Improving the performance of a Lattice Boltzmann fluid solver using TensorFlow

Vikram Singh

The Perse School, Cambridge
Project conducted at University of Cambridge Research Computing Services

August 23, 2019

The Lattice Boltzmann Method

- Macroscopic variables
  $\rho = \sum f_{\text{in}}$
  $E(i, \rho, u) = \rho t_i (1 + u + \frac{u^2}{2} - \frac{3|u|^2}{2})$
- Collision
  $f_{\text{out}} = f_{\text{in}} - \omega(f_{\text{in}} - E)$
- Streaming
- Boundary Conditions

Tensorflow

- Used primarily in machine learning
- Efficient large array operations
- Parallelisable

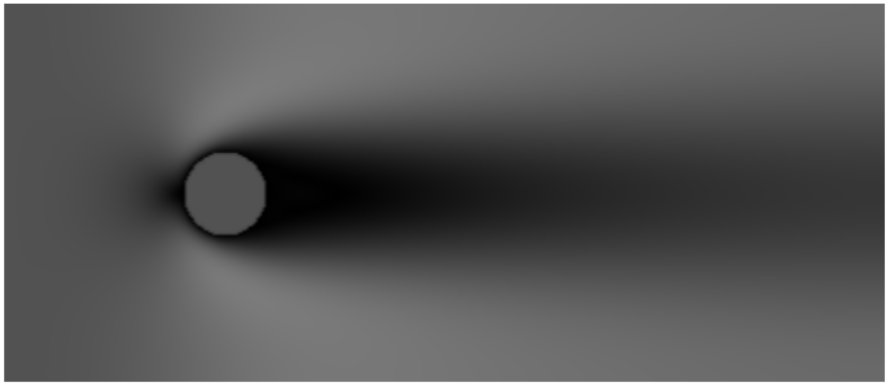
Implementation and Testing

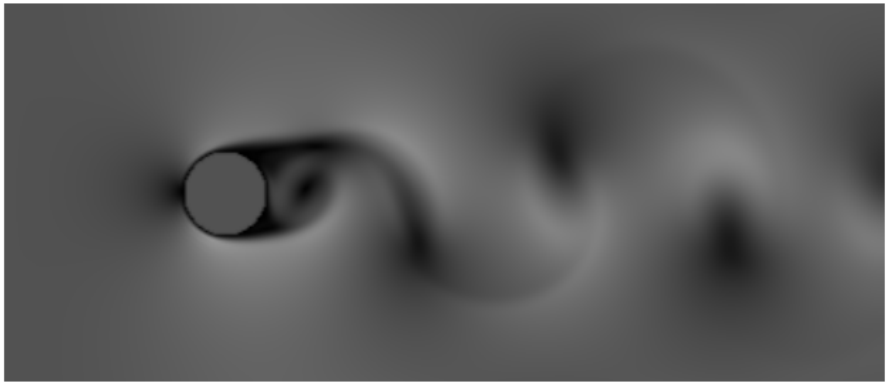
- Implemented two D2Q9 Lattice Boltzmann solvers:
  - Using only Numpy
  - Using TensorFlow for all lattice calculations

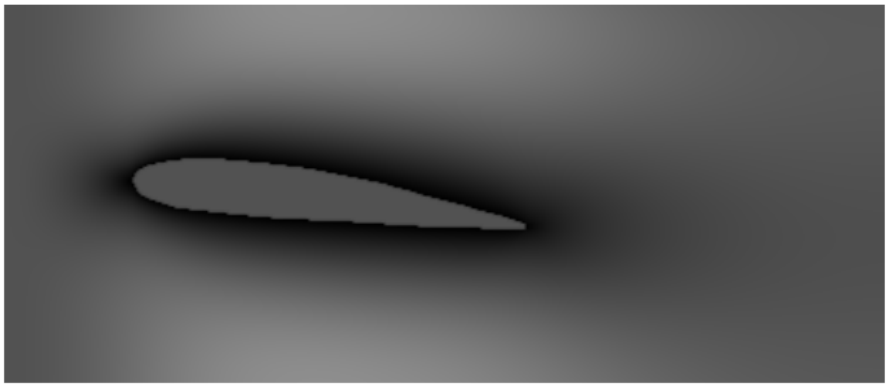
- Performed the following simulations with both solvers:
  1. Flow around a cylinder with low viscosity (420×180 lattice points)
  2. Flow around a cylinder with high viscosity (420×180 lattice points)
  3. Flow around an airfoil (420×180 lattice points)
  4. Flow through a narrowing pipe (420×180 lattice points)
  5. Flow through a bending pipe (1000×452 lattice points)
  6. Flow in a lid driven cavity (180×180 lattice points)

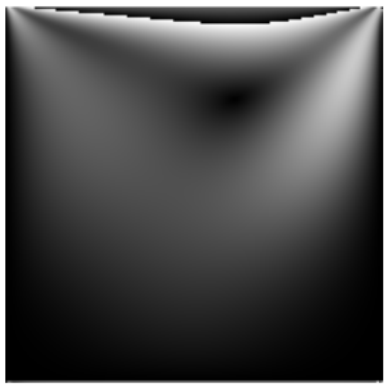
- `github.com/vikram8128/PythonLatticeBoltzmann`

Simulations



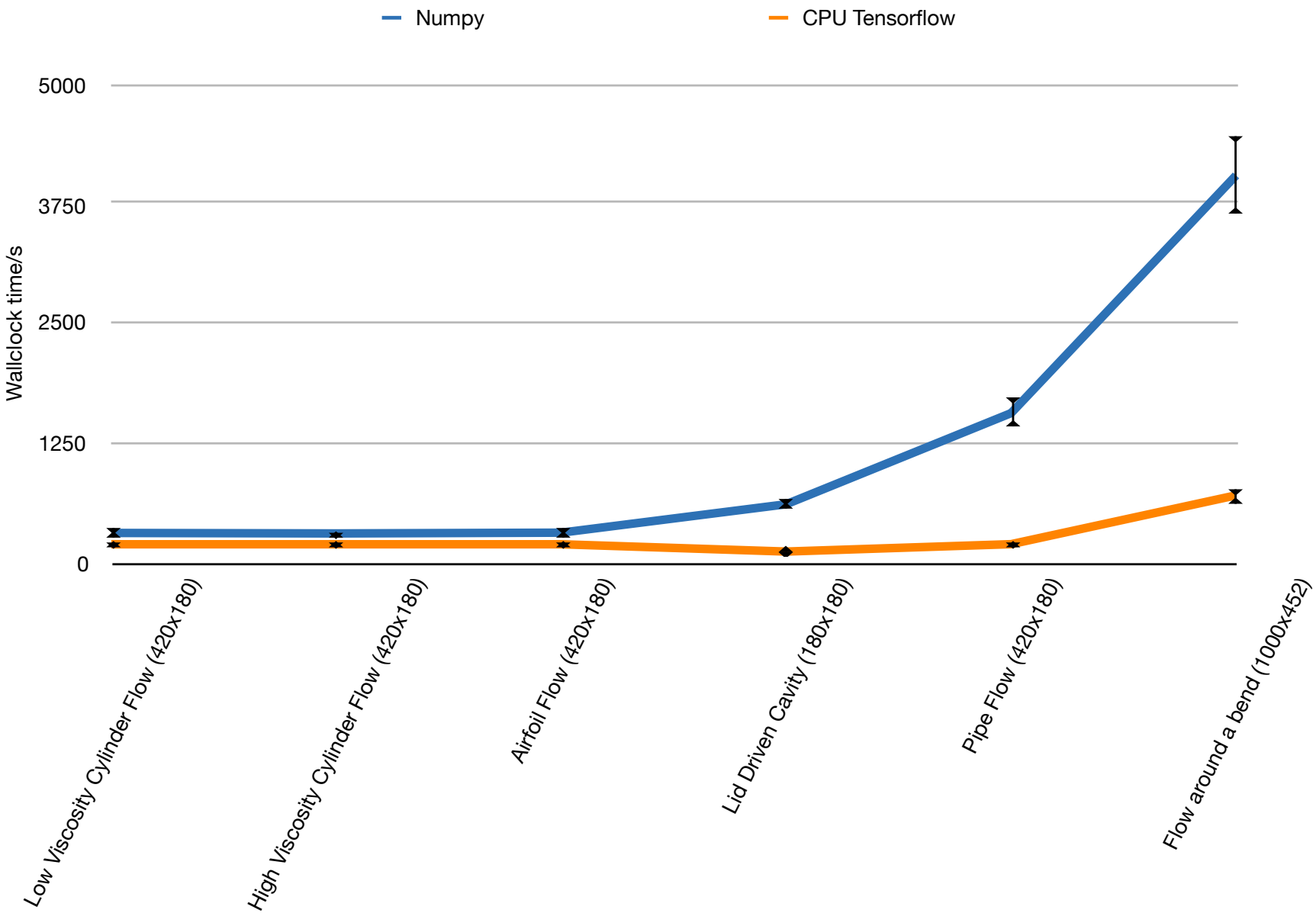






Animations

Results

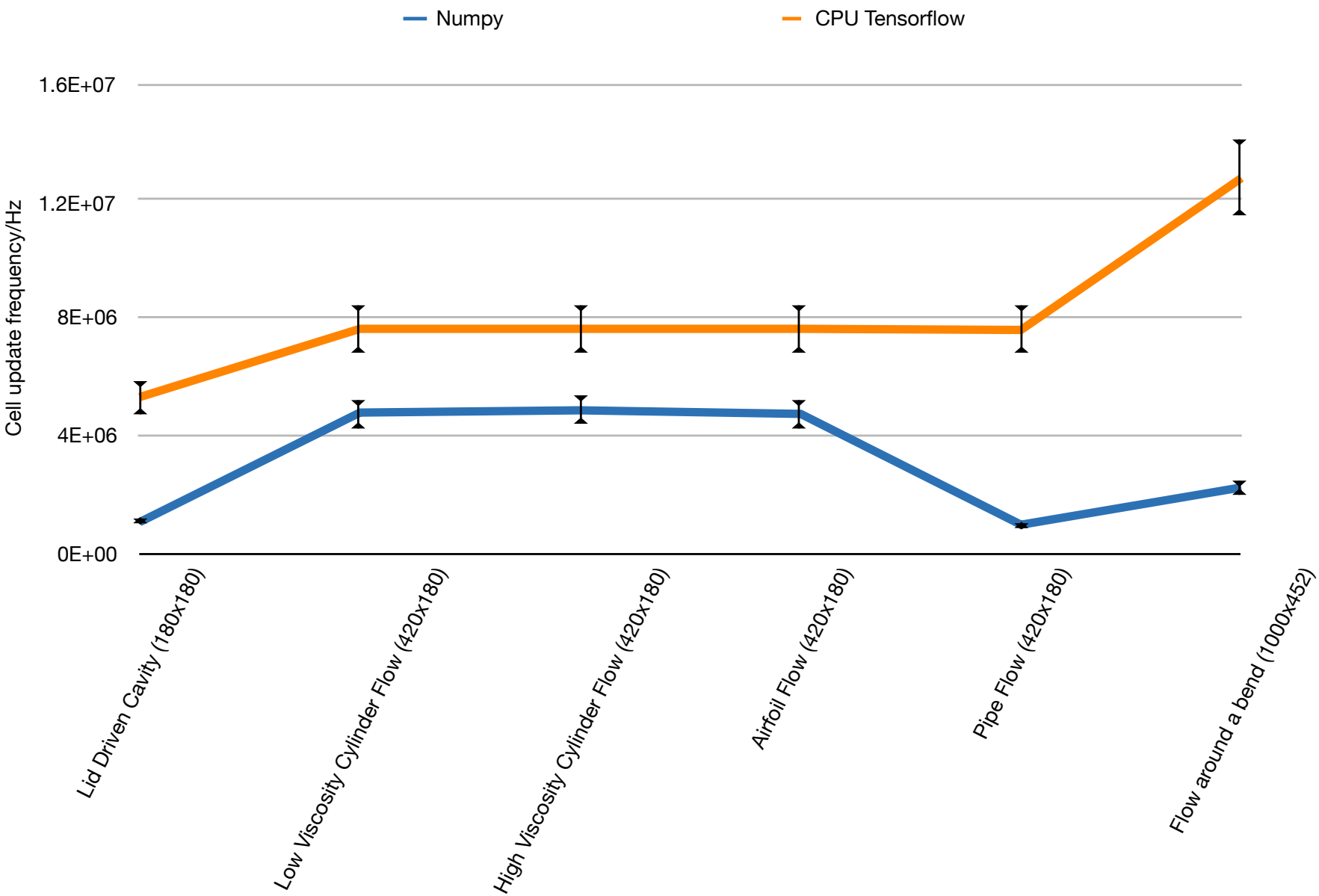

Numpy
CPU Tensorflow
5000
3750
2500
1250
0
Wallclock time/s
Low Viscosity Cylinder Flow (420x180)
High Viscosity Cylinder Flow (420x180)
Airfoil Flow (420x180)
Lid Driven Cavity (180x180)
Pipe Flow (420x180)
Flow around a bend (1000x452)

Results

Table: Wallclock times in seconds taken by each simulation

Simulation	1	2	3	4	5	6
Numpy	316	311	319	1565	4043	615
TensorFlow	198	198	198	199	708	122

Results


Numpy
CPU Tensorflow
1.6E+07
1.2E+07
8E+06
4E+06
0E+00
Cell update frequency/Hz
Lid Driven Cavity (180x180)
Low Viscosity Cylinder Flow (420x180)
High Viscosity Cylinder Flow (420x180)
Airfoil Flow (420x180)
Pipe Flow (420x180)
Flow around a bend (1000x452)

Results

Table: Cell updates per second (MHz) for each of the simulations

Simulation	1	2	3	4	5	6
Numpy	4.78	4.86	4.74	0.966	2.24	1.05
TensorFlow	7.64	7.64	7.64	7.60	12.8	5.31

Conclusions

- Tensorflow outperforms Numpy in all simulations
- Improvement scales well with size and obstacle complexity

Further Work

- Expanding to 3D
- Using GPU Tensorflow

Thank you

- Thank you very much to Jeffrey for suggesting this project and supervising my work.
- Thank you for listening.

References

Chen, Shiyi, Doolen, Gary D. (1998). *LATTICE BOLTZMANN METHOD FOR FLUID FLOWS* Annual Review of Fluid Mechanics 30 (1): 329364 doi:10.1146/annurev.fluid.30.1.329

Alexander J. Wagner (2008) *A Practical Introduction to the Lattice Boltzmann Method*, North Dakota State University

TensorFlow Documentation (`www.tensorflow.org/api_docs/python/`)