Computational PDEs M345N10, 2019-2020, Project 2
The project mark, will be weighted to comprise Total 50% of the overall Module.
Blackboard upload deadline : 11.59pm, 20 March
Please name your files in following way:
Proj2_part_1of_X_yourCID.m (all your Matlab and/or Python scripts, with X the total number of scripts etc.); Your accompanying technical report as: Proj2_yourCID.pdf.
A zipped folder will also be acceptable (ideally), in this case call your zipped folder: Proj2_yourCID.zip (etc.)

Project 2: Elliptic Problem

The equation describing a fluid flowing over a surface is given by the equation

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = 0, \tag{1}$$

where x, y represent spatial coordinates and ϕ is a velocity potential, whereby the velocity perturbations (u, v) in the (x, y)-directions are given by

$$\frac{\partial \phi}{\partial x} = u; \qquad \frac{\partial \phi}{\partial y} = v. \tag{2}$$

A solid surface $y_b(x)$ (a symmetrical biconvex aerofoil) is given by

$$y_b(x) = 2\tau x(1-x), \qquad 0 \leq x \leq 1, \tag{3}$$

where τ denotes the maximum thickness of y_b. The inviscid flow tangency condition on the surface $y_b(x)$ is satisfied through the following:

$$\frac{\partial \phi}{\partial y} - (1 + \frac{\partial \phi}{\partial x})\frac{dy_b}{dx} = 0, \qquad 0 \leq x \leq 1. \tag{4}$$

On either side of the aerofoil, i.e. $y = 0$ a flow symmetry condition of

$$\frac{\partial \phi}{\partial y} = 0, \qquad \text{for} \quad x < 0 \quad \text{and} \quad x > 1, \tag{5}$$

is satisfied, while as $y \to \infty$ the normal velocity $v = 0$; also as $x \to \pm\infty$ the streamwise potential $\frac{\partial \phi}{\partial x} \to 0$, see Figure 1.

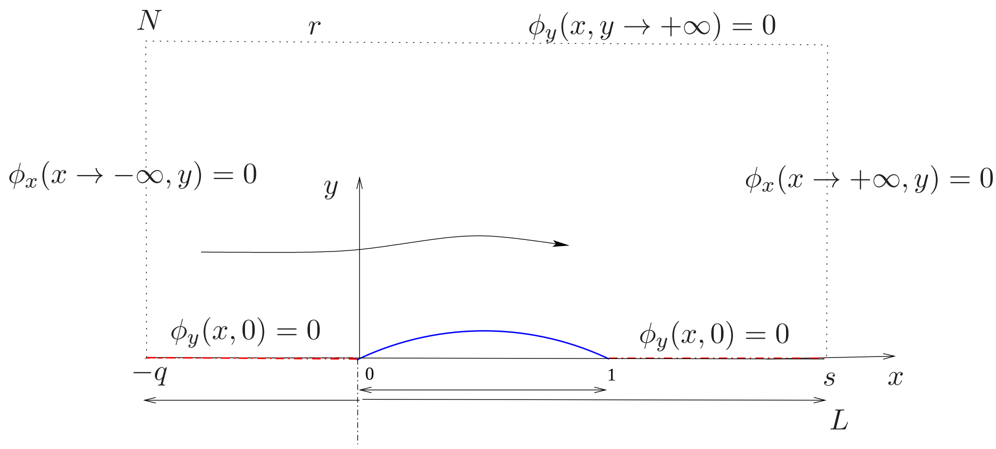

Figure 1: Problem description.

A very good approximation when $\tau << 1$ is to transfer the boundary-condition given by Eqn. 4 to $y_b(x) = 0$, namely :

$$\frac{\partial \phi}{\partial y} - (1 + \frac{\partial \phi}{\partial x})\frac{dy_b}{dx} = 0, \qquad 0 \leq x \leq 1, \quad \text{at} \quad y = 0. \tag{6}$$

as discussed in lecture. You are firstly required (below in parts 1,2 and 4) to use this approximate flow tangency condition and solve Eqn. 1, satisfying the other far-field conditions given.

Define a $L \times N$ rectangular grid and solve Eqn. 1 numerically using second-order finite differences. Your code should be flexible enough to be able to vary the extent of your numerical computational domain on either sides of the aerofoil placement in x, i.e. $(-q : s)$ and and the far-field $y-$domain given by r (note Figure 1).

1. Devise appropriate tests of convergence, and thus explore the merits from a computational efficiency viewpoint the Jacobi, Gauss-Seidel and Successive Over-relaxation (SOR) techniques to solve the PDE for $\tau = 0.05$. You will need to be able to prescribe appropriate criteria of where you place your far-field domains in $x = \pm\infty$ and y_∞, and prescription of number of unknowns (i.e (L, N) grid points) as well as work to the limitations of your computer/laptop! Describe in your reporting the convergence criteria you used to confirm grid independence of your numerical solutions.

   A convenient property of interest is the surface velocity perturbation component $U_{surf} = u = \frac{\partial \phi}{\partial x}$ on the aerofoil surface $x(0 : 1)$, approximated here to be at $y = 0$, which you should use to ascertain what your solution looks like. Make plots of how U_{surf} varies with $x(0:1)$ and along the entire $y = 0, x(-q : s)$ plane. Likewise make plots of the surface $v-$ velocity component along $y = 0$. A colour plot of your entire (u, v) perturbation field solution over your complete (x, y) computational domain will also be instructive – use the pcolor matlab command, or alternative. (20 Marks)

2. For the SOR method make a plot of convergence acceleration achieved as you fix the number of $L \times N$ unknowns with the acceleration parameter ω. Further make plots of optimal ω giving you the fastest convergence as you vary the grid resolution through varying L, N. Compare your findings with the theoretically expected rate/curves and comment. (3 Marks)

3. By a coordinate transformation, namely:

$$\eta = y - y_b(x), 0 \leq x \leq 1, \tag{7}$$

transform Eqn. 1 and thus the flow tangency boundary condition to the new (x, η)-coordinate plane. Hence modify the SOR code developed earlier to solve your transformed equations. Outline clearly in your report the form of the equations that you discretise. Compare your result for $\tau = 0.05$ obtained with the transformed equation set with that found earlier. Subsequently investigate how/whether your solutions differ, as you increase/decrease τ the aerofoil thickness and with solutions obtained with the untransformed equation. Comment on any special features in the solution or *tricky* numerical issues that you had to circumvent to compute the most accurate solutions. (11 Marks)

4. For the Gauss-Seidel method, develop a simple multigrid approach to accelerate convergence of your method. Compare the numerical efficiency (computer timing) and convergence acceleration that arises with your multigrid approach. Choose appropriate and reasonable number of (L, N) grid points that you are able to compute solutions for in good time with your computer. Your reporting should highlight aspects of convergence efficiencies and speedup of your code achieved, as you vary (L, N) with the multigrid approach. (16 Marks)

Use the TestMG.m MultigridV.m residual.m GS.m restrict.m interpolate.m as guides to develop your codes if required.

Notes:

1. Marking will consider both the correctness of your code as well as the soundness of your analysis *and* clarity and legibility of the technical report.
2. In particular describe clearly your treatment of the discretised boundary conditions in your report.
3. All figures created by your code should be well-made and properly labelled.
4. In order to assign partial credit, comment your matlab (or Python) scripts to indicate steps being undertaken or what is being attempted (SHORT COMMENTS!).
5. You are allowed to discuss general aspects of Matlab/Python with each other, however you are trusted not to discuss your code or analysis with other students.

Dr M. S. Mughal
6 March, 2020