# Lattice Boltzmann Method - Excercise

## 1D Heat equation

Dr.-Ing. Kannan Masilamani

DLR German Aerospace Center
Software Methods for Product Virtualization
Simulation Frameworks
Dresden

February 8, 2024


DLR

# Exercise: 1D heat equation

- We want to solve the heat equation on a 1D rod:

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x^2}.$$

- On the right side we have adiabatic boundary condition:

$$\left.\frac{\partial T}{\partial x}\right|_{x=L} = 0.$$

- On the left side we have a fix temperature:

$$T(x = 0) = 1.$$

- As initial condition we set:

$$T(x, t = 0) = 0.$$



# Exercise Basic: 1D heat equation

- $D1Q2$
- $L_x = 50$, number of elements
- $L_z = 2$, number of distribution functions
- $\mathrm{mstep} = 250000$, max number of iterations
- $\mathrm{epsilon} = 10^{-5}$, residual value at convergence
- $T_i = 0.0$, initial condition
- $\mathrm{T\_ToBW} = D$, means Dirichlet BC at west wall
- $T_{\mathrm{west}} = 1.0$, value of Temperature at the west wall
- $\mathrm{T\_ToBE} = A$, means adiabatic BC at east wall
- $\alpha = 0.15$, value of the thermal diffusion coefficient



# Exercise Basic: 1D heat equation

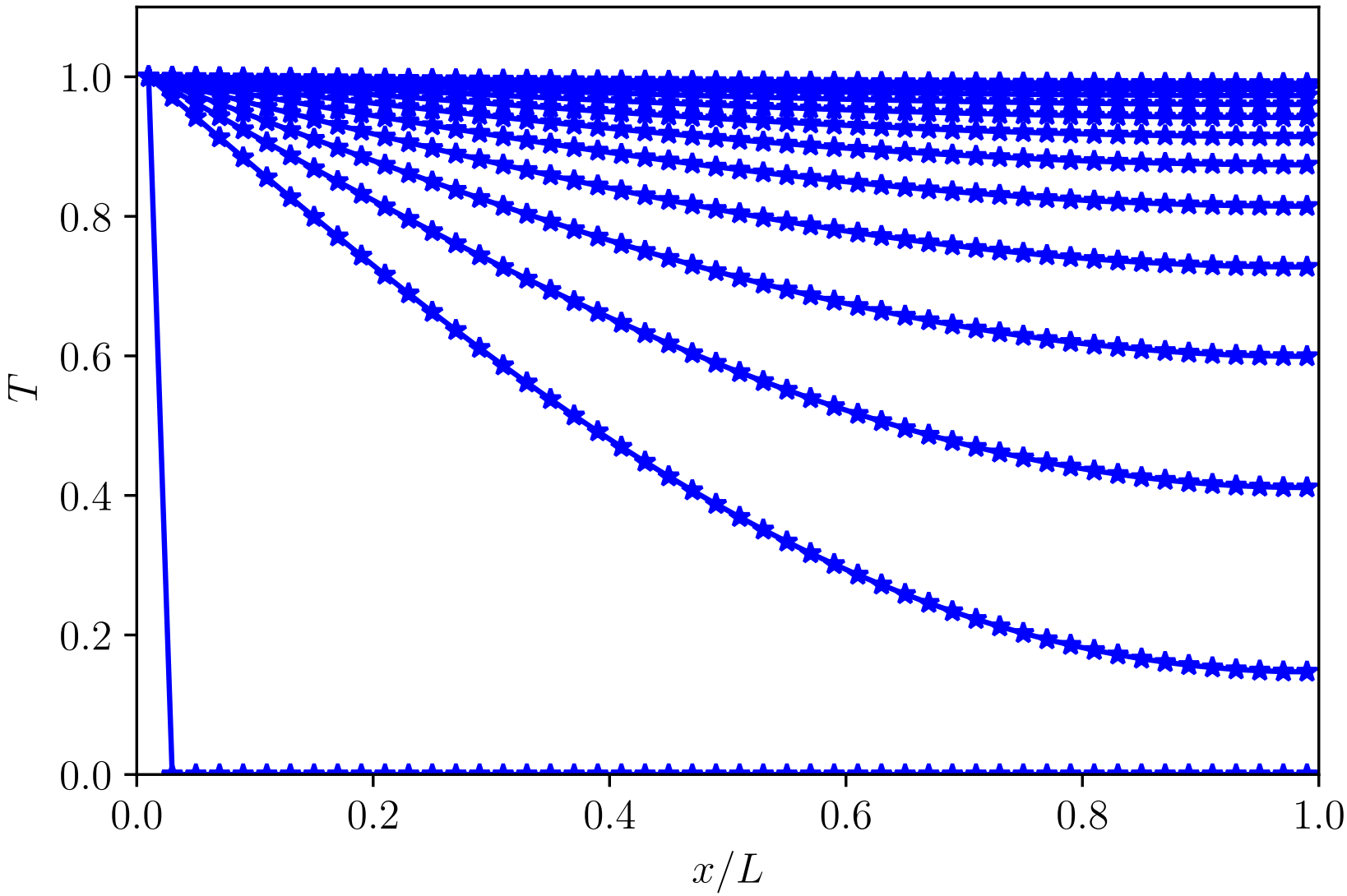
Compile, run and visualize the temperature evolution.

```
$ cp -r $KURS/exercises/LBM_Exercises $MYWS
$ cd $MYWS/LBM_Exercises
$ module load GCC/11.3.0
$ export CXX=g++
$ cd D1Q2_HeatDiffusion
$ mkdir build
$ cd build
$ cmake -DCMAKE_BUILD_TYPE=Release ..
$ make
$ cd ..
$ qsub run_cluster.sh
$ evince TemperatureEvolution.pdf
```



# Exercise Basic: 1D heat equation

Result with 50 elements on the rod



# Exercise Intermediate: 1D heat equation

Apply adiabatic BC to west wall and Dirichlet BC to east wall.

```
$ cd $MYWS/LBM_Exercises
$ cp D1Q2_HeatDiffusion
       D1Q2_HeatDiffusion_Reverse -r
$ cd D1Q2_HeatDiffusion_Reverse
```

Modify main.cpp, then

```
$ rm -r build D1Q2*
$ mkdir build
$ cd build
$ cmake -DCMAKE_BUILD_TYPE=Release ..
$ make
$ cd ..
$ qsub run_cluster.sh
$ evince TemperatureEvolution.pdf
```



# Exercise Intermediate: 1D heat equation

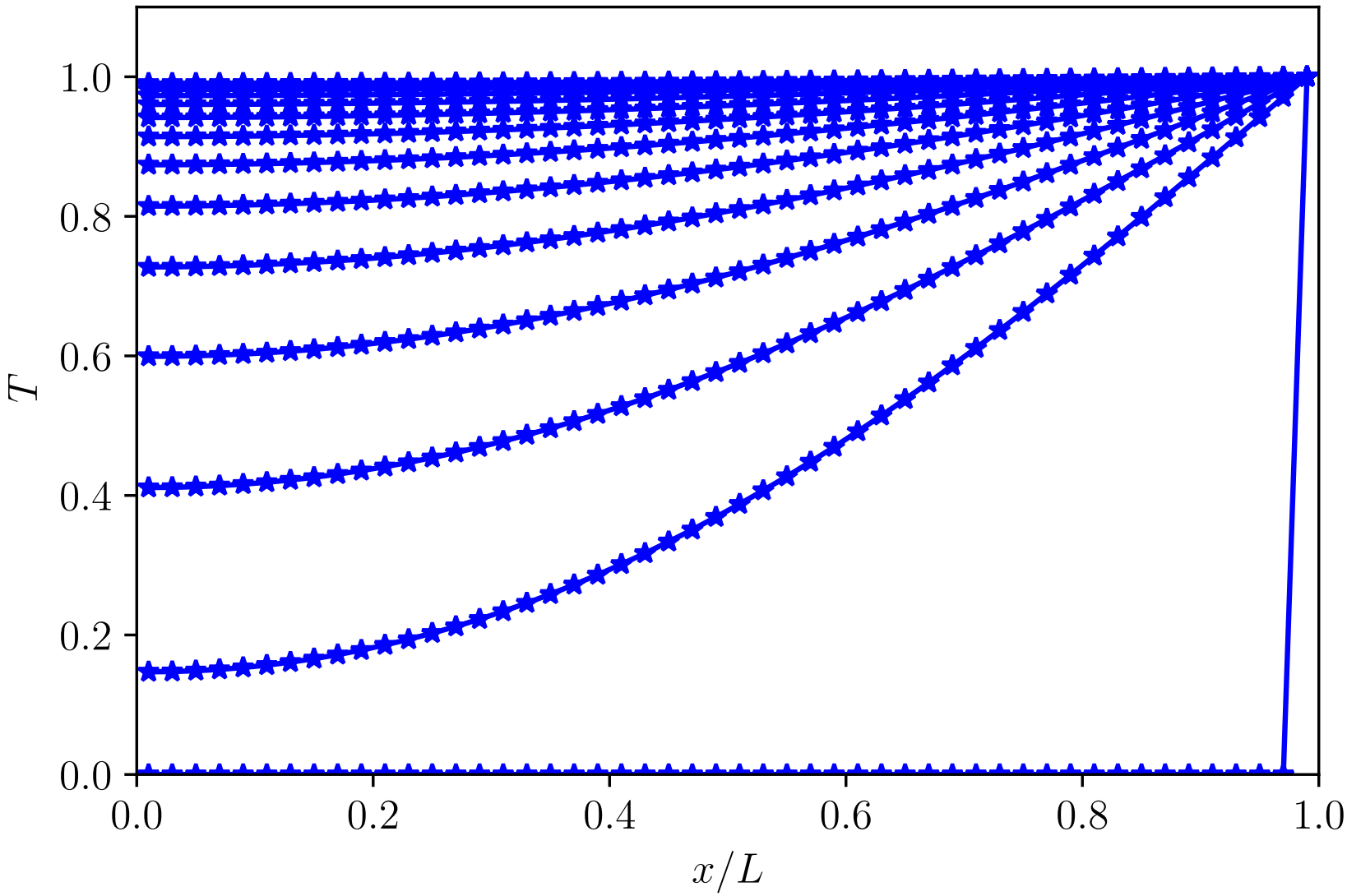
How to recompile the main.cpp file and submit the job.

```
$ cd build
$ make
$ cd ..
$ qsub run_cluster.sh
$ evince TemperatureEvolution.pdf
```



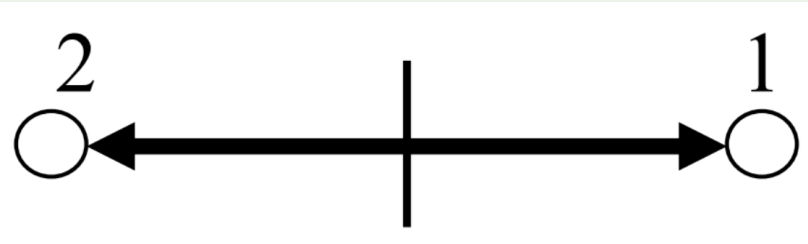
# Exercise Intermediate: 1D heat equation

Result with 50 elements on the rod

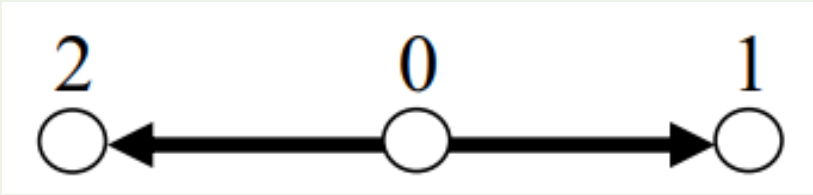



# Exercise Advanced: 1D heat equation

## Velocity space discretization




[1]




[1]

- Use $D1Q3$ instead of $D1Q2$ (see slide 50 of LBM Presentation).

# Exercise Advanced: 1D heat equation

```
$ cd $MYWS/LBM_Exercises
$ cp D1Q2_HeatDiffusion
       D1Q3_HeatDiffusion -r
$ cd D1Q3_HeatDiffusion
```

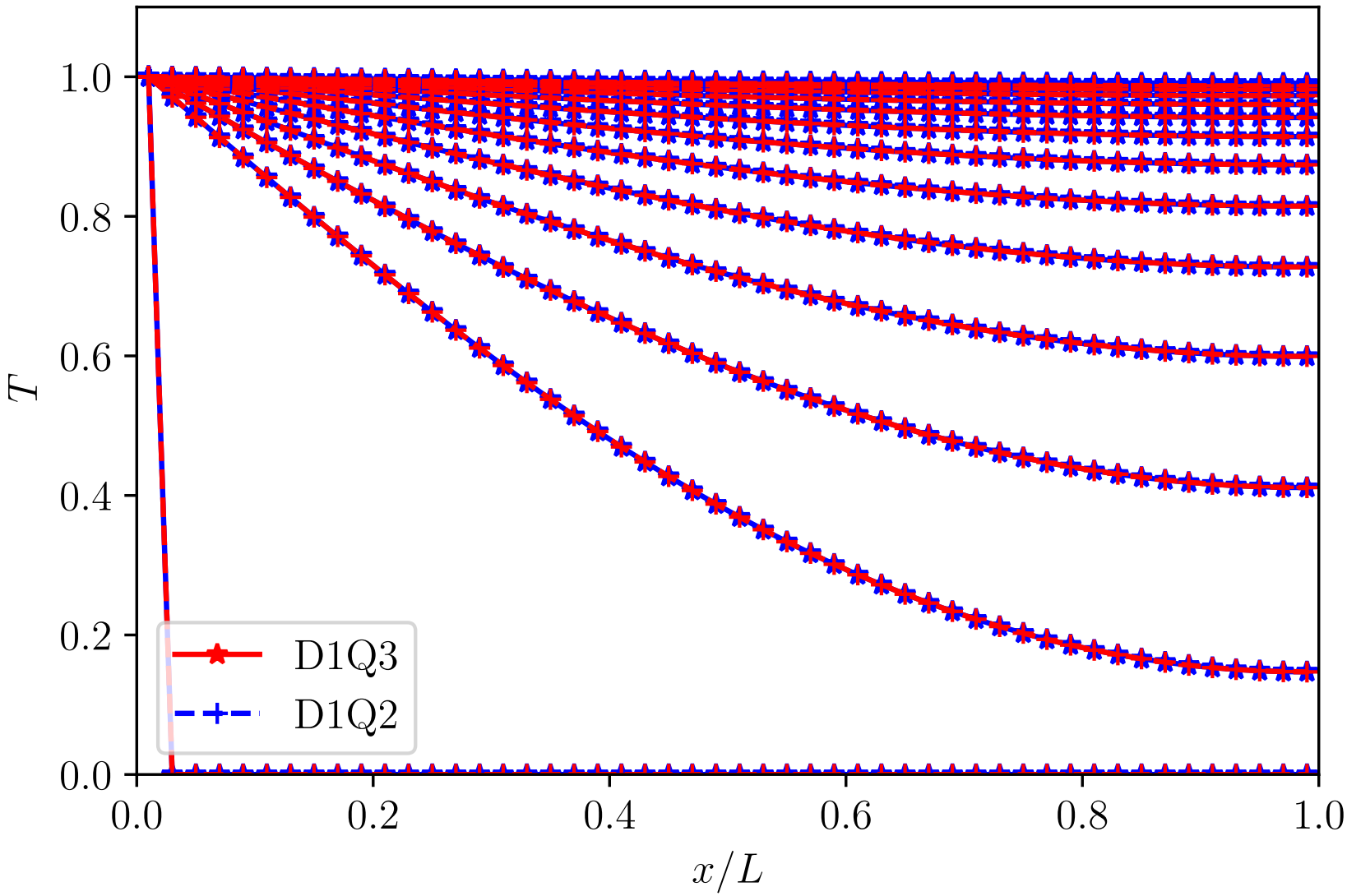
Modify main.cpp, then

```
$ rm -r build D1Q2*
$ mkdir build
$ cd build
$ cmake -DCMAKE_BUILD_TYPE=Release ..
$ make
$ cd ..
$ qsub run_cluster.sh
$ evince TemperatureEvolution.pdf
```



# Exercise Advanced: 1D heat equation

Result with 50 elements on the rod




# Exercise Advanced: 1D heat equation

To generate the comparison graph:

```
$ cd $MYWS/LBM_Exercises
$ qsub run_cluster.sh
$ evince TemperatureComparison.pdf
```

# Exercise Intermediate: 1D heat equation - solution

```
296 //Temperature BC
297 // kind of BCs E=East, W=West
298 // insert:
299 // - A for Adiabatic BC
300 // - D for Dirichlet BC, fixed Tw
301 const char T_ToBW = 'A';
302 const char T_ToBE = 'D';
303
304 // for Dirichlet
305 const mytype T_west = 0.;
306 const mytype T_east = 1.;
```



# Exercise Advanced: 1D heat equation

- HINT 1: Store the value at the center (0 in the picture) in the **f** element number 2. Why?
- HINT 2: We need to change the BCs, both or just one kind?
- HINT 3: We need to adjust the weight factors and the versors for the streaming directions
- HINT 4: did you change the speed of sound?



# Exercise Advanced: 1D heat equation - answers

- HINT 1: Streaming does not need to be modified. Also we avoid index shifting for all coded functions.
- HINT 2: Only Dirichlet needs rewriting
- HINT 3:

$$w_0 = 4/6, w_{1,2} = 1/6, c_2 = 0$$

- HINT 4:

$$c_s = 1/\sqrt{3}$$



# Exercise Advanced: 1D heat equation - solution

```
104  f[1*Lx+Lx-1]=Tw-f[0*Lx+Lx-1]-f[2*Lx+Lx-1];
125  f[0*Lx + 0] = Tw - f[1*Lx + 0] - f[2*Lx + 0];
243  const int Lz = 3; //D1Q3 so Lz = 3
313  const mytype cs = c / sqrt(3);
322  w[0] = 1. / 6.;
323  w[1] = 1. / 6.;
324  w[2] = 4. / 6.;
329  cx[2] = 0.;
```

# URLs for pictures

1. https://link.springer.com/book/10.1007/978-0-85729-455-5