Introduction to Fluid Dynamics
Homework Assignment – Numerical solution of Couette Flow
Due: April 21, 2022

## Problem statement

In this assignment, you will solve the unsteady Couette flow using the numerical methods we learned in class. For Couette flow, under the parallel and fully-developed flow assumption, the Navier-Stokes equations simplify to the follow partial differential equation (PDE):

$$\frac{\partial u}{\partial t} = \nu \frac{\partial^2 u}{\partial y^2} \ , \tag{1}$$

where $u(y,t)$ is the streamwise velocity, satisfying the boundary conditions

$$u(0,t) = 0 \ , \quad u(h,t) = U \ . \tag{2}$$

Since the problem is unsteady, we also need an initial condition, which we will take as:

$$u(y,0) = U\frac{y}{h} + U \sin\left(\frac{\pi y}{h}\right) \tag{3}$$

## Methodology

The preliminary step in solving a PDE numerically typically involves non-dimensionalizing it. Restate the problem using the dimensionless variables $u^* = u/U$, $y^* = y/h$ and $t^* = \nu t/h^2$. Henceforth, we will drop the superscript $^*$ for convenience and all variables are understood to be non-dimensional (unless stated otherwise).

The next step is to discretize the domain to generate a grid (or mesh). For the current problem, we will simply use a uniformly spaced grid in $y$-direction. We will denote the number of grid points as $N$. Note, the grid spacing $\Delta y = 1/(N-1)$, and the grid point location $y_j = (j-1)\Delta y$, for $j = 1, 2, ..N$.

Next we will discretize the equation using the finite-difference approximation. We will use 2nd order *central* difference for the spatial derivative and 1st order *backward* difference for the temporal derivative:

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = \frac{u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}}{(\Delta y)^2} \ . \tag{4}$$

Here, $n$ is the the time step number, such that time $t_n = n\Delta t$, for $n = 0, 1, 2...$ This is an unconditionally stable implicit scheme where $u_j^{n+1}$ can be obtained by solving a system of linear equations:

$$r u_{j-1}^{n+1} - (2r+1)u_j^{n+1} + r u_{j+1}^{n+1} = -u_j^n \ , \quad \text{with} \quad r = \Delta t/(\Delta y)^2 \ . \tag{5}$$

The system of equations can be solved by writing them in the vector form $Ax = b$, where $x$ is a vector containing $u_j^{n+1}$, $b$ is a vector based on $u_j^n$, and $A$ is a tridiagonal matrix containing the

coefficients. Note, the solution is only needed at interior points ($2 \leq j \leq N-1$), so one must be mindful of the boundary conditions. In principle, $x$ can be obtained by simply inverting the $A$ matrix, i.e., $x = A^{-1}b$. Since the $A$ matrix is fixed, it can be inverted once at the beginning and reused for time marching. However, given $A$ is tridiagonal, one can also use the Thomas algorithm, which is more computationally efficient and stable compared to matrix inversion. For the current assignment, you are free to use either of the two methods.

### Exact solution and error analysis

It can be shown that the PDE we are trying to solve, with the accompanying boundary/initial conditions, has the following exact solution:

$$u(y,t) = y + \exp(-\pi^2 t)\sin(\pi y) \ . \tag{6}$$

It is evident that $u(y,\infty) = y$, which corresponds to the steady Couette flow solution (the linear velocity profile that we learnt about earlier in the course). Using this knowledge, we can compute two kinds of errors as we numerically solve the equation.

The first error is computed w.r.t. the exact solution, defined as:

$$E_1(t_n) = \left( \frac{1}{N_j} \sum_j |u_j^n - u(y_j, t_n)|^2 \right)^{1/2} \ . \tag{7}$$

This is simply the root-mean-square (RMS) error which tells us how accurate our solution is. Note, the boundary points are not used in calculating the error, since the solution there is always imposed to be exact. Thus, $N_j = N-2$. The second error is based on the steady-state solution:

$$E_2(t_n) = \left( \frac{1}{N_j} \sum_i |u_j^n - u(y_j, \infty)|^2 \right)^{1/2} \ , \tag{8}$$

which evidently tells us how close we are to the steady-state solution. Once the steady-state is reached, we can stop the time marching, since the solution will not change (except for round-off errors). For this reason, we will obtain the solution until the condition $E_2 < \epsilon$ is satisfied, where $\epsilon$ is the allowed tolerance in the round-off error. For the current assignment, use $\epsilon = 10^{-6}$.

For the current assignment, you are required to write a code which implements the numerical scheme to obtain the solution at the grid points and time steps until the steady-state solution is reached, as prescribed by the condition $E_2 < \epsilon$. Try $N = 11, 21, 41, 81$, and for each case try $\Delta t = 0.001, 0.003, 0.01, 0.03, 0.1$. Perform a detailed assessment on the behavior of the error $E_1$ and $E_2$ as a function of both spatial and temporal resolution. Also assess how the numerical solution compares with the exact solution for these cases. For instance, you should investigate things like how the error(s) behaves as a function of $N$ (or $\Delta y$) and $\Delta t$; how it changes as a function of total number of time steps (or time itself) for various cases, or in general, anything else that can think of. A portion of the total points will be awarded on the basis of novelty of the analysis and the conclusions drawn from it.

## Things to submit

1. The source code which solves the problem, with an accompanying readme file with clear instructions on how to compile and run the code. The entire source code, with all functions, subroutines, etc. should be in one file (which can be compiled and executed). Everyone should write their own code. All the relevant parameters of the problem, e.g. $N$, $\Delta t$, $\epsilon$, etc. should be changeable at the beginning the code.

2. A text/data file containing the sample solution corresponding to the case $N = 21$ and $\Delta t = 0.003$, and an accompanying text/data file containing the errors. The first 'solution' file should contain six columns, viz. time step number, time, $y$-location, numerical solution, exact solution (the columns should be clearly readable). Whereas, the second 'error' file should have four columns, viz. time step number, time, $E_1$ error, $E_2$ error.

3. A detailed project report (as a PDF/DOC file) discussing all the various cases/results, with appropriate figures/graphs. You are free to present the analysis in the manner you want, but the report should be no longer than 10 pages (standard formatting), and you are allowed a maximum of 6 figures. Essentially, you have to present your findings in a concise manner, but at the same time ensure all the important and relevant results are conveyed clearly. The problem should be briefly outlined in the first page, and the last page should contain a conclusion/summary section.

## Bonus point

Make sure to run your code with all real number variables in 'double precision' floating-point format. As a bonus point, investigate the effect of using 'single precision' on the error and convergence of the numerical solution. You may also want to play with the parameter $\epsilon$ and assess its role.