Technische Universität München
Lehrstuhl für Aerodynamik und Strömungsmechanik
Lehrstuhl für Informatik V
N. Hosseini, M.Sc.
N. Fleischmann, M.Sc.
J. Kaiser, M.Sc.
F. Menhorn, M.Sc.
B. Rüth, M.Sc.

WS 2018/19
Worksheet 1
Oct 22 2018

Turbulent Flow Simulation on HPC-Systems

Worksheet 1: Getting Started
Deadline: Nov 11 2018 (11:59pm)

1 Prerequisites

Download the source code of the Navier-Stokes code NS-EOF and install it on your machine. The code has been tested extensively under Linux (Ubuntu). It is suggested that you also work with a Linux system, since the HPC systems are based on similar systems as well. You need a valid installation of a MPI implementation. It is suggested to use OpenMPI under Ubuntu. You further need to install the PETSc library, version 3.5, including MPI support and a library which implements MPI. A detailed installation guide can be downloaded from the Moodle pages of this lecture.

2 Getting Familiar with NS-EOF

In order to get started with NS-EOF, you can find a tutorial on the Moodle pages which comprises the essentials of the (final) version of the code which you are going to further develop. Study the tutorial together with the source code of NS-EOF. The following exercise is meant to make you familiar with the principles of NS-EOF (data structures, stencils, iterator concept etc.). The work packages for all worksheets are marked with *TODO* statements in the source code. However, own extensions are possible at several stages.

3 VTK Output

In order to visualize the simulation results, the software ParaView[1] is used. Exemplary files filled with data for cavity scenarios (2D/3D) are provided in moodle. Your task is to write Paraview-compatible files in vtk-format.

1. Implement the class `VTKStencil`. The respective header file is given in the folder `stencils` of NS-EOF. Implement functionality to write the grid structure, that is the vertices of the Cartesian grid. Since the stencil classes only provide functionality to loop over

[1] www.paraview.org

cells, but not over the vertices of the grid, you can just write your own nested for-loop formulation for this purpose (use a lexicographic ordering of the vertices). Use the functionality of the pointer `meshsize` of the `Parameters` object to determine the position of all vertices. You may use the variables `localSize` and `firstCorner` of the `ParallelParameters` to determine the coordinate range of the grid.
In order to write cellwise data (i.e. pressure and velocity values), implement the methods `apply(...)` of the `VTKStencil`. If the current cell under consideration is a fluid cell, the flow field (pressure and velocity) of the simulation is written to a file; otherwise, zero values should be output. The flow field information should correspond to the evaluation of the fluid quantities in the midpoint of the grid cell. Since your fluid velocities are stored on a staggered grid, interpolation of the fluid velocity is required in the cell midpoints. You can use the method `getPressureAndVelocity(...)` of the `FlowField` for this purpose.
The filename of each vtk-file should consist of a file stem which is specified in the xml-config file, see `VTKParameters` in `Parameters.h` (you do not have to implement the parsing from the config, this is already available), the number of the current time step and the suffix ”vtk”.

2. Incorporate the call to `VTKStencil` into the class `Simulation` using the method `plotVTK()`. The output of a vtk-file should be triggered after subsequent equidistant time intervals; the length of this time interval should be taken from `VTKParameters` (variable `interval`).

3. Test your method for sequential flow simulation, using various flow scenarios such as the lid-driven cavity, channel flow or flow across a backward-facing step. Make sure that your implementation works for 2D/3D and on both uniform and stretched meshes.

4 Flow Physics and Profiling

1. Study the scenarios `cavity`, `channel` and `channel` with backward-facing step in more detail. Therefore, investigate
   - the influence of the Reynolds number on the velocity and pressure field. In particular,
     - how many vortices can you observe in the cavity scenario, depending on the choice of the Reynolds number? Where are these vortices located?
     - how does the velocity/pressure field of the channel flow depend on the Reynolds number?
     - how is the velocity/pressure field behind the backward-facing step affected by the choice of the Reynolds number?
   - the influence of the length and height of the backward-facing step on the flow field.
   - the influence of the used mesh on the overall accuracy and time step size. You may restrict your considerations to the channel flow scenario for this sub-task.

2. Use gprof[2] to investigate the sequential performance of your simulation. Install gprof on your linux system. In order to use gprof, add the flag “-pg” when compiling your program NS-EOF. After execution of a simulation, a file `gmon.out` should be created in

[2] For details on GProf, see https://www.cs.utah.edu/dept/old/texinfo/as/gprof.html

your main directory. Call “`gprof ./ns &> analysis.txt`” to evaluate the performance statistics and write the corresponding information to a file `analysis.txt`. What do you observe? Which routines take most of the time in your simulation? How does the VTK output affect your performance profile? How could you improve the performance of your code?