TU Clausthal

The lattice-Boltzmann Method

Practical aspects and implementation

Gunther Brenner

Institute of Applied Mechanics
Clausthal University

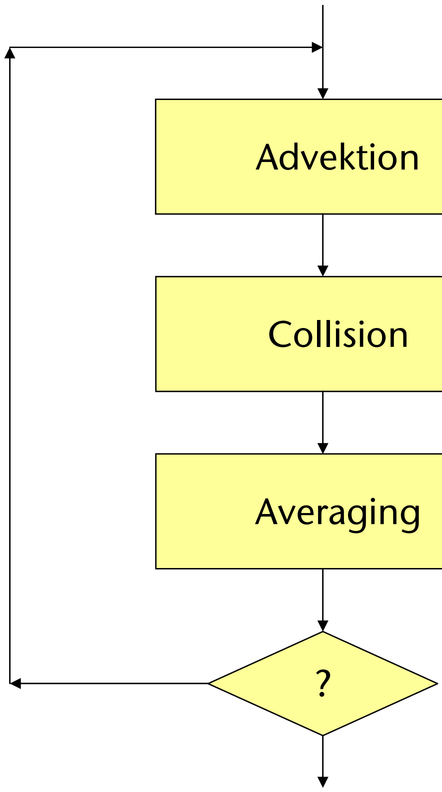
Antalya, May 2007

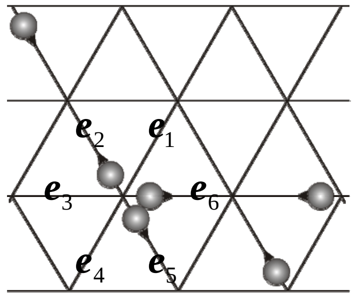
TU Clausthal

Part 2: LBM in practice

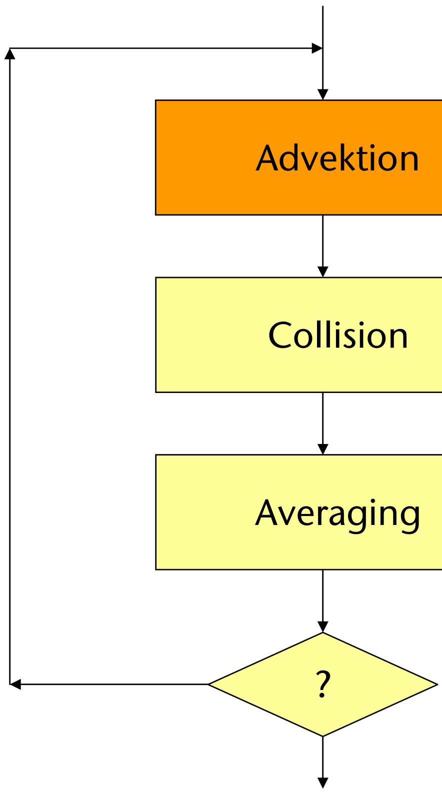
- Lattice Boltzmann algorithm
- Boundary Conditions
- Implementation

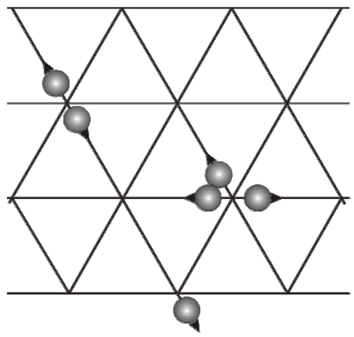
Contents





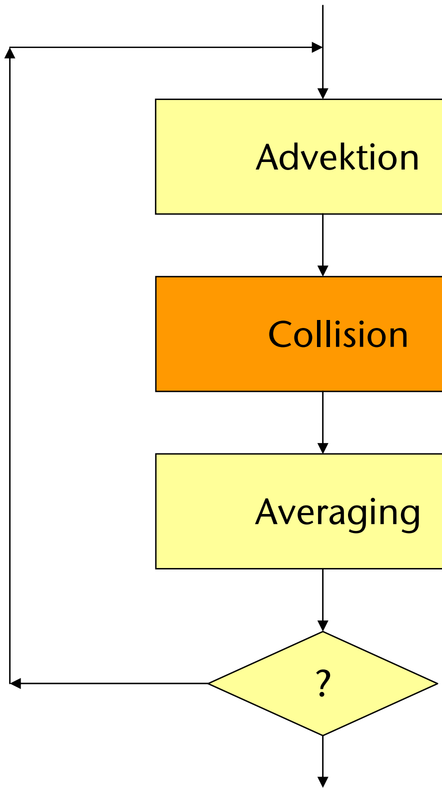
$$\boldsymbol{e}_\alpha = \begin{pmatrix} \cos \frac{\pi}{3}\alpha \\ \sin \frac{\pi}{3}\alpha \end{pmatrix} \qquad n_\alpha(t, \boldsymbol{x})$$

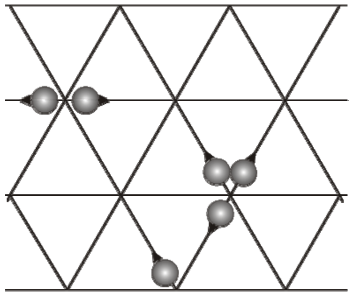




$$n_\alpha^*(t+\Delta t, \boldsymbol{x}+\boldsymbol{e}_\alpha \Delta t) = n_\alpha(t, \boldsymbol{x})$$

TU Clausthal



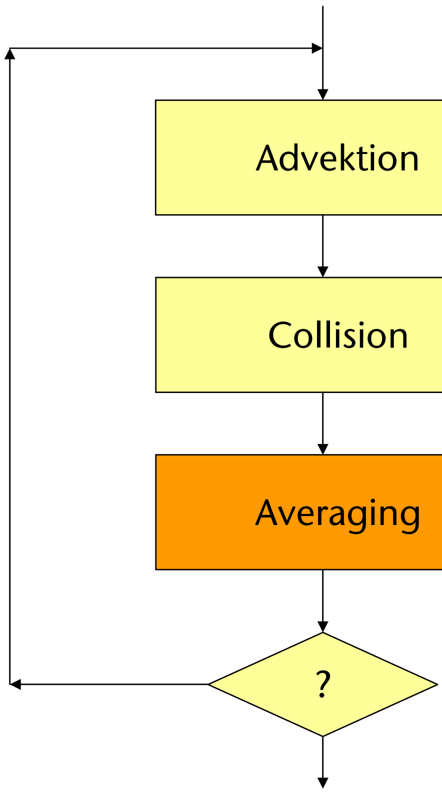


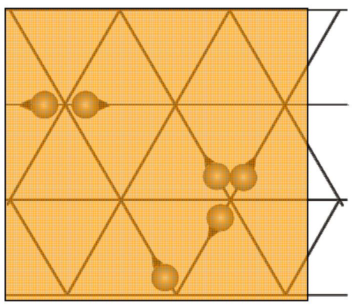
$$n^*_\alpha(t+\Delta t, \boldsymbol{x}+\boldsymbol{e}_\alpha \Delta t) = n_\alpha(t, \boldsymbol{x})$$

$$n_\alpha = \Omega_\alpha(n^*_1, \ldots, n^*_6)$$

LGA Algorithm

TU Clausthal





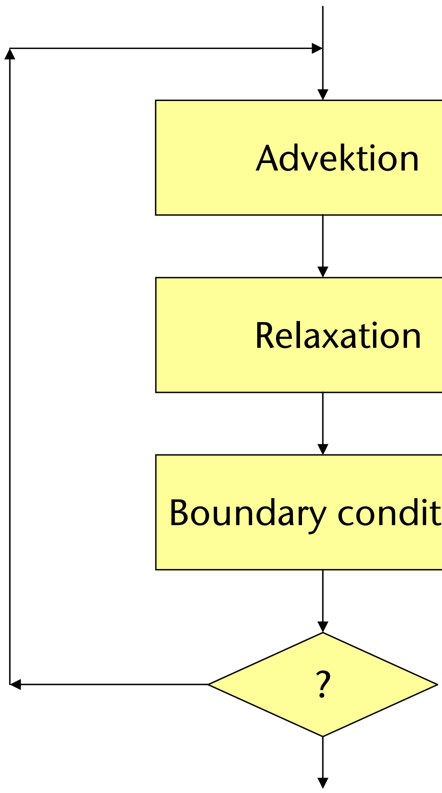
$$f_\alpha = \langle n_\alpha \rangle$$

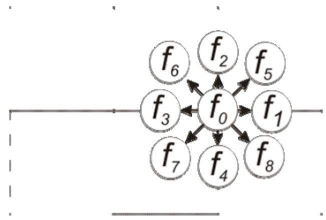
$$\rho = \sum_\alpha f_\alpha$$

$$\rho \boldsymbol{u} = \sum_\alpha \boldsymbol{e}_\alpha f_\alpha$$

LGA Algorithm

TU Clausthal

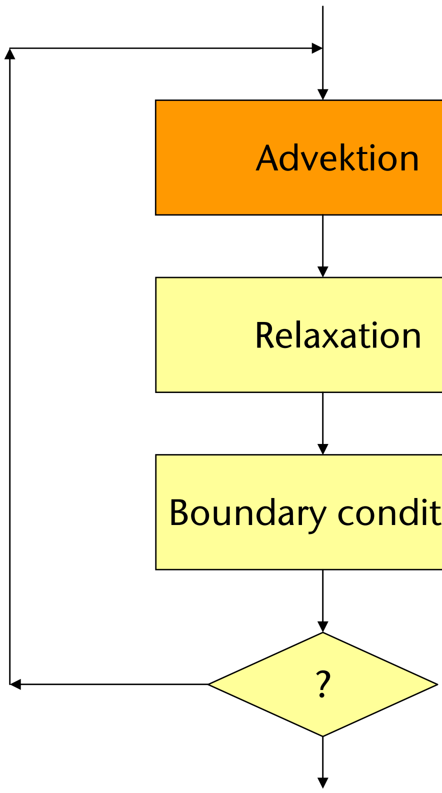


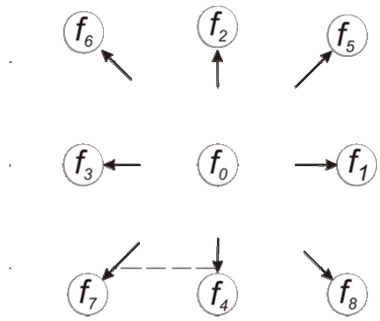


$$f_\alpha(t, \boldsymbol{x})$$

LBM Algorithm

TU Clausthal

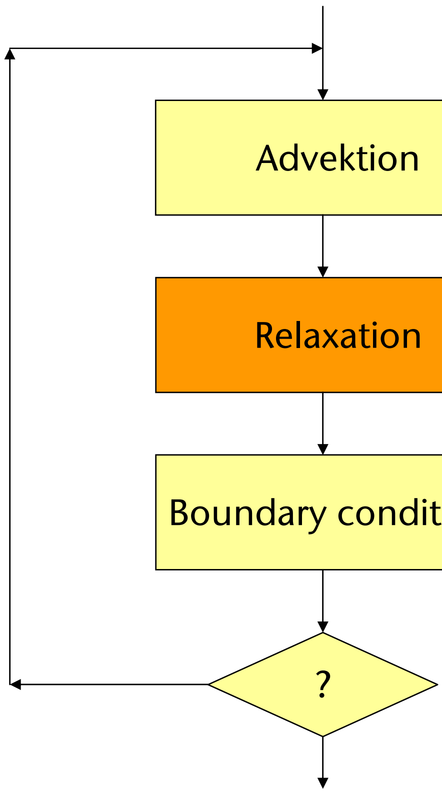


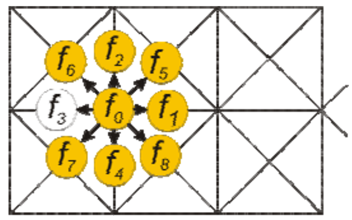


$$f_\alpha^*(t+\Delta t, \boldsymbol{x}+\boldsymbol{e}_\alpha \Delta t) = f_\alpha(t, \boldsymbol{x})$$

LBM Algorithm

TU Clausthal



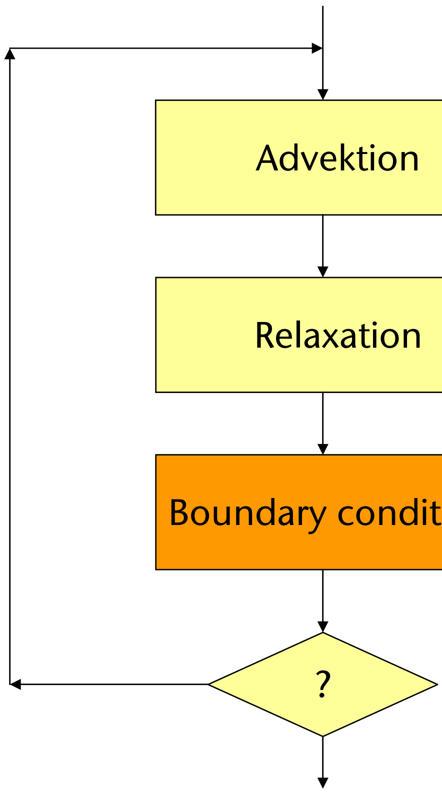


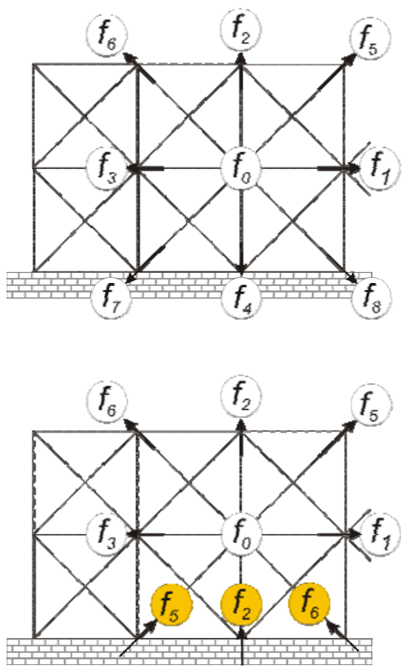
$$f^*_\alpha(t+\Delta t, \boldsymbol{x}+\boldsymbol{e}_\alpha \Delta t) = f_\alpha(t, \boldsymbol{x})$$

$$f_\alpha \equiv \frac{1}{\tau}\left(f^*{}_\alpha - f^{eq}_\alpha\right)$$

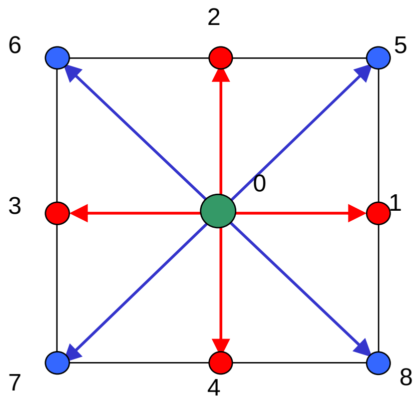
LBM Algorithm

TU Clausthal

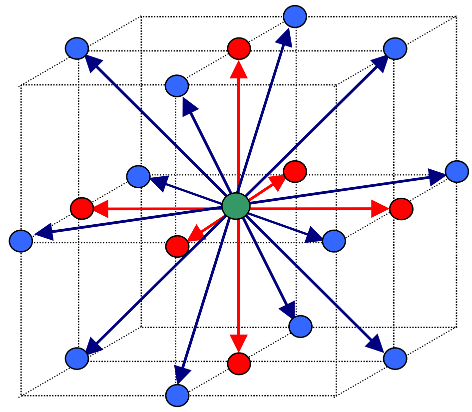


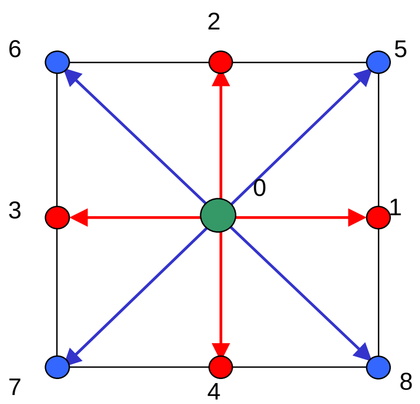


LBM Algorithm



D2Q9


D3Q19



D2Q9

$$\boldsymbol{e}_0 = (0,0)$$
$$\boldsymbol{e}_1 = (1,0)$$
$$\boldsymbol{e}_2 = (0,1)$$
$$\boldsymbol{e}_3 = (-1,0)$$
$$\boldsymbol{e}_4 = (0,-1)$$
$$\boldsymbol{e}_5 = (1,1)$$
$$\boldsymbol{e}_6 = (-1,1)$$
$$\boldsymbol{e}_7 = (-1,-1)$$
$$\boldsymbol{e}_8 = (1,-1)$$

$$\Delta x = \Delta y = \Delta t = 1$$
$$\left|\boldsymbol{e}_{1,2,3,4}\right| = 1 \qquad c_s^2 = \frac{1}{3}$$
$$\left|\boldsymbol{e}_{5,6,7,8}\right| = \sqrt{2}$$

- equilibrium distributions

$$f_\alpha^{eq}(\rho,\boldsymbol{u}) = t_p \cdot \rho \cdot \left[1 + 3(\boldsymbol{e}_\alpha \cdot \boldsymbol{u}) + \frac{9}{2}(\boldsymbol{e}_\alpha \cdot \boldsymbol{u})^2 - \frac{3}{2}\boldsymbol{u}^2\right]$$

	t_0	t_1	t_2	t_3
$D2Q9$	$\frac{4}{9}$	$\frac{1}{9}$	$\frac{1}{36}$	0
$D3Q15$	$\frac{2}{9}$	$\frac{1}{9}$	0	$\frac{1}{72}$
$D3Q19$	$\frac{1}{3}$	$\frac{1}{18}$	$\frac{1}{36}$	0

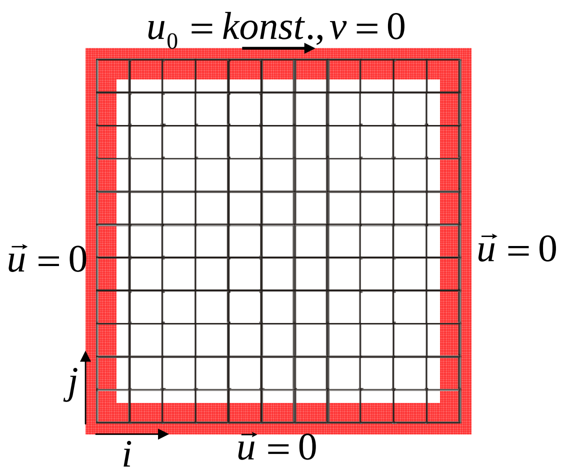
- viscosity $\nu = \frac{1}{3}\left(\tau - \frac{1}{2}\right)$ $\omega = \frac{1}{\tau}$
- density $\rho = \sum_\alpha f_\alpha$
- velocity $\boldsymbol{u} = \frac{1}{\rho}\sum_\alpha \boldsymbol{e}_\alpha f_\alpha$
- pressure $p = \frac{1}{3}\rho$

TU Clausthal

- Program `lbm_ldc.f`

Input:
```
itmax   Number Iterationen
vis     viscosity
force    "forcing term"
```

Boundaries:

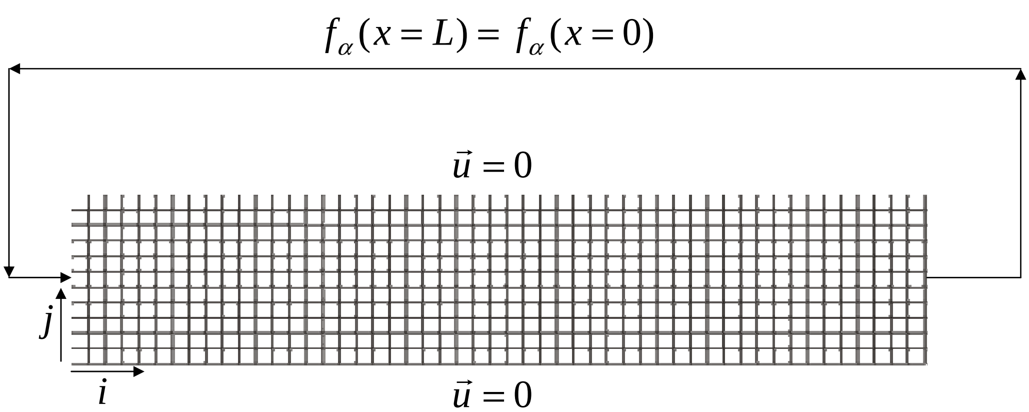


example 1: „lid driven cavity“

TU Clausthal

- Program `lbm_ldc.f`

modify boundary conditions:



example 2: „Hagen-Couette Flow“

```
c ----------------------------------------------------
C     input parameter from stdin
c ----------------------------------------------------

      read(5,*) itmax
      read(5,*) vis
      read(5,*) force

      omega  = 1./(0.5 + 3.0*vis )
      eomega = 1.0-omega
      dens   = 1.0
      t1x    = force/6.0
```



```
C init with equilibrium distribution function for zero velocity

      t0  = dens*4.d0/9.d0
      t1  = dens*1.d0/9.d0
      t2  = dens*1.d0/36.d0

      do j=1,jmax
         do i=1,imax
            f(0,i,j)=t0
            f(1,i,j)=t1
            f(2,i,j)=t1
            f(3,i,j)=t1
            f(4,i,j)=t1
            f(5,i,j)=t2
            f(6,i,j)=t2
            f(7,i,j)=t2
            f(8,i,j)=t2
         enddo
      enddo
```

$$f_\alpha^{eq}(\rho,\boldsymbol{u}) = t_p \cdot \rho \cdot \left[1 + 3(\boldsymbol{e}_\alpha \cdot \boldsymbol{u}) + \frac{9}{2}(\boldsymbol{e}_\alpha \cdot \boldsymbol{u})^2 - \frac{3}{2}\boldsymbol{u}^2\right]$$

TU Clausthal

```
C propagation step, assuming periodic boundary conditions

         do i=1,imax
            do j=1,jmax

               ie = mod(i,imax) + 1
               iw = imax - mod(imax+1-i,imax)
               jn = mod(j,jmax) + 1
               js = jmax - mod(jmax+1-j,jmax)

               fn(1,ie ,j ) = f(1,i,j)
               fn(2,i  ,jn) = f(2,i,j)
               fn(3,iw ,j ) = f(3,i,j)
               fn(4,i  ,js) = f(4,i,j)
               fn(5,ie ,jn) = f(5,i,j)
               fn(6,iw ,jn) = f(6,i,j)
               fn(7,iw ,js) = f(7,i,j)
               fn(8,ie ,js) = f(8,i,j)
               fn(0,i  ,j ) = f(0,i,j)
            enddo
         enddo
```

Program

TU Clausthal

```
c boundary conditions: bounce back
C lower wall j=1

         j=1
         do i=1,imax
            f(0,i,j) =  fn(0,i,j)
            f(1,i,j) =  fn(3,i,j)
            f(2,i,j) =  fn(4,i,j)
            f(3,i,j) =  fn(1,i,j)
            f(4,i,j) =  fn(2,i,j)
            f(5,i,j) =  fn(7,i,j)
            f(6,i,j) =  fn(8,i,j)
            f(7,i,j) =  fn(5,i,j)
            f(8,i,j) =  fn(6,i,j)
         enddo
```

Program

TU Clausthal

```
c north wall: add source term to drive flow in x-direction

          j=jmax
          do i=1,imax
             f(7,i,j) =  f(7,i,j) -  t1x
             f(8,i,j) =  f(8,i,j) +  t1x
          enddo
```

Program

TU Clausthal

```
c relaxation step
  t0   = 4.d0/9.d0
  t1   = 1.d0/9.d0
  t2   = 1.d0/36.d0
  do j = 2,jmax-1
    do i = 2,imax-1
      u = fn(1,i,j) + fn(5,i,j) + fn(8,i,j)   ..
      v = fn(5,i,j) + fn(2,i,j) + fn(6,i,j)   ..
      r = fn(0,i,j) + fn(1,i,j) + fn(2,i,j)   ..
      u = u/r
      v = v/r
       ...
      fe(1) = t1rl * (1.d0 + 3.d0*u + 4.5d0*u2  - usq )
       ...
      do n=0,nspeed
         f(n,i,j) = eomega*fn(n,i,j) + omega*fe(n)
      enddo
    enddo
  enddo
```

Program