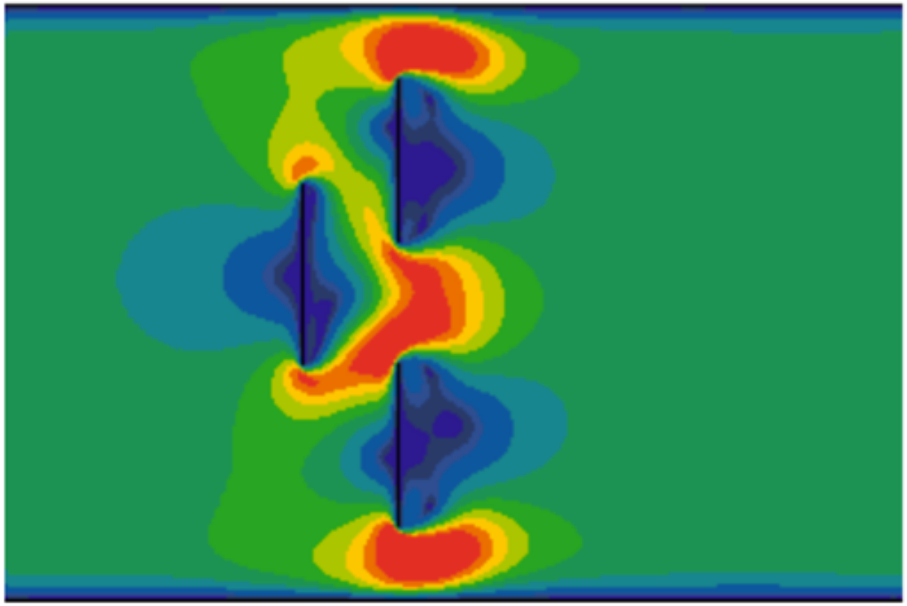
Lattice-Boltzmann:

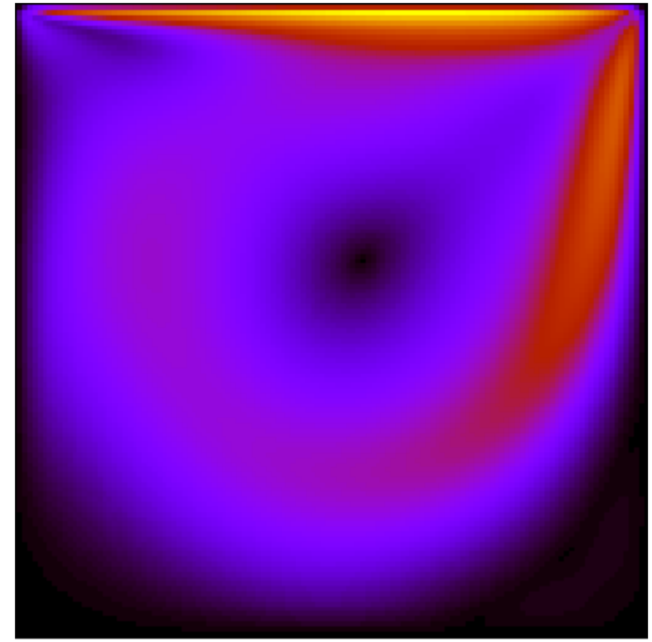
What is it?
Why does it matter?
What are we going to do with it?

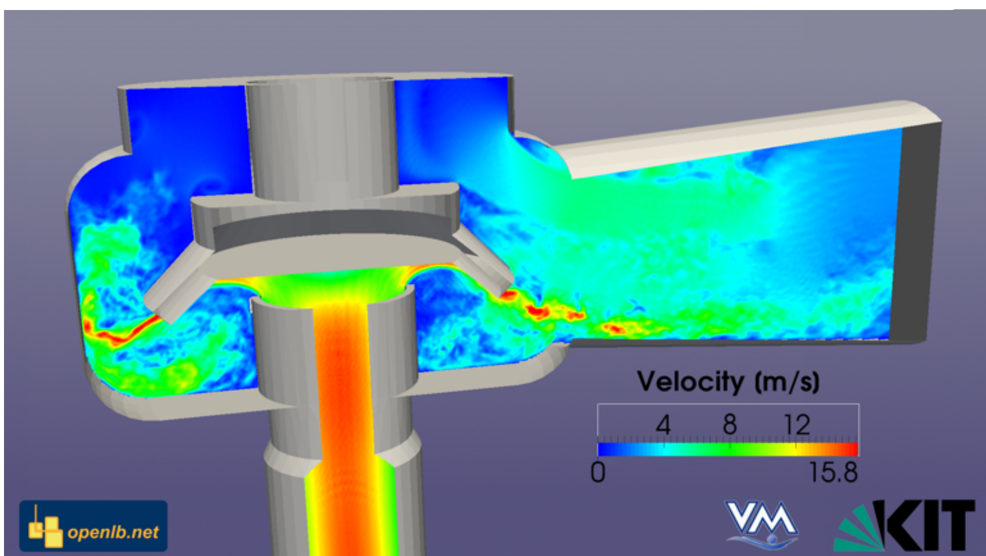
Video of a Lattice-Boltzmann simulation of the human vocal fold.

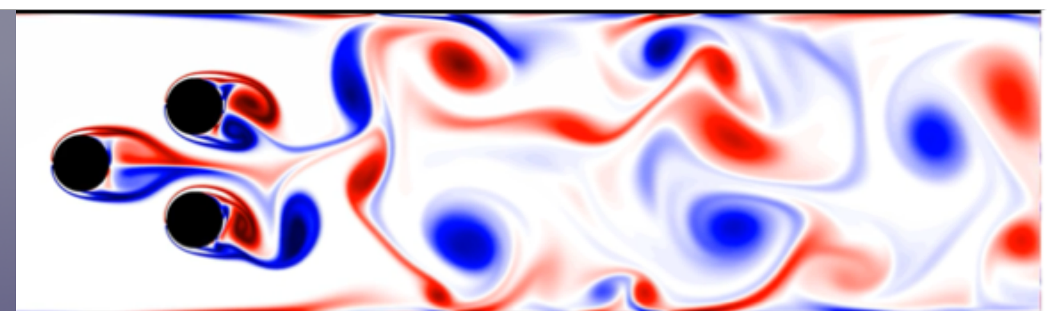
Lattice-Boltzmann

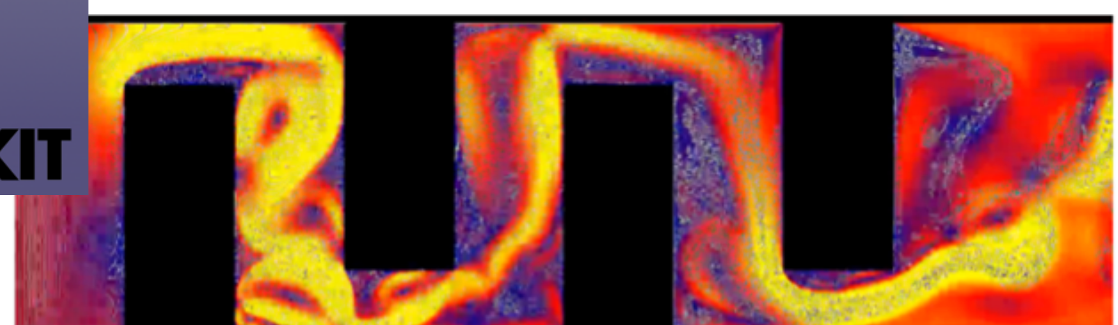
- Used for simulation of fluids according to the Boltzmann equation.
- An alternative to solving the Navier-Stokes equation.
- Practically used for simulating:
  - Aerodynamics, e.g. http://optilb.org/openlb/automotive
  - Plasma flows
  - Flow through porous media (e.g. soil)
  - Simulation of respiration for medical science, e.g. http://optilb.org/openlb/respiration-nose
  - Blood flow through veins and arteries, e.g. Sun, C., & Munn, L. L. (2008). Lattice Boltzmann simulation of blood flow in digitized vessel networks. *Computers & Mathematics with Applications (Oxford, England : 1987)*, *55*(7), 1594–1600. http://doi.org/10.1016/j.camwa.2007.08.019
- Many real codes use LBM; e.g.:
  - OpenLB: http://optilb.org/openlb/
  - Ludwig: http://ludwig.epcc.ed.ac.uk
  - waLBerla: http://walberla.net











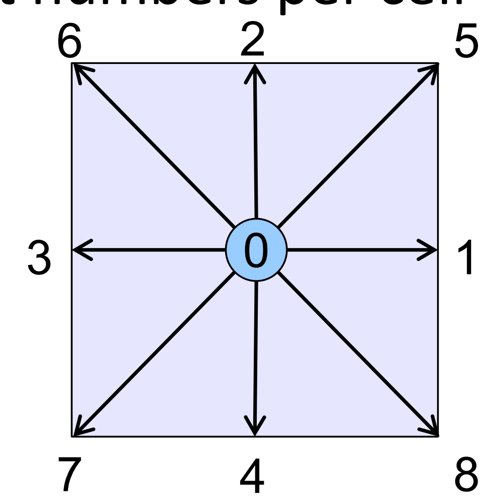
https://youtu.be/pBKZnqhhs28

Lattice-Boltzmann

- The equation describes the change of particle distribution through the sum of changes due to external forces, diffusion and collisions.
- The “maths” models a probability distribution, not individual particles.

Structured grid

- Lattice-Boltzmann is a member of the structured grid dwarf.
- The 2D spatial domain is discretised into square cells.
- The probability distribution is represented as 9 "speeds" (or directions) of fluid flow in each cell.
    - I.e. 9 floating point numbers per cell

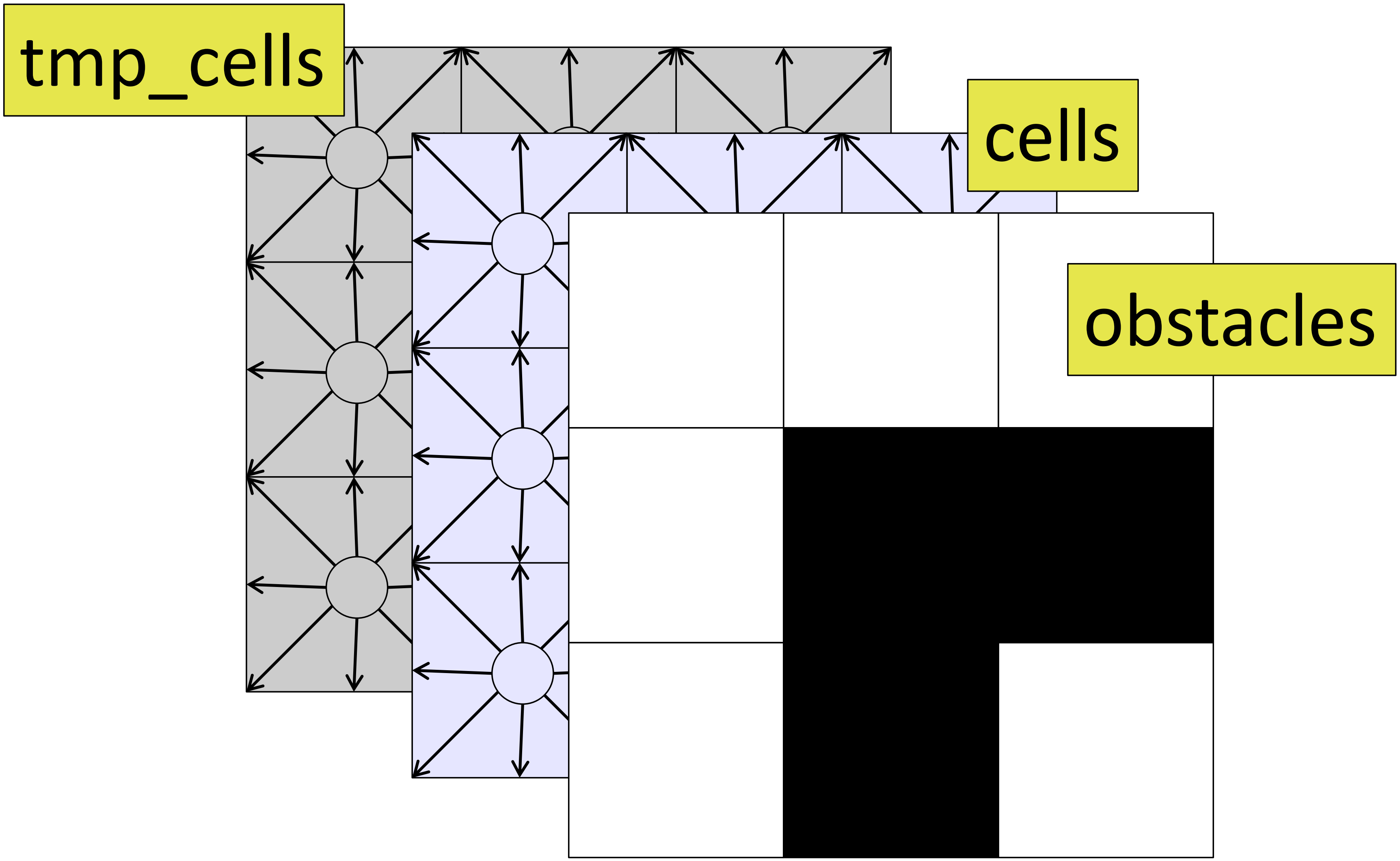


D2Q9: 2 dimensions, 9 speeds

Coursework: implementing D2Q9

- We give you a serial D2Q9 Lattice-Boltzmann code.
- It reads in problem parameters and a file of obstacle positions.
- It then runs the solution for a number of timesteps.
- Finally it writes 2 files, and prints a summary:
  - The final state of the grid
  - The average velocity across the whole grid for each time step

Data structures – as given



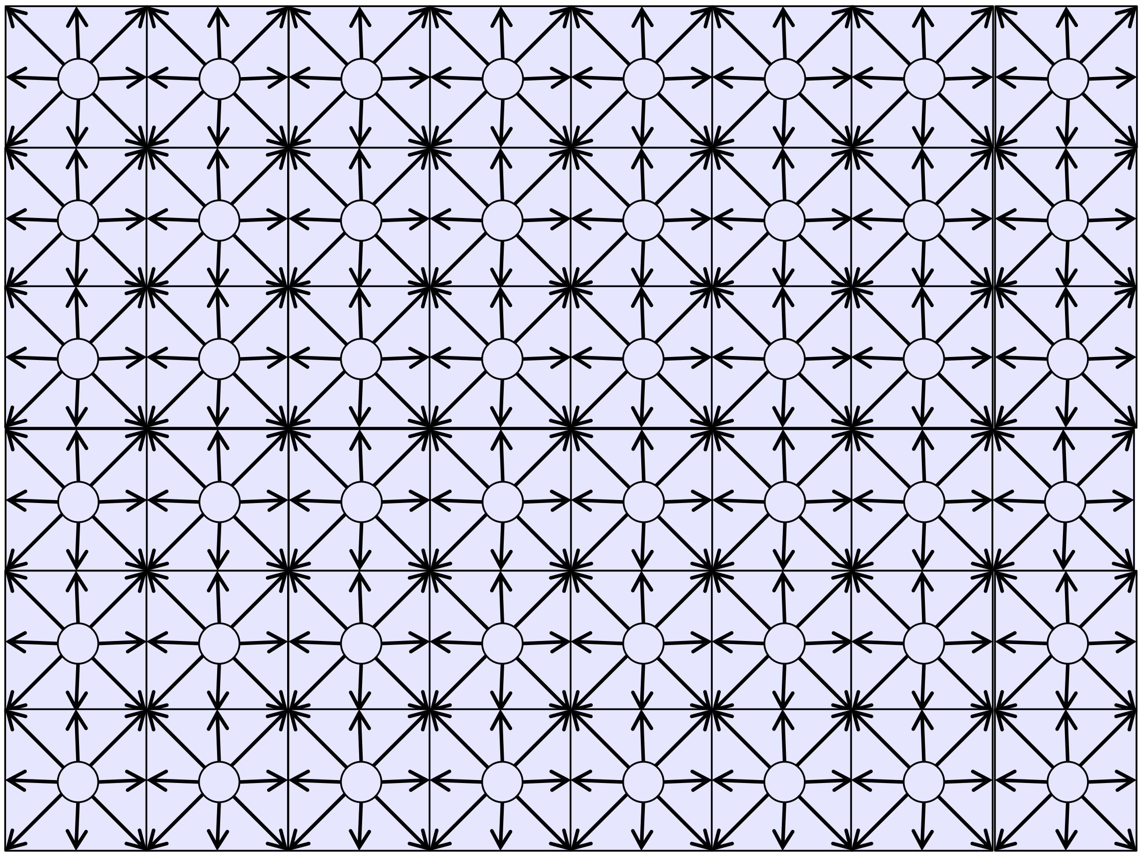
Halo exchange

- All cells can be computed independently
    - Good for parallelism!
- Decompose the grid data across multiple MPI ranks
    - Each MPI rank works on a portion (sub-domain) of the grid

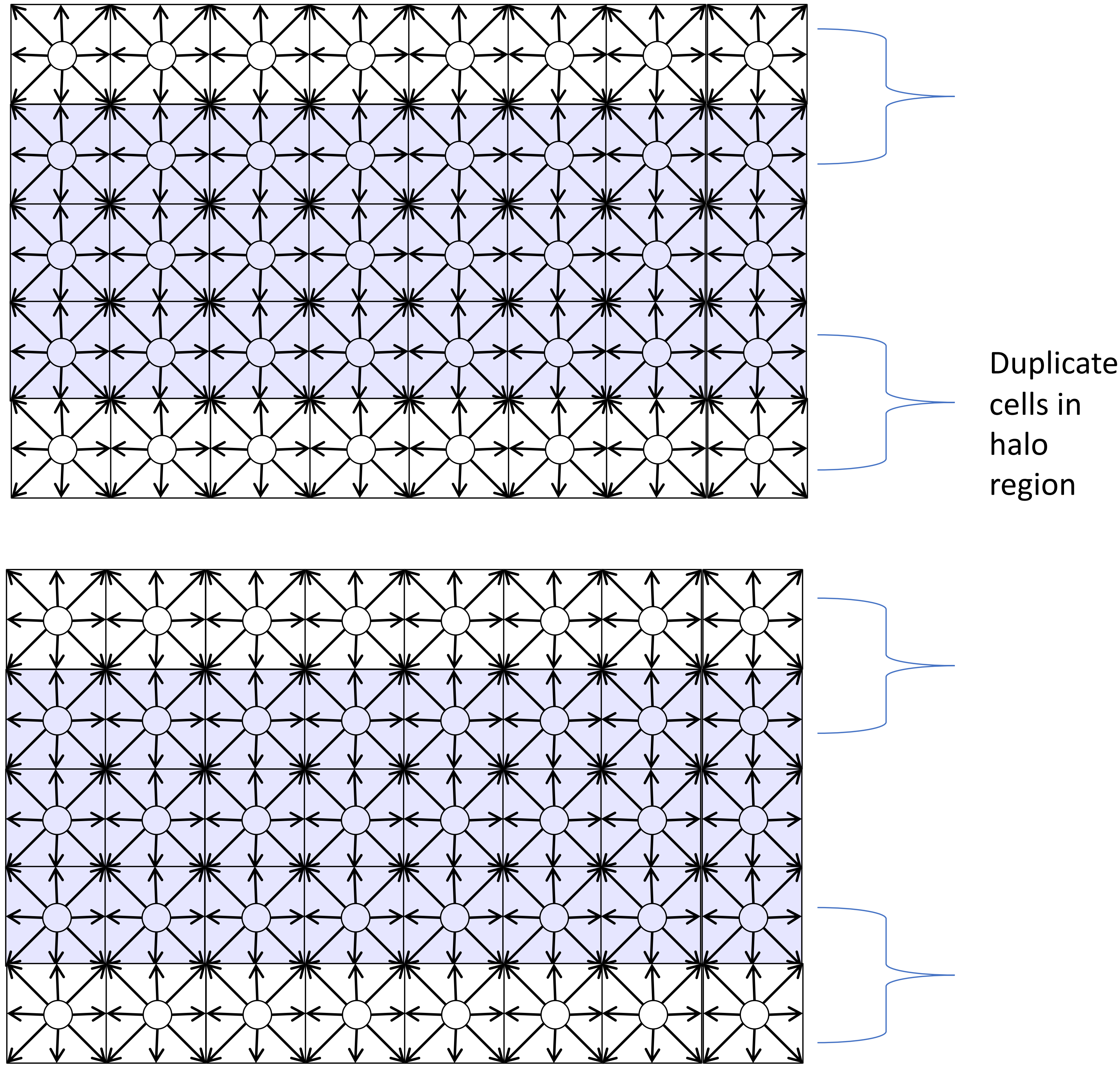
But!

- One routine needs data from neighbouring cells
- Need to implement a halo exchange scheme to communicate the sub-domain boundary data to neighbouring ranks

Example decomposition



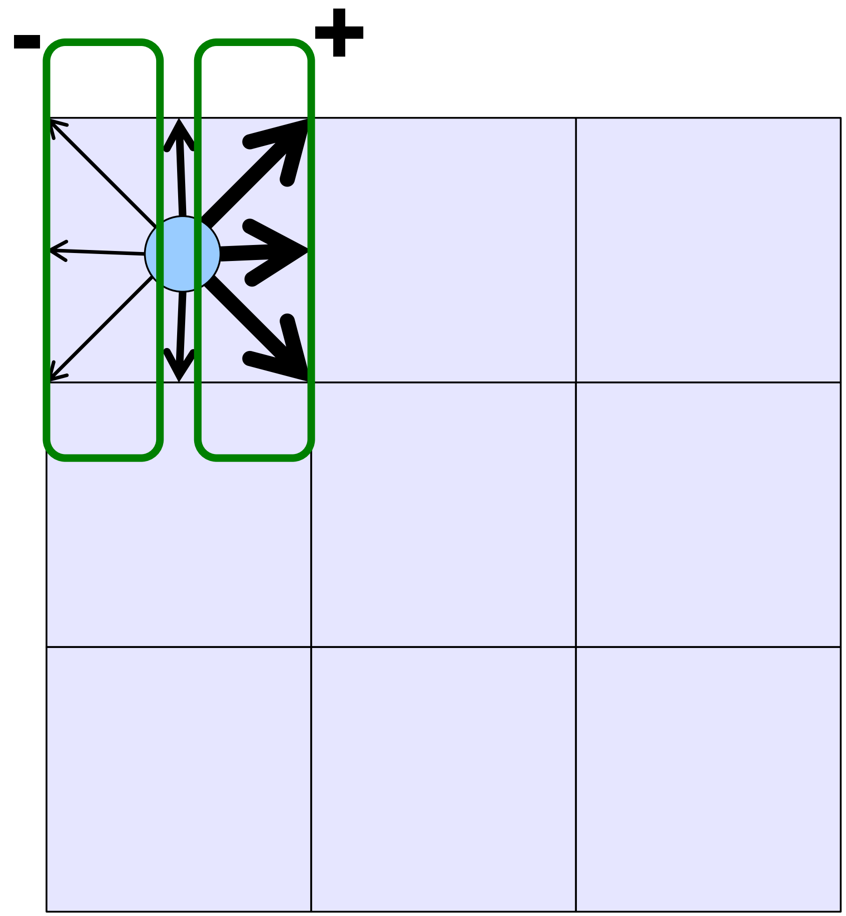
Decompose along rows between two ranks


Duplicate cells in halo region

Basic code structure:

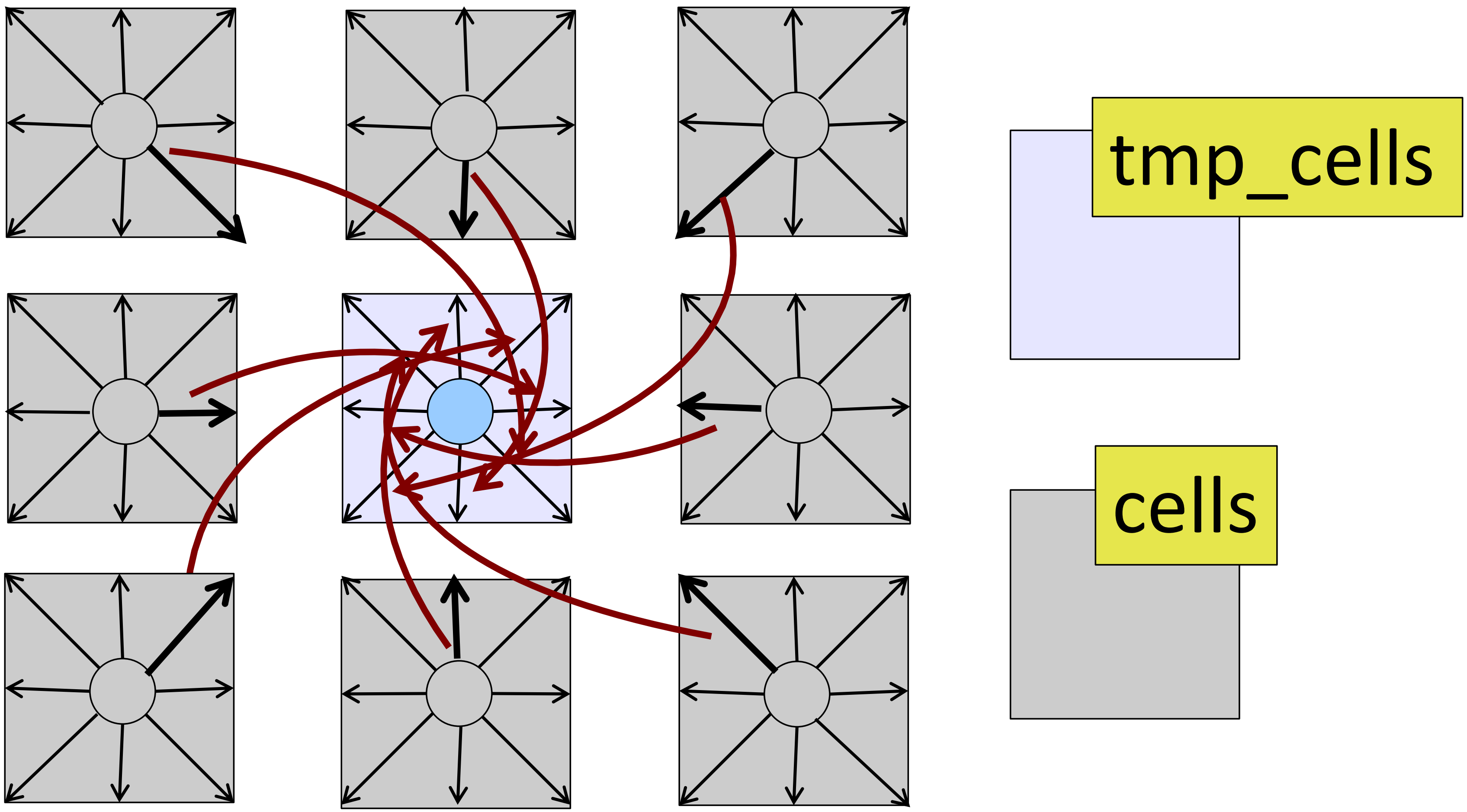
- initialise()
- for each timestep:
    1. **accelerate flow()**
    2. **propagate()**
    3. **rebound()**
    4. **collision()**
    5. **av_velocity()**
- write_values()
- finalise()

Accelerate Flow

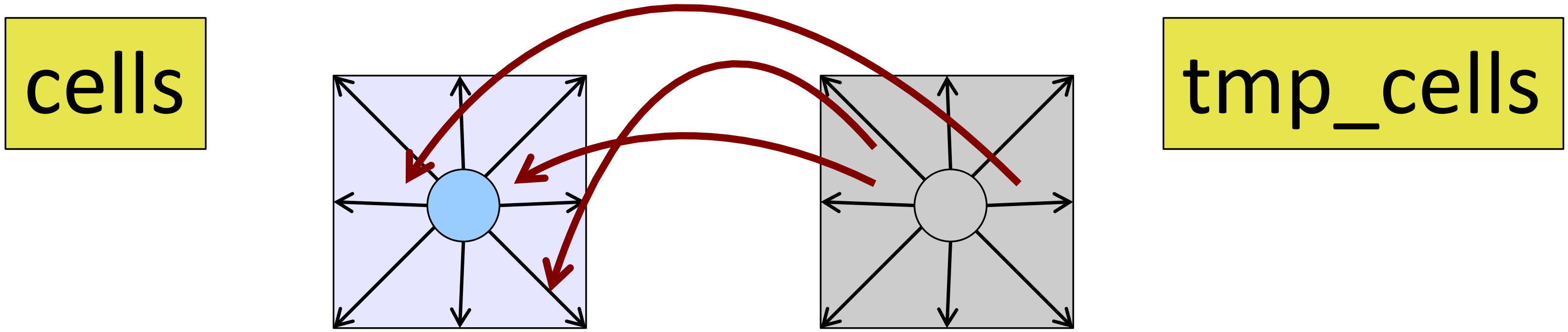


On one row of cells, add to the right pointing speeds, and subtract from the left pointing speeds

Propagation Step

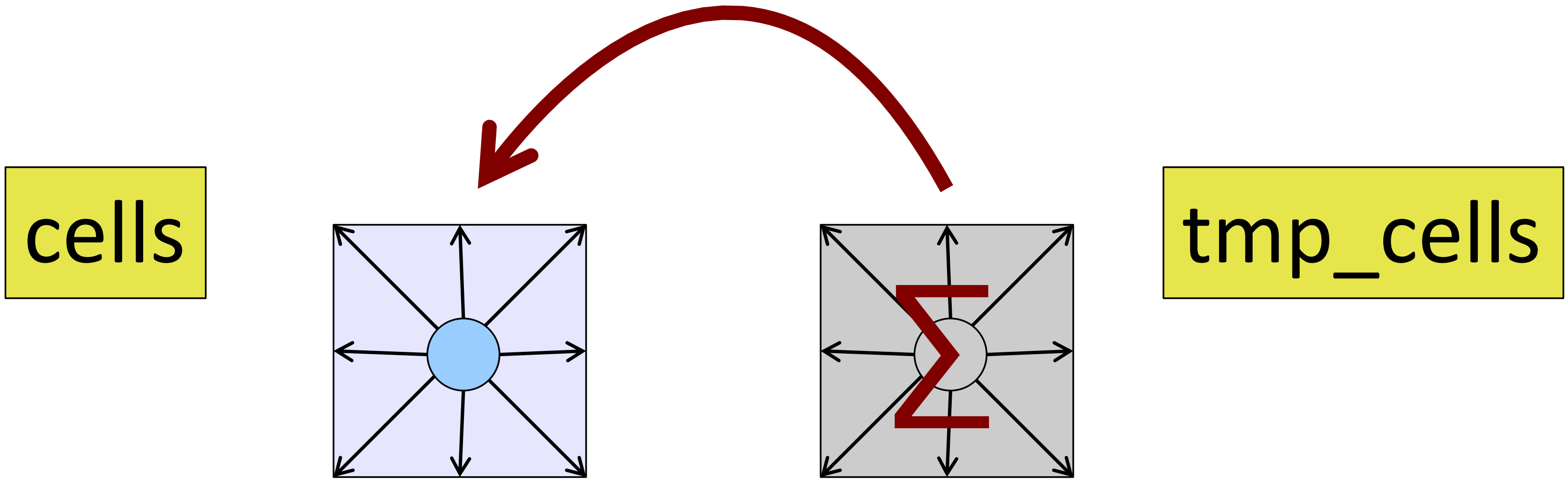


Rebound Step: Obstacle Cells



Mirror the speeds in each cell
E.g. east becomes west, north-west becomes south-east

Collision Step: Non-Obstacle Cells



Perform 'relaxation' arithmetic in each cell

Loops

```
/* iterate for maxIters timesteps */
for (tt=0; tt<params.maxIters; tt++) {
  ...
  e.g. propagate() {
  /* loop over _all_ cells */
  for (jj=0; jj<params.ny; jj++) {
    for (ii=0; ii<params.nx; ii++) {
      ...
```

```
cells[ii + jj*params.nx].speeds[0]
```

Getting the coursework

- Code will be on GitHub
  https://github.com/UoB-HPC/advanced-hpc-lbm

- There are currently FOUR input files.
  - We will be checking that your code works for all of them
    - So you should too!
  - You are free to create extra problem sizes to work on too
    - Might help when you're running on multiple GPUs

The assignments

The Advanced HPC course differs from the intro's ones

- More challenging parallel programming tasks
- Targeting multiple nodes of BCp4 *and* GPUs
- Designed to encourage "exploration based discovery":
  - Port the LBM code to use an "MPI+X" style of parallelism, to run across multiple nodes and GPUs at once
  - You get to choose the "X": OpenCL, OpenMP 4.5, Kokkos…
  - We encourage you to explore interesting technologies and ideas

Assignment 1 – Flat MPI

- Parallelise the LBM code using just MPI (“flat MPI”)
- **Crucial** to get this step working first!!!
- Submit working code by the end of week 17
  - No report required at this stage
  - We will auto-mark your code and check it works though
- Formative

Assignment 2 – MPI+X

- Parallelise the LBM code using MPI and some other language that supports running on the GPUs
  - “MPI+X”
- This should leverage your working MPI code
- Submit working code by the end of week 24
- Include a 3-4 page report describing what you did
- Summative: 100%

Important guidance

- Aim to get the workload right:
  - ~100 hours for the course, of which ~12 hours will be in lectures
  - The rest of the time is all available for the assignments
  - → 88 hours over 11 weeks, or 8 hours per week
  - Don't overdo it!! The right answer is what you can do in the time allocated
- This is a deliberately open-ended assignment
  - You need to practice knowing when you're done
  - We do not need perfection
  - If you're enjoying the assignment and you want to keep going, only do so if this doesn't negatively impact your health, or other coursework
- Don't be fooled by distant deadlines
  - This is a highly challenging assignment, you'll need the time!
  - You **really really** need working flat MPI before week 18…

Further reading

Learn more about lattice Boltzmann at NASA:

- https://youtu.be/I82uCa7SHSQ?list=PLYfV6sBy5qTI0iZhmI8L1rhg0cpYUHGCl

Watch lattice Boltzmann simulations:

- The respiratory system: https://youtu.be/FmPvHIZSjyk?list=PLYfV6sBy5qTI0iZhmI8L1rhg0cpYUHGCl
- The circulatory system: https://youtu.be/o11NDvrZMNs?list=PLYfV6sBy5qTI0iZhmI8L1rhg0cpYUHGCl