[i][j][3]; f[i][j][3] = f[i][j][7]; f[i][j][7] = temp;
            temp = f[i][j][2]; f[i][j][2] = f[i][j][6]; f[i][j][6] = temp;
            temp = f[i][j][4]; f[i][j][4] = f[i][j][8]; f[i][j][8] = temp;
        }
        else //Collsion between distributions
        {
            for (int ii = 0; ii < 9; ii++)
            {
                f[i][j][ii] = f[i][j][ii] - (1/tau)*(f[i][j][ii] - feq[i][j][ii]);
            }
        }
    }
}
```



Bounceback condition on solid

$$f = f - \frac{1}{\tau}(f - f_{eq})$$

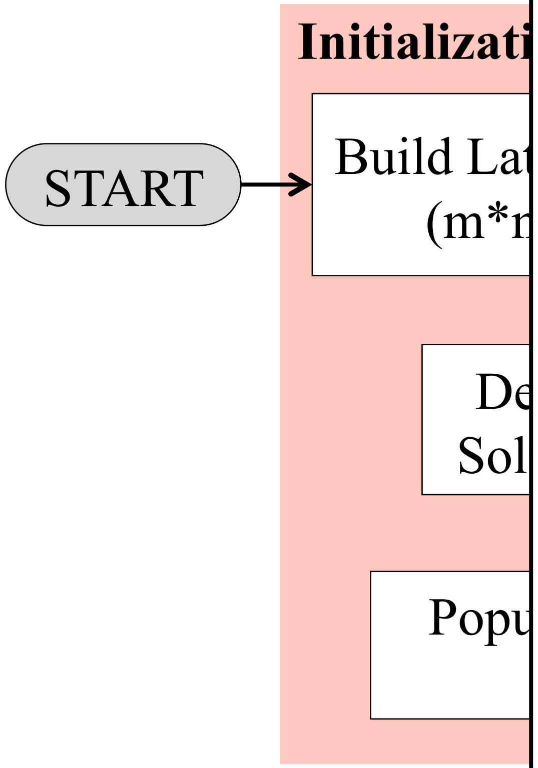
Implementation



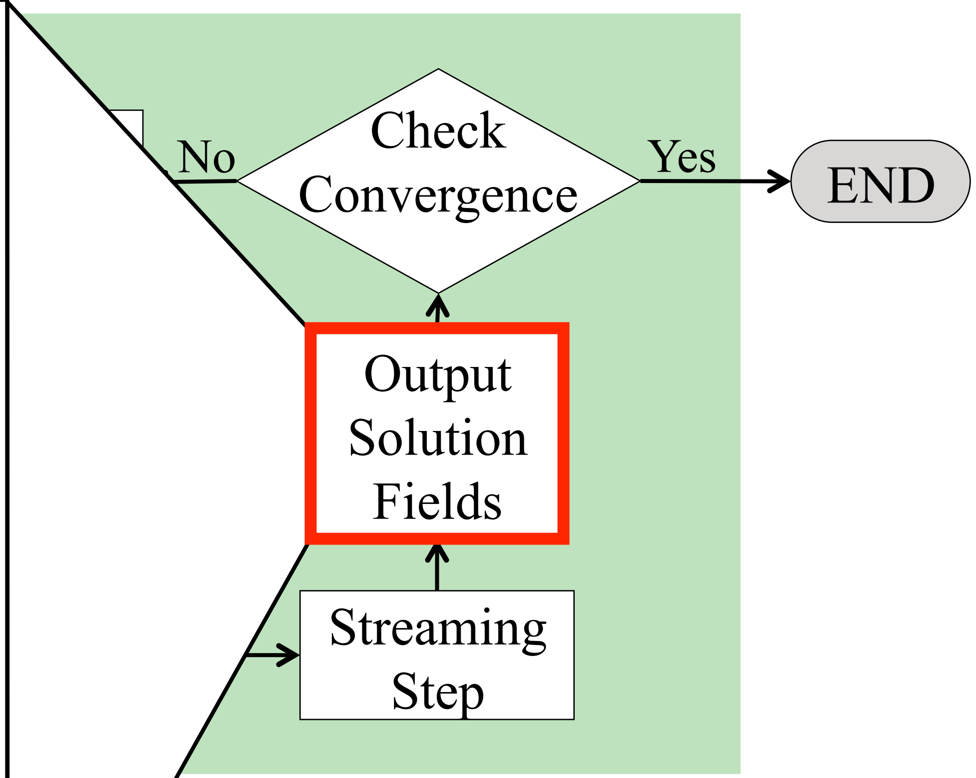

```
//Streaming
for (int j = 0; j < n; j++)
{
    for (int i = 0; i < m; i++)
    {
        // Ensure periodicity of streaming at boundaries
        i_n = i-1;
        i_p = i+1;
        j_n = j-1;
        j_p = j+1;

        if (i_n < 0)
            i_n = (m-1)+(1+i_n);
        else if (i_p > (m-1))
            i_p = (i_p-(m-1))-1;

        if (j_n < 0)
            j_n = (n-1)+(1+j_n);
        else if (j_p > (n-1))
            j_p = (j_p-(n-1))-1;

        ftemp[i][j][0] = f[i][j][0];
        ftemp[i_p][j][1] = f[i][j][1];
        ftemp[i_p][j_p][2] = f[i][j][2];
        ftemp[i][j_p][3] = f[i][j][3];
        ftemp[i_n][j_p][4] = f[i][j][4];
        ftemp[i_n][j][5] = f[i][j][5];
        ftemp[i_n][j_n][6] = f[i][j][6];
        ftemp[i][j_n][7] = f[i][j][7];
        ftemp[i_p][j_n][8] = f[i][j][8];
    }
}
for (int i = 0; i < m; i++) //Update streamed values
{
    for (int j = 0; j < n; j++)
    {
        for (int ii = 0; ii < 9; ii++)
        {
            f[i][j][ii] = ftemp[i][j][ii];
        }
    }
}
```

Ensure periodicity of streaming at boundaries

Implementation

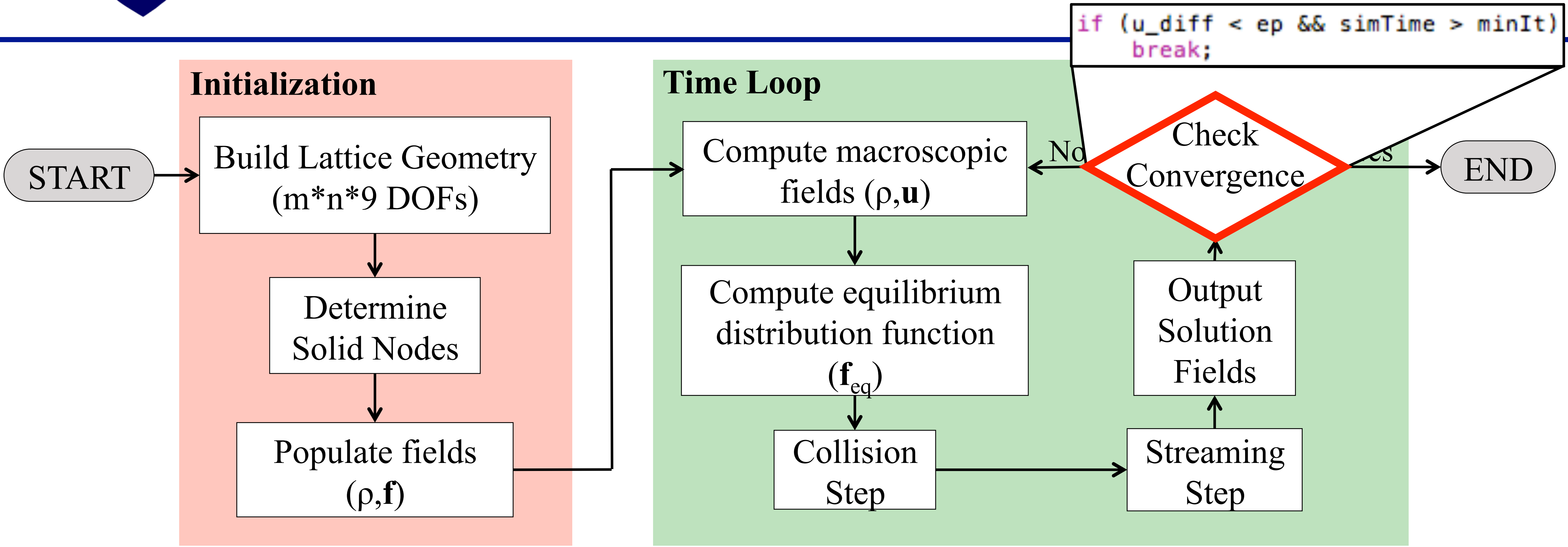



```
//Postprocessing
ofstream fout;
std::ostringstream fileNameStream("u_x");
fileNameStream << "u_x_" << simTime << ".csv";
std::string fileName = fileNameStream.str();
fout.open (fileName.c_str());
for (int j = 0; j < n; j++) //Visualize velocity field
{
    for (int i = 0; i < m; i++)
    {
        if (i == m-1)
            fout << u[i][j][0] << endl;
        else
            fout << u[i][j][0] << ",";
    }
}
fout.close();

ofstream fout2;
std::ostringstream fileNameStream2("u_y");
fileNameStream2 << "u_y_" << simTime << ".csv";
std::string fileName2 = fileNameStream2.str();
fout2.open (fileName2.c_str());
for (int j = 0; j < n; j++) //Visualize velocity field
{
    for (int i = 0; i < m; i++)
    {
        if (i == m-1)
            fout2 << u[i][j][1] << endl;
        else
            fout2 << u[i][j][1] << ",";
    }
}
fout2.close();
```



Output velocity CSV files to visualize in Matlab

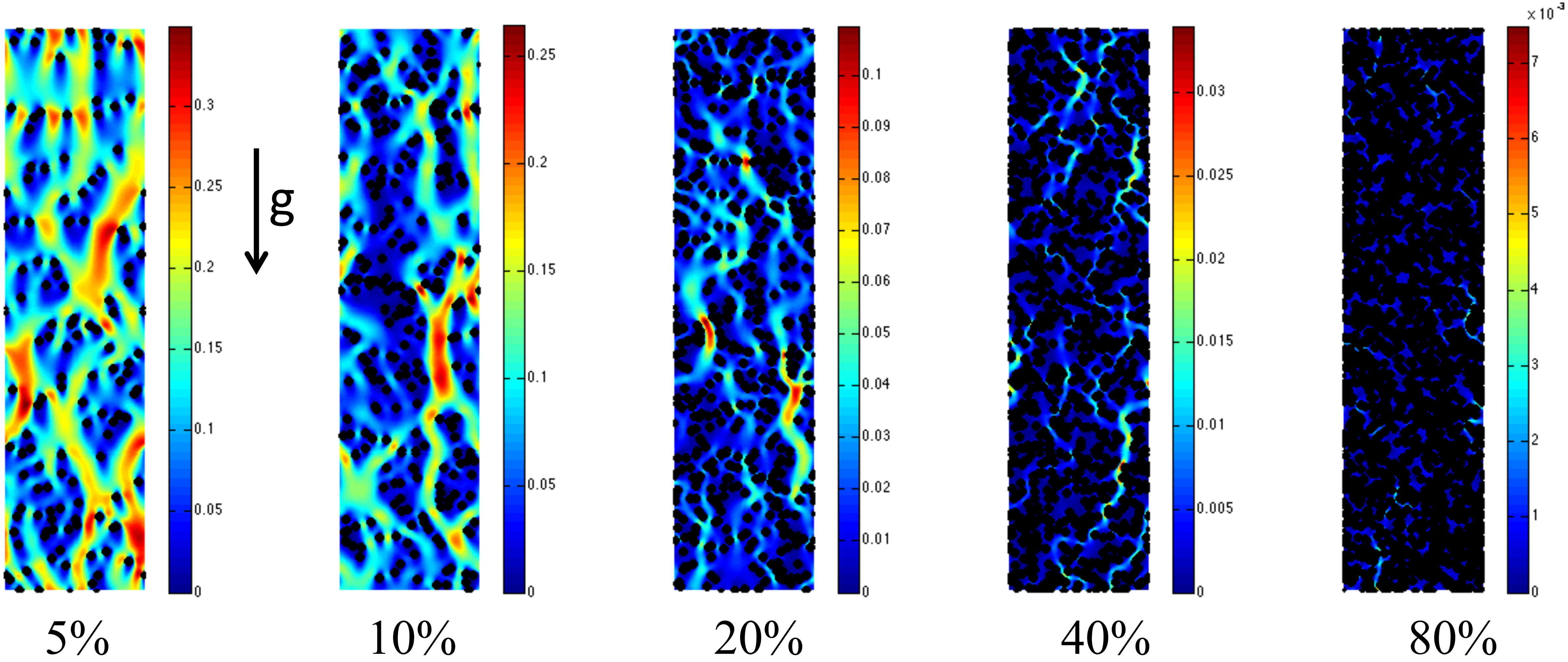
Implementation



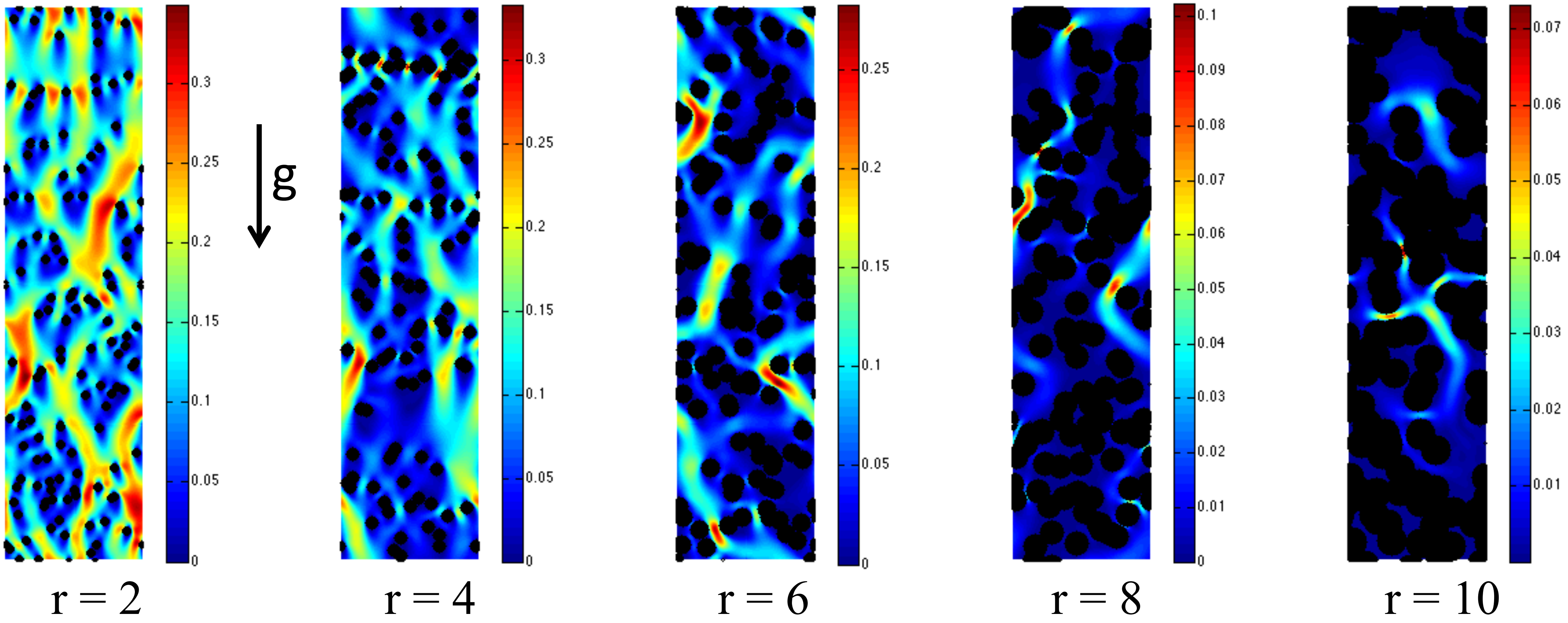
$$\text{iter} < \text{maxIter}$$

$$\frac{\left|\vec{u} - \vec{u}_{old}\right|}{\left|\vec{u}\right|} < \epsilon$$

Varying Grain Density (r = 2)

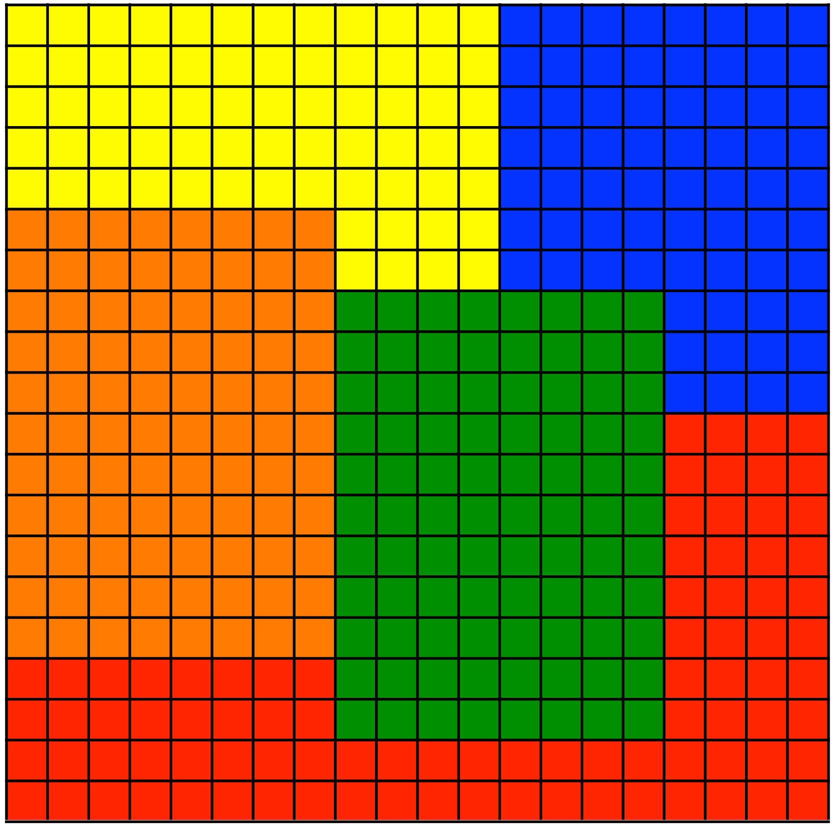


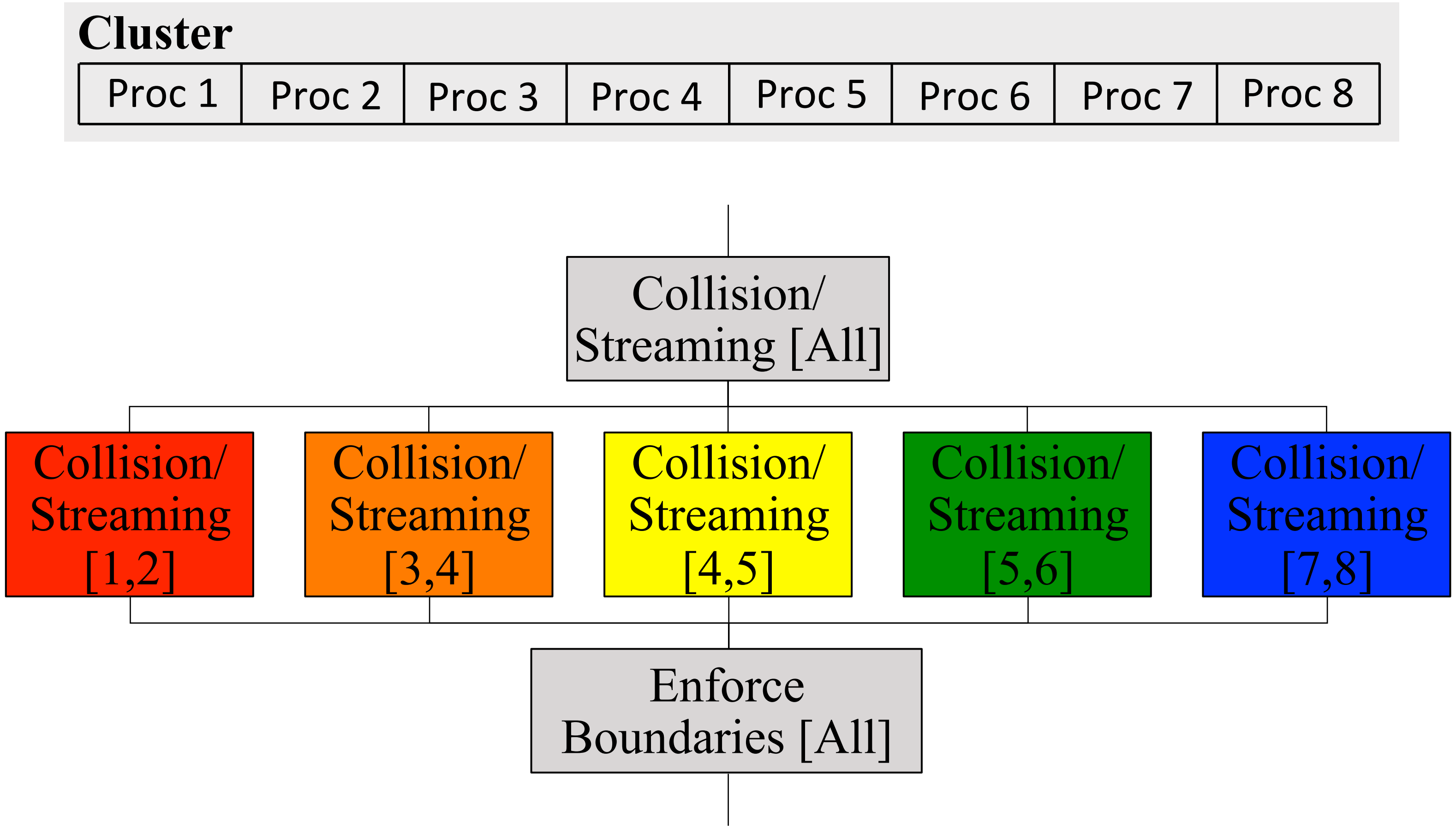
Varying Grain Size (5%)



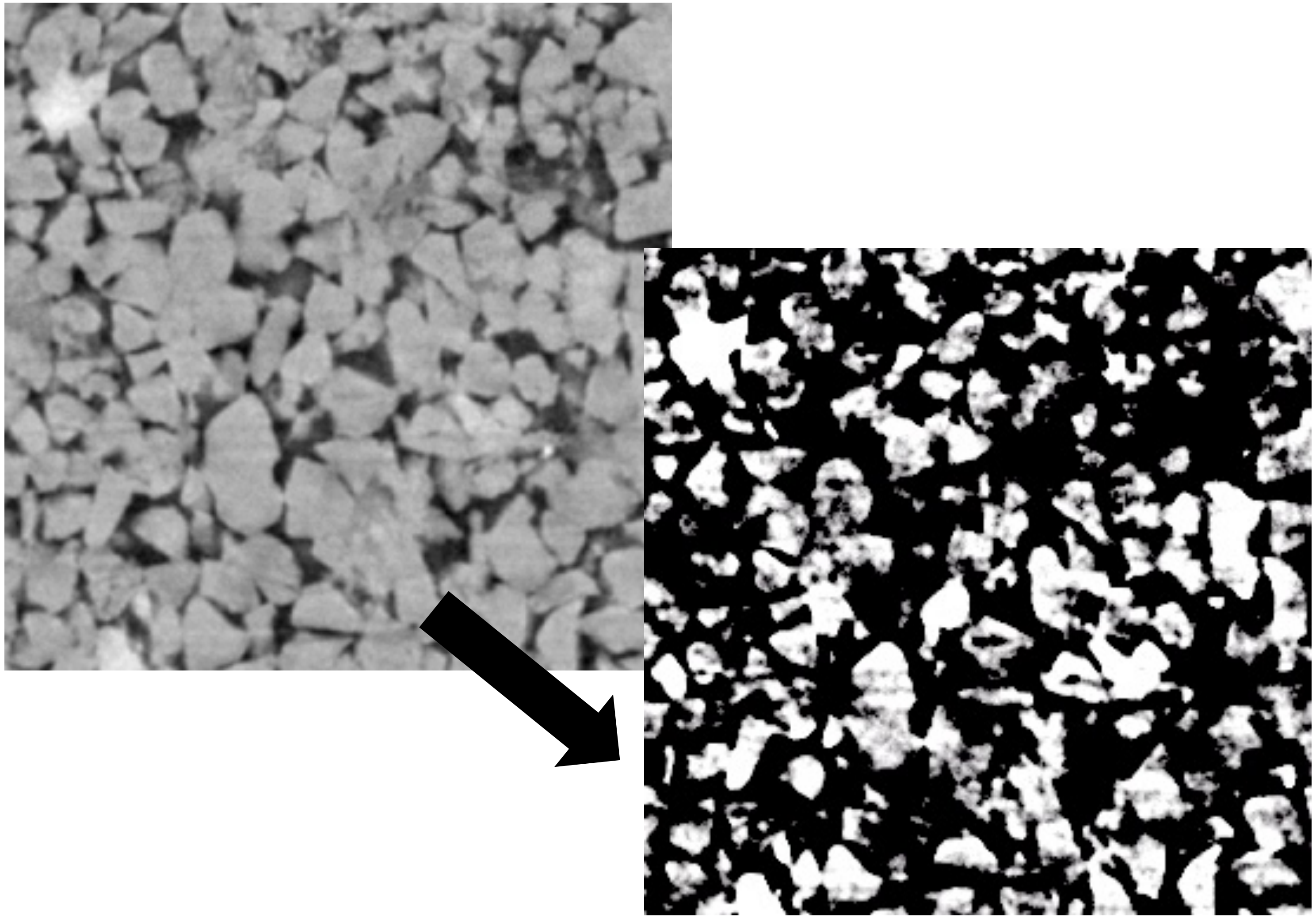
Non-Uniform Porosity

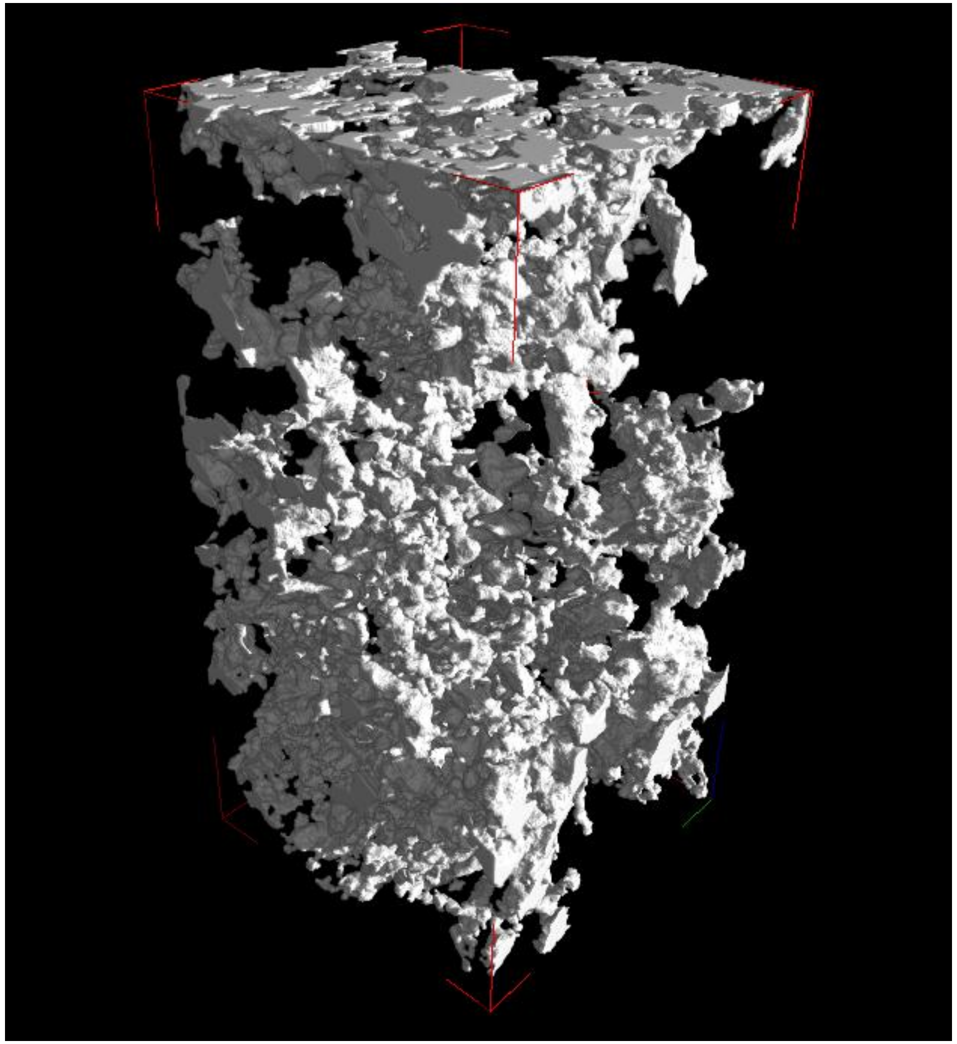
Parallelism (Domain Decomposition)





Rock Sample Tomography





Example Applications

Fancy stuff we can do with LBM

Examples

- Air Conditioner
  - LBM simulates two conditions
    - 1. Fixed Fan Air Conditioner
    - 2. Sweeping Fan Air Conditioner
  - Parallelism
    - Extremely High Resolution Simulation Required
    - Parallel computations allow LBM to be extremely efficient with reasonable computer hardware
    - Allowed for the billion grid points required to accurately simulate this case

http://youtu.be/I82uCa7SHSQ?t=21m20s

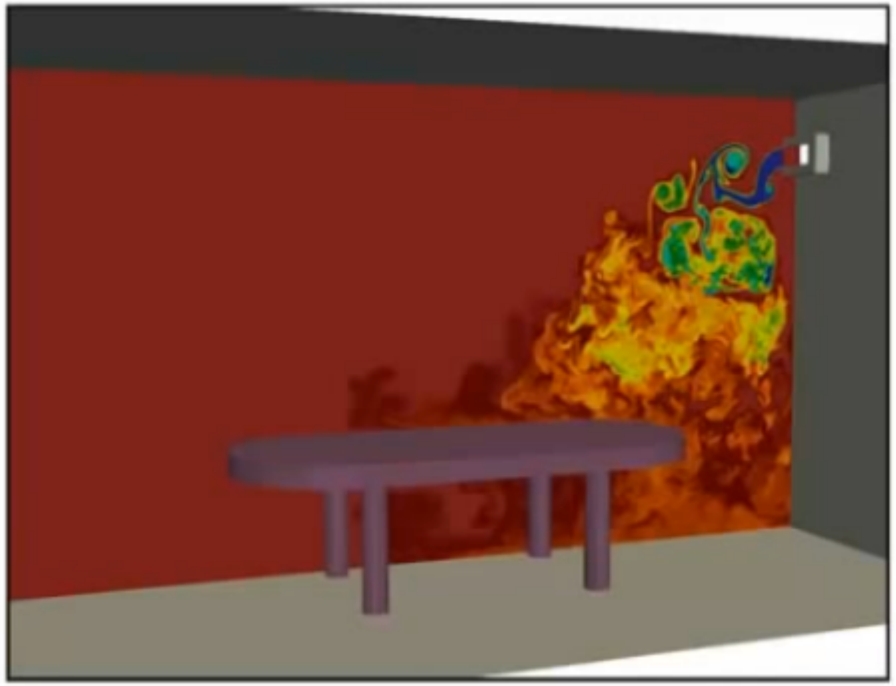


Image: https://www.youtube.com/watch?v=I82uCa7SHSQ

Examples

- Blood Clotting in a Human Artery
  - Arteries that have been affected by disease can be at high risk to rupture.
  - These ruptures can possibly be prevented by blood clotting in the vulnerable area
  - Need to simulate red blood cell changing from liquid to solid behavior and stick to artery wall.
    - LBM is effective at achieving this because of its hybrid particle/continuum nature

Image: https://www.youtube.com/watch?v=I82uCa7SHSQ

Examples

- Blood Clotting in a Human Artery
  - Parallelism
    - Using LBM, it is very easy to send different calculations to different processors
    - Allows for high efficiency when using computers with a high number of processors

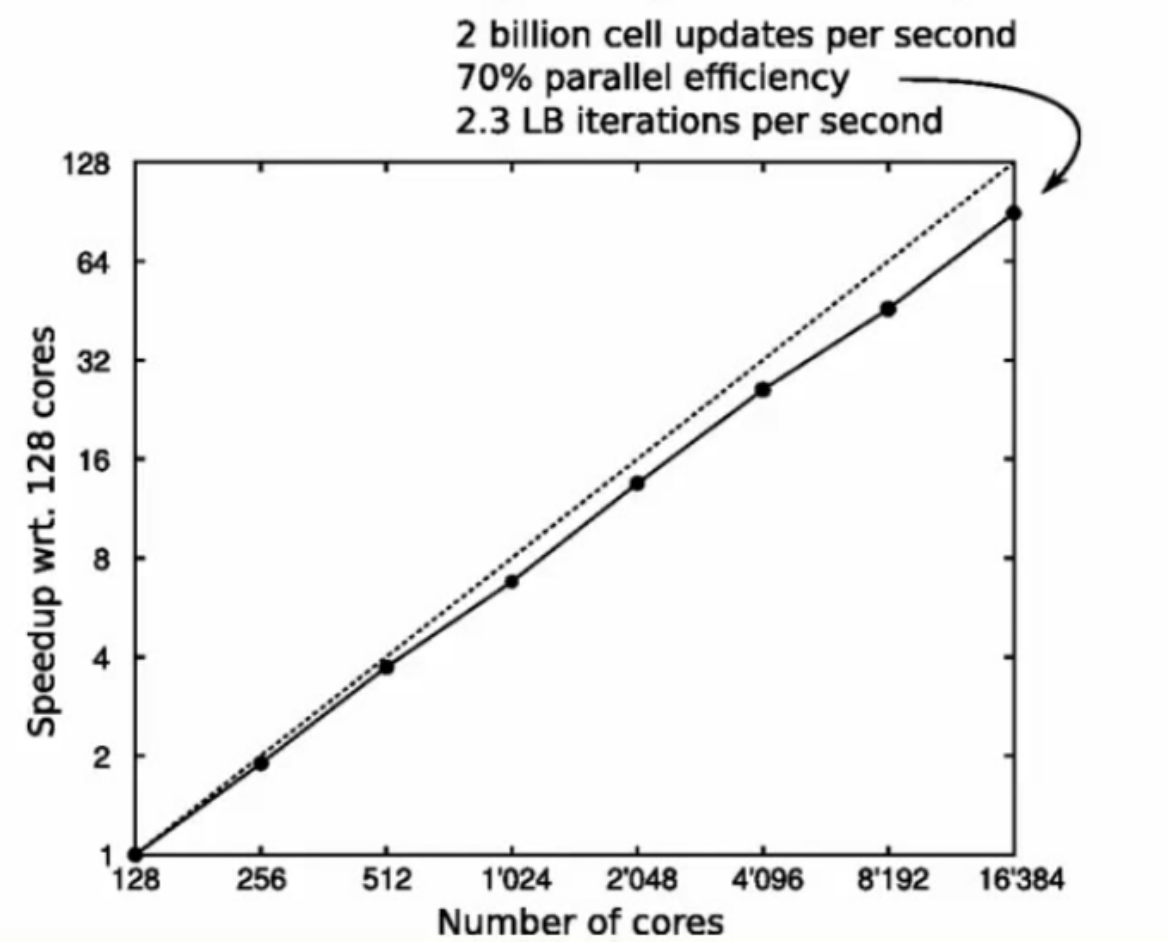


Image: https://www.youtube.com/watch?v=I82uCa7SHSQ

Examples

- Turbulence Modeling
  - Models flow between two parallel plates
  - Large- eddy simulation approach
  - Replace Lattice Boltzmann equation with a filtered form
- Comparison to Spectral Method
  - Solution is virtually identical
  - LBM able to simulate with a 200x reduction in resolution
  - LBM much less computationally intensive in this case

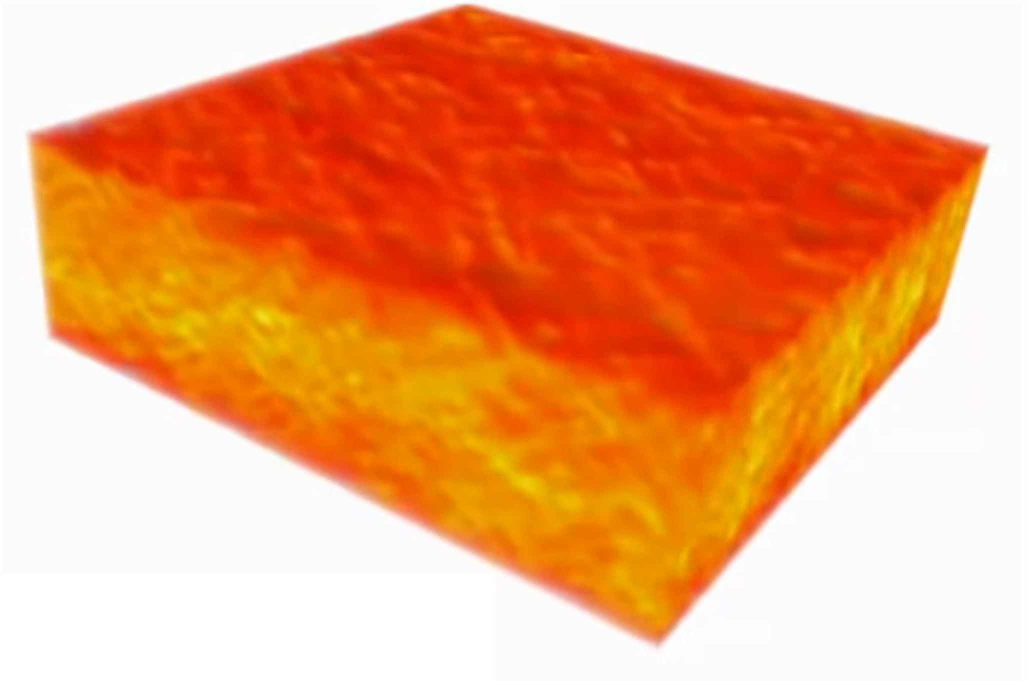


Image: https://www.youtube.com/watch?v=I82uCa7SHSQ

References

- Latt, J. (2013). *Introduction to Lattice Boltzmann Method @ NasaGlenn 2013*
- [Video]. https://www.youtube.com/watch?v=I82uCa7SHSQ
- Mele, I. (2013). *Lattice Boltzmann Method.* Univerza v Ljubljani.
- Fathi, Ebrahim, and I. Yucel Akkutlu. "Lattice Boltzmann method for simulation of shale gas transport in kerogen." *SPE Journal* 18.01 (2012): 27-37.
- Engler, Simon T. "Benchmarking the 2D lattice Boltzmann BGK model." *Short communication. Amsterdam Center for Computacional Science, Amsterdam, The Netherlands* (2003).
- Bao, Yuanxun Bill, and Justin Meskas. "Lattice Boltzmann Method for Fluid Simulations." (2011).
- Adhvaryu, Chinmay. "The Lattice Boltzmann Method For Computational Fluid Dynamics Applications." (2008).
- Xiaoyi He, Li-shi Luo. " Theory of the lattice Boltzmann method: From the Boltzmann equation to the lattice Boltzmann equation." (1997).

Thank You!

Questions?