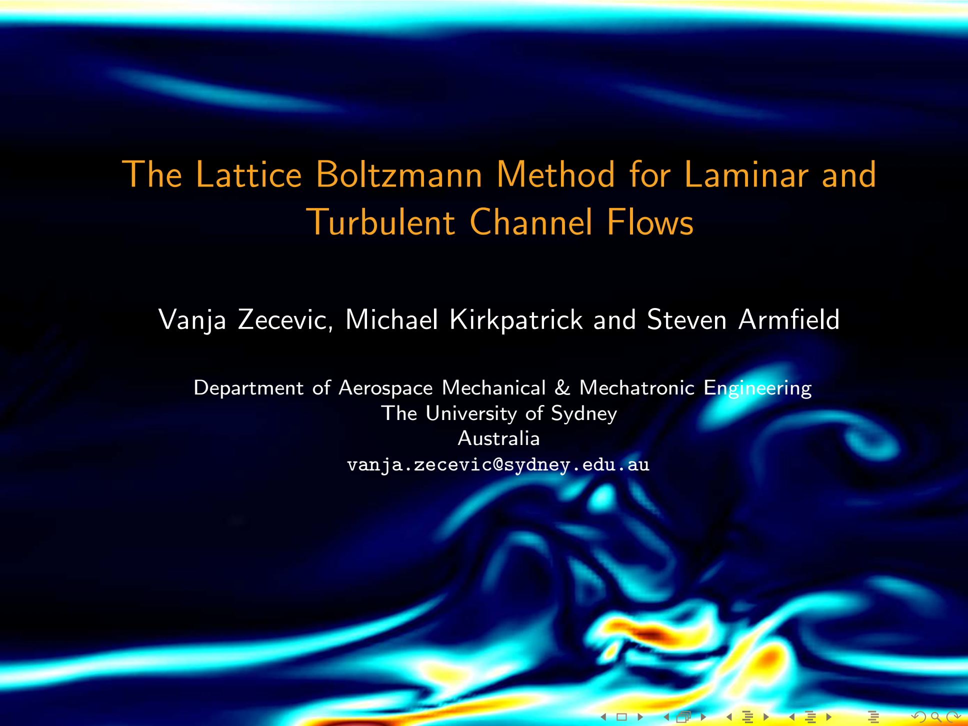


The Lattice Boltzmann Method for Laminar and Turbulent Channel Flows

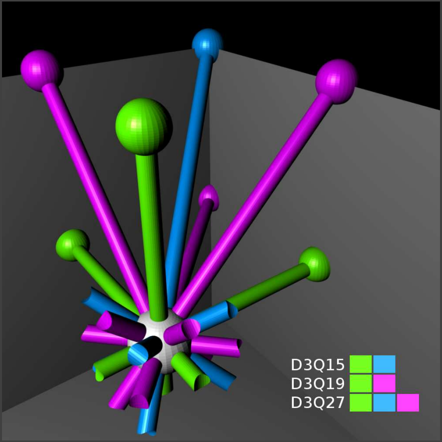
Vanja Zecevic, Michael Kirkpatrick and Steven Armfield

Department of Aerospace Mechanical & Mechatronic Engineering
The University of Sydney
Australia
`vanja.zecevic@sydney.edu.au`

Introduction

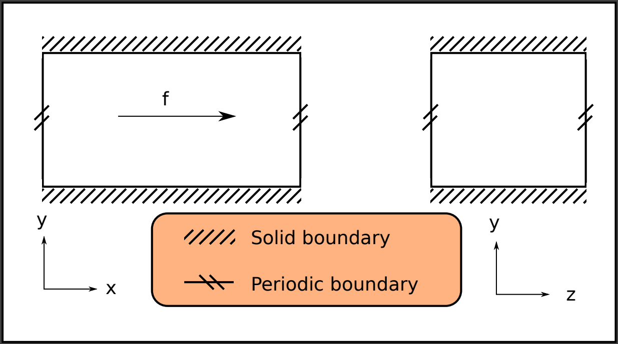
- We aim to develop a high performance, highly scalable parallel code for the simulation of fundamental turbulent flows.
- In this work, we use the lattice Boltzmann method to simulate laminar and turbulent channel flows.
- This method allows an efficient implmentation on highly parallel architectures, in this case graphics processing units (GPUs). It is also suitable for more coarse parallelization across multiple processors or GPUs.
- One drawback of the method is that non uniform grids are difficult to implement. We compare a relatively high resolution uniform grid using our code to results published by Moser et. al. [3] using a spectral method with the grid refined at the walls.
- Our results aproach Mosers results as the grid is refined.

LBM - Summary



- Tracks particle populations $f_i(x, y)$
- Time advancement: collision step + streaming step.
- $\sum f_i = \rho$ and $\sum f_i c_{ia} = \rho u_a$ conserved.
- Artificially compressible.
- Review by Benzi, Succi & Vergassola [1].
- Bhatnagar Gross Krook collision operator.

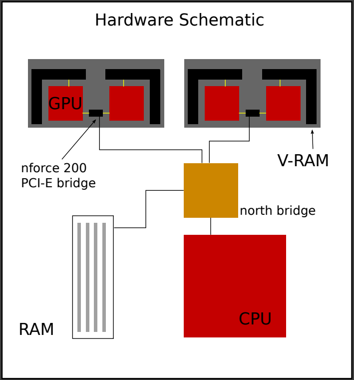
Domain



- Simplest BC's are 'bounce-back'.
- We use the Guo BC [2].

Hardware & Technologies

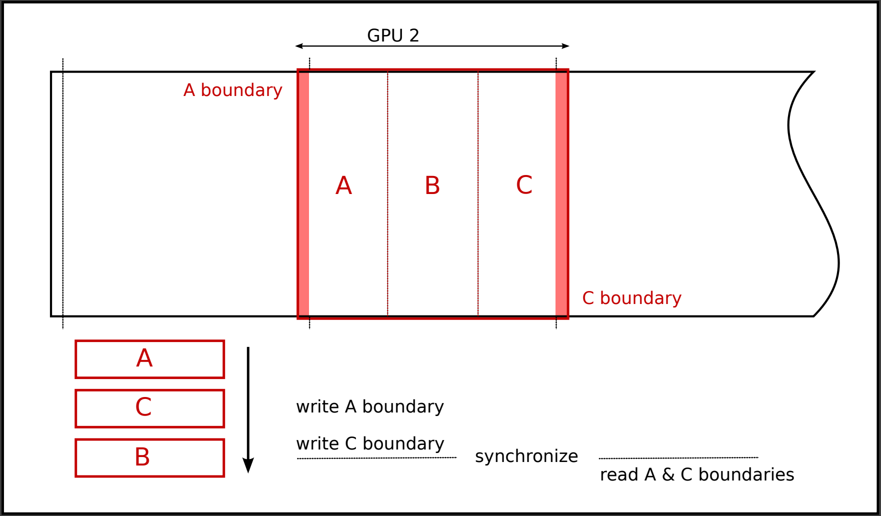
Hardware	
CPU	intel E8600
GPU	2 × GTX295
RAM	4 GB 1066 MHz DDR3
PSU	1500 W
Mobo	GA-X48
Chipset	intel X48
PCI-e	2 × 16 PCI-e 2.0
Technologies	
OS	Debian Lenny AMD64
Compiler	gcc, nvcc
libs	lcudart, pthreads



Parallel Implementation

- GPU uses fine grained data-parallel approach.
- Many execution units (eg. 240) and even more threads (millions).
- Synchronization over the entire domain cripples performance. [4]
- All communications in LBM are nearest neighbor and data dependence requires two synchronizations per time step.
- Conjugate Gradient solvers require multiple synchronizations and whole domain communications per solver iteration .
- Allows us to get 92% parallel efficiency when scaling from one to four GPU's.

Domain Decomposition



Laminar Channel Flow - Parameters 1

Define the following dimensionless quantities,

$$l_{char} = \delta \qquad l_d = \frac{l_{lu}}{l_{char}} \tag{1}$$

$$u_{char} = u_{max} \qquad u_d = \frac{u_{lu}}{u_{char}} \tag{2}$$

$$t_{char} = \frac{l_{char}}{u_{char}} \qquad t_d = \frac{t_{lu}}{t_{char}} \tag{3}$$

The following are constants.

$$\Delta t_{lu} = 1 \qquad \Delta x_{lu} = 1 \tag{4}$$

$$c_s = \text{speed of sound} = \frac{1}{\sqrt{3}} \tag{5}$$

Laminar Channel Flow - Parameters 2

The following equations can be used as constraints,

$$\Delta t_d = \frac{1\,\mathrm{lt}}{t_{char}} = \frac{u_{char}}{l_{char}} \cdot (1\,\mathrm{lt}) = \frac{u_{max}}{\delta} \cdot (1\,\mathrm{lt}) \tag{6}$$

$$\Delta x_d = \frac{1\,\mathrm{lu}}{l_{char}} = \frac{1}{\delta} \cdot (1\,\mathrm{lu}) \tag{7}$$

$$Ma = \frac{u_{max}}{c_s} = \frac{\Delta t_d}{c_s \Delta x_d} \tag{8}$$

$$Re = \frac{u_{max}\delta}{\nu} \tag{9}$$

For a laminar channel flow we can estimate:

$$u_{max} = \frac{f\delta^2}{2\nu\rho} \tag{10}$$

Laminar Channel Flow - Parameters 3

- 8 free variables are Δt_d, Δx_d, δ, Ma, ν, f, u_{max} and Re
- 5 equations linking these variables.
- Usually we would specify Δt_d, Δx_d and Re, however it is more helpful to specify Ma.

The Mach number plays a similar role to the finite element CFL number.

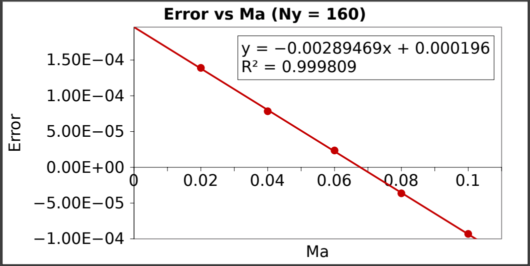
$$CFL = \frac{\Delta t_{lu} u_{max}}{\Delta x_{lu}} = u_{max} \tag{11}$$

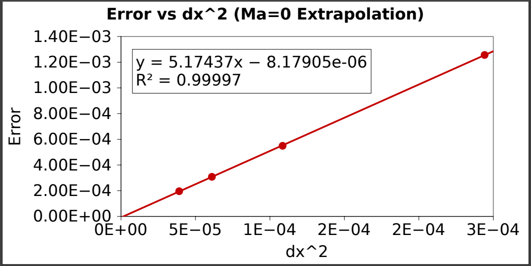
$$Ma = \frac{u_{max}}{c_s} = \frac{CFL}{c_s} = \sqrt{3} \times CFL \tag{12}$$

We use $Ma = 0.1$ for our turbulent simulation which is somewhat equivalent to,

$$CFL = \frac{1}{\sqrt{3}} \times Ma = 0.06 \tag{13}$$

Laminar Channel Flow - Results





- Error compared to the Poiseuille profile.
- 2nd order in space and 1st order in Mach number.

Turbulent Channel Flow - Parameters 1

Compared to spectral method of Moser et. al. [3]

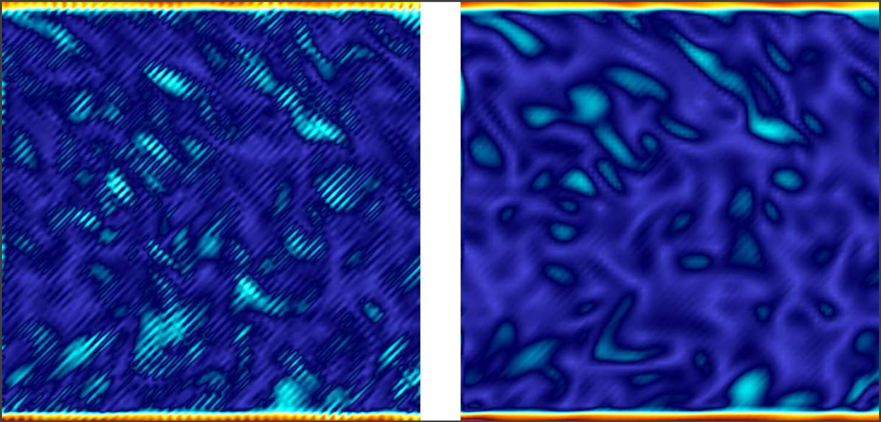
$$f\delta = \tau_w = \frac{\partial u}{\partial z}\mu \Rightarrow u_\tau = \sqrt{\frac{\tau_w}{\rho}} = \sqrt{\frac{f\delta}{\rho}} \tag{14}$$

$$Re_\tau = \frac{u_\tau \delta}{\nu} = \frac{\delta}{\nu}\sqrt{\frac{f\delta}{\rho}} = 180 \tag{15}$$

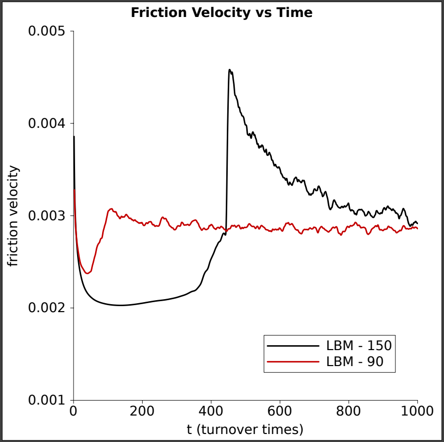
	N_y	l_y	N_x	l_x	N_z	l_z	N_{tot}
LBM	92	2	288	2π	176	π	4.7×10^6
LBM	112	2	352	2π	176	π	6.9×10^6
LBM	132	2	416	2π	212	π	11.6×10^6
LBM	152	2	480	2π	246	π	18.0×10^6
Moser	129	2	128	4π	128	$\frac{4}{3}\pi$	2.1×10^6

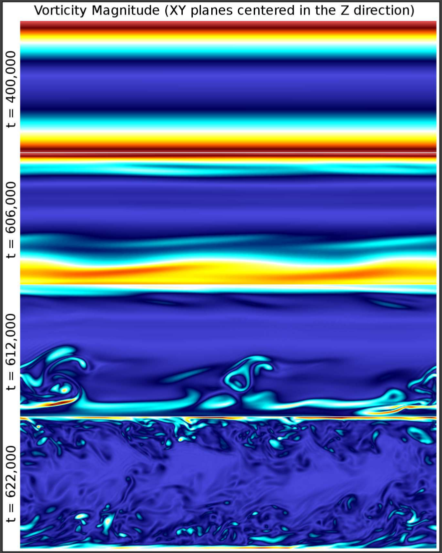
Turbulent Channel Flow - Parameters 2

- Initial condition = curl noise.
- D3Q15 had instability and aliasing.

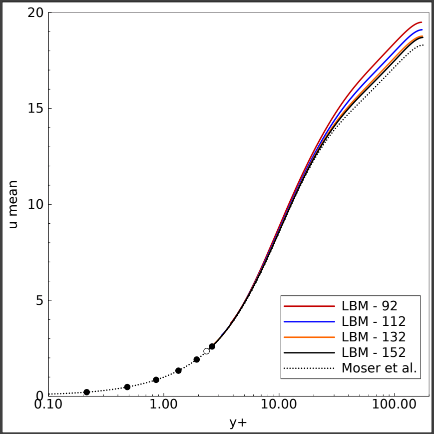


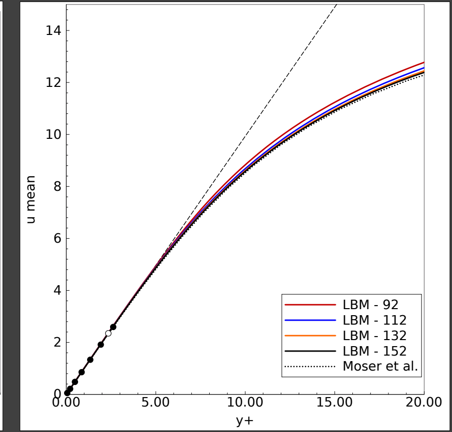
Turbulent Channel Flow - Onset of Turbulence





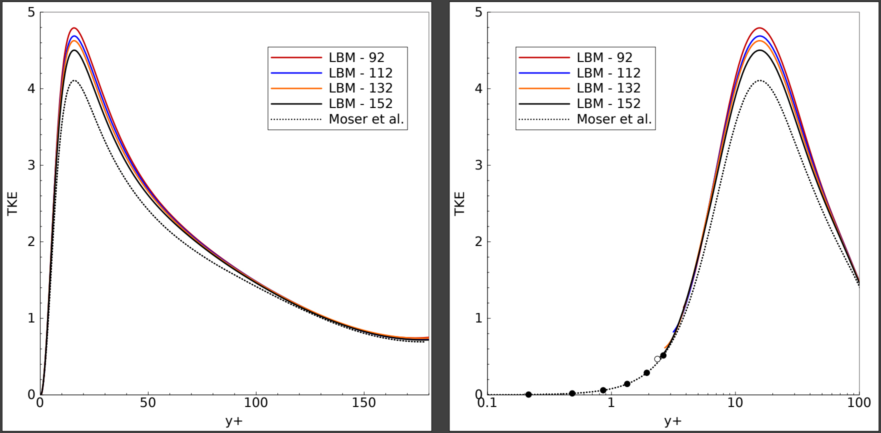
Turbulent Channel Flow - Mean Profile





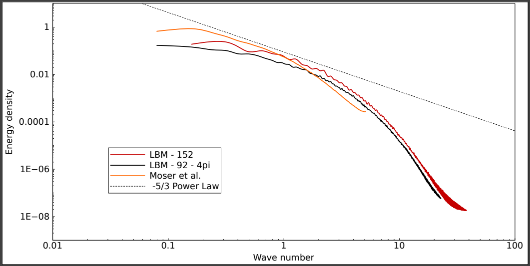
Log law region and viscous sublayer both apparent.

Turbulent Channel Flow - TKE



Height of peaks is a good indicator of agreement.

Turbulent Channel Flow - Energy Spectrum



- Dotted line shows power law expected in inertial range.
- Nyquist frequency in Mosers results has a wave number 32.
- 2π stream-wise dimension appears to be too small with significant energy in the largest wavelength.

Performance

- LBM appears compute bound.
- High performance.
- Good scaling.

	Performance	Theoretical
Updates per s	100×10^6	
GFLOPS (DP)	56	276
Gbps	64	4096

Conclusion

- Convergence was verified using laminar flow.
- The LBM was successful in simulating a fully turbulent flow.
- Found interesting turbulent channel flow results comparing a fine square grid with a locally refined stretched grid.
- High performance and good parallel scaling.
- Research will continue on to thermal turbulent flows using a Boussinesq approximation.

References

[1] R. Benzi, S. Succi, and M. Vergassola.
The lattice boltzmann-equation - theory and applications.
Phys. Rep., 222(3):145–197, Dec 1992.

[2] Z. L. Guo, C. G. Zheng, and B. C. Shi.
An extrapolation method for boundary conditions in lattice boltzmann method.
Phys. Fluids, 14(6):2007–2010, Jun 2002.

[3] R. D. Moser, J. Kim, and N. N. Mansour.
Direct numerical simulation of turbulent channel flow up to $\mathrm{Re}_\tau=590$.
Phys. Fluids, 11(4):943–945, Apr 1999.

[4] NVIDIA.
NVIDIA CUDA C Programming Best Practices Guide, 2009.