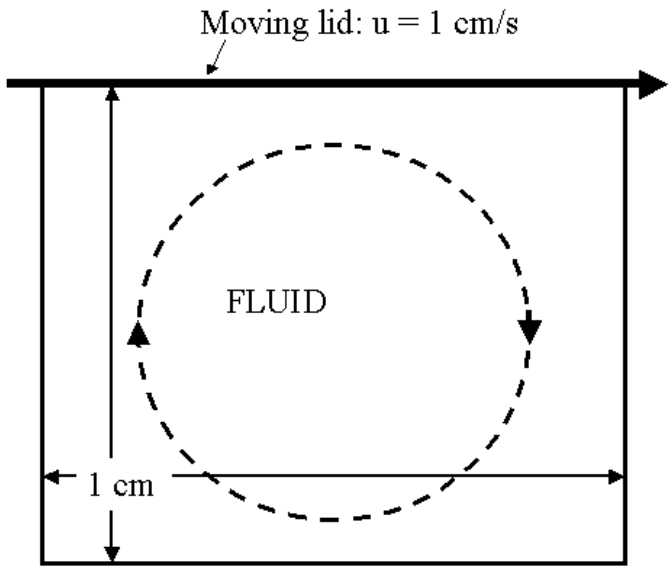
ME 811: Computational Methods for Incompressible Flow

Spring Quarter, 2011
The Ohio State University
Homework 4
Due Friday May 6, 2011 by 5 pm

The objective is to solve the incompressible Navier-Stokes equation on a staggered mesh using the finite-volume method and the SIMPLE algorithm. This HW is worth 300 points.

The test problem to be solved is the so-called driven cavity problem. This is a case where a square cavity is filled with a fluid (water, in this case), and the top lid is moved tangentially in a continuous mode, as if it were a conveyor belt. Due to viscous drag, the fluid next to the lid starts moving, eventually creating a steady strongly recirculating flow within the cavity (see figure below). Note that this particular problem does not have any inlets or outlets. This is a test problem that is commonly used even today to benchmark new and better numerical algorithms for solution of the Navier-Stokes equation.



Here are the steps needed to solve the problem:

(1) Develop discrete equations using finite-volume integration of the governing equations on a staggered grid, as discussed in class. Pay close attention to details, especially at boundary nodes. In the end you must neatly write down equations for all nodes in the algebraic form (five-band form) that we have been discussing in class. Use central differencing for diffusion and first order upwind differencing for advection.

(2) Develop a computer program that will sequentially solve these algebraic equations as needed for the SIMPLE algorithm. Use an iterative solver of your choice for the individual (inner) equations, remembering that the simplest method (point-by-point Gauss-Siedel) may not always converge. Thus, it is strongly recommended that you use the ADI method or something better.

Use $\rho = 1000$ kg/m^3, and $\mu = 10^{-3}$ kg/m/s for your calculations. For simplicity, when deriving your discrete equations and writing your code, you may assume these to be constant. Neglect gravity effects.

For parametric studies, vary your Reynolds number ($\rho u_0 L / \mu$, where L is the cavity size, and u_0 is the lid velocity) from the baseline value of 100 by varying the speed of the lid. Perform computations for Reynolds numbers of 500 and 1000, in addition to the baseline case. In each case, perform computations with 40 and 80 cells in each direction. When you are debugging your code, you may want to use just 5 cells in each direction.

For each Reynolds number, plot *u* and *v* velocity components along *x* = 0.5 cm and *y* = 0.5 cm lines (*i.e.*, at the cavity centerlines). Also, make a contour plot of the pressure field in the whole cavity. To best visualize the flow field (you will get extra credit for this), you may want to make vector plots in the whole cavity. Note that to do so on a staggered mesh, you have to interpolate the final *u* and *v* values to the same spatial node before plotting. For each case, also plot the residuals of all three equations (log scale) versus number of outer loop iterations. Comment on your results both from physical and numerical perspective. Other comments of a general nature are also welcome.

General Advice

- Work at a consistent pace. Do not wait until the second week to start the homework. By the end of the first week, you should have already derived all your equations, and written the first draft of your code. The second and third weeks should be spent in debugging the code and getting it to work.
- Do not take short-cuts. If you do, you will make mistakes. Draw as many figures (stencil diagrams as possible). Write down all equations neatly on paper before programming. The most important aspect in writing a good code is planning.
- Do not be in a hurry to finish the program. Take your time, and be careful when writing the program. It is advised that you get your individual momentum and pressure correction equation solvers working in standalone mode before coupling them together in the outer iteration loop.
- Your entire code should be in the form of smaller subroutines (or functions). If the whole thing is written as one mammoth code, you will find it difficult to debug.
- Seek the instructor's help whenever you have any doubts.
- Do not make assumptions or improvisations. If you alter the original SIMPLE algorithm and take short-cuts, chances are that your code will not converge. Alterations to the original algorithm should only be made after you have had sufficient exposure and experience with it, and understand the full implications of any modification.