AE 6042 Computational Fluid Dynamics

Computer Problem # 1

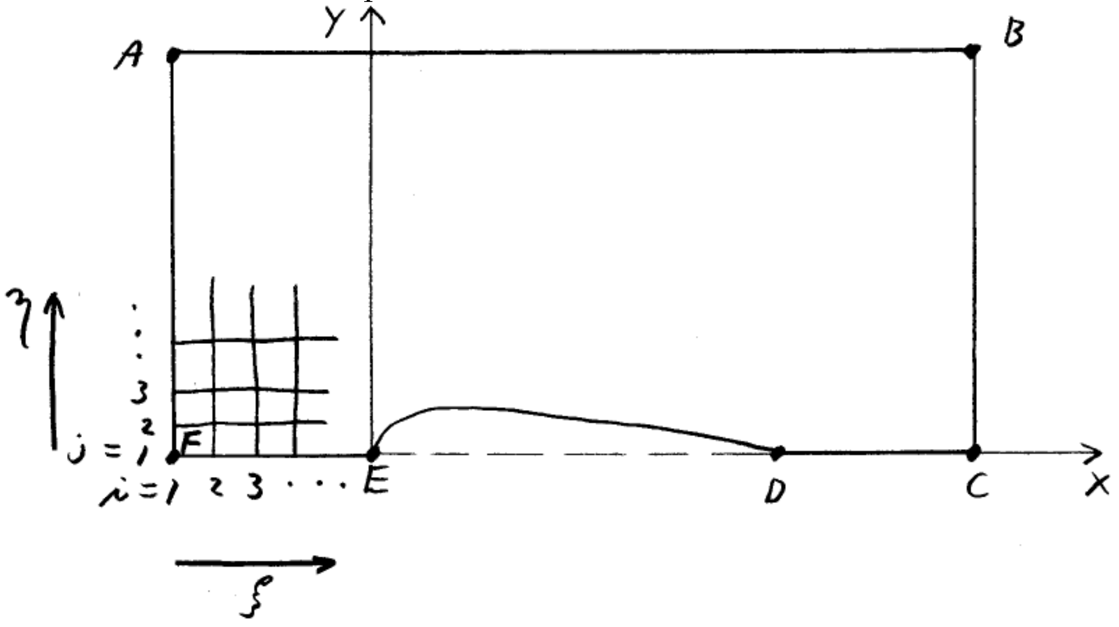
Grid Generator

In this exercise you will generate an inviscid, 2-D computational grid around a modified NACA 00xx series airfoil in a channel. The thickness distribution of a modified NACA 00xx series airfoil is given by

$$y(x) = \pm 5t\left[0.2969\sqrt{x_{int}x} - 0.126x_{int}x - 0.3516(x_{int}x)^2 + 0.2843(x_{int}x)^3 - 0.1015(x_{int}x)^4\right]$$

where the "+" sign is used for the upper half of the airfoil, the "-" sign is used for the lower half and x_{int}=1.008930411365. Note that in the expression above *x, y,* and *t* represent values which have been normalized by the airfoil chord. In this problem we will consider a maximum thickness of t=0.15.

A sketch of the computational domain is shown below.



Each grid point can be described by (*x,y*) location or (*i,j*) location where *i* is the index in the ξ direction and the j index is in the η direction. The grid should have imax=41 points in the ξ direction and jmax=19 points in the η direction. The coordinates of points A-F shown in the figure are given in the following table:

Point	**(i,j)**	**(x,y)**	
A	(1,19)	(-0.8,1.0)	
B	(41,19)	(1.8,1.0)	
C	(41,1)	(1.8,0.0)	
D	(31,1)	(1.0,0.0)	(trailing edge)
E	(11,1)	(0.0,0.0)	(leading edge)
F	(1,1)	(-0.8,0.0)	

Algebraic Grid

To complete this project you will first generate a grid using algebraic methods. Use uniform spacing in the x direction along FE, along ED, and along DC. (However, note that the spacing in the x direction along FE and DC will be different from the spacing along ED). Use uniform spacing in the x direction along AB. (However, note that the spacing in the x direction along AB will be different from that along FE, ED, and DC). For the interior points of the initial algebraic grid use a linear interpolation (in computational space) of the boundary x values:

$$x(i,j) = x(i,1) + \left(\frac{j-1}{jmax-1}\right)[x(i,jmax) - x(i,1)]$$

Use the following stretching formula to define the spacing in the y direction:

$$y(i,j) = y(i,1) - \frac{y(i,jmax) - y(i,1)}{C_y} ln\left[1 + (e^{-C_y} - 1)\left(\frac{j-1}{jmax-1}\right)\right]$$

where C_y is a parameter that controls the amount of grid clustering in the y-direction. (If nearly uniform spacing were desired we would use C_y =0.001).

The algebraic grid generated now serves as the initial condition for the subroutines which generate the elliptic grid. The boundary values of the initial algebraic grid will be the same as those of the final elliptic grid.

Elliptic Grid

The elliptic grid will be generated by solving Poisson Equations,

$$\xi_{xx} + \xi_{yy} = P(\xi, \eta)$$

$$\eta_{xx} + \eta_{yy} = Q(\xi, \eta)$$

where the source terms, P and Q, provide control over interior point distributions. The Middlecoff-Thomas method is used for the control terms:

$$P = \left(\xi_x^2 + \xi_y^2\right)\phi(\xi, \eta)$$

$$Q = \left(\eta_x^2 + \eta_y^2\right)\psi(\xi, \eta)$$

Thus the governing equations can be written:

$$A_1(x_{\xi\xi} + \phi x_\xi) - 2A_2 x_{\xi\eta} + A_3(x_{\eta\eta} + \psi x_\eta) = 0$$

$$A_1(y_{\xi\xi} + \phi y_\xi) - 2A_2 y_{\xi\eta} + A_3(y_{\eta\eta} + \psi y_\eta) = 0$$

where the A's were given in class.

On the boundaries, ϕ and ψ are defined as follows:

$$\text{On } j = 1 \text{ and } j = \text{jmax}: \quad \phi = \begin{cases} -\dfrac{x_{\xi\xi}}{x_\xi} \text{ for } |x_\xi| > |y_\xi| \\ -\dfrac{y_{\xi\xi}}{y_\xi} \text{ for } |x_\xi| \le |y_\xi| \end{cases}$$

$$\text{On } i = 1 \text{ and } i = \text{imax}: \quad \psi = \begin{cases} -\dfrac{x_{\eta\eta}}{x_\eta} \text{ for } |x_\eta| > |y_\eta| \\ -\dfrac{y_{\eta\eta}}{y_\eta} \text{ for } |x_\eta| \le |y_\eta| \end{cases}$$

At interior points, ϕ and ψ are found by linear interpolation (in computational space) of these boundary values. For example,

$$\psi_{i,j} = \psi_{1,j} + \frac{i-1}{imax-1}\left(\psi_{i\max,j} - \psi_{1,j}\right)$$

For all derivatives, central difference approximations (such as those shown below) are used:

$$(x_\xi)_{i,j} \approx \frac{x_{i+1,j} - x_{i-1,j}}{2\,(\Delta\xi)}$$

$$(x_{\xi\xi})_{i,j} \approx \frac{x_{i+1,j} - 2x_{i,j} + x_{i-1,j}}{(\Delta\xi)^2}$$

$$(x_{\xi\eta})_{i,j} \approx \frac{x_{i+1,j+1} - x_{i+1,j-1} - x_{i-1,j+1} + x_{i-1,j-1}}{4\,(\Delta\xi)(\Delta\eta)}$$

where we will use $\Delta\xi = \Delta\eta = 1$ at all grid locations.

In this exercise a Line Gauss-Seidel Iterative procedure is used to solve the governing equations. In this procedure, solutions are obtained along successive j=constant grid lines. To employ this technique, use finite difference approximations to discretize the governing equations. Assign time index "n" to x and y terms on the j+1 line and assign time index "n+1" to x and y terms on the j and j-1 lines. However, base A_1, A_2, A_3 only on x and y values at time index "n". Next, arrange the finite difference approximations for the governing equations into the following forms:

$$b_{i,j}\, x_{i-1,j}^{n+1} + d_{i,j}\, x_{i,j}^{n+1} + a_{i,j}\, x_{i+1,j}^{n+1} = c_{i,j}$$

$$\beta_{i,j}\, y_{i-1,j}^{n+1} + \delta_{i,j}\, y_{i,j}^{n+1} + \alpha_{i,j}\, y_{i+1,j}^{n+1} = \gamma_{i,j}$$

where $c_{i,j}$and $\gamma_{i,j}$ are not functions of x or y on line j. Notice that on each line each j=constant line each of these expressions fits the tri-diagonal matrix form shown in section 4.3.3 of the textbook. Thus the system can be solved on a j=constant grid line using the Thomas Algorithm described in that section and given in the Appendix A. On each j=constant line, i ranges from 1 to imax and the (x,y) values specified at the boundaries will be held fixed by using:

$$b_{1,j} = b_{i\max,j} = \beta_{1,j} = \beta_{i\max,j} = a_{1,j} = a_{i\max,j} = \alpha_{1,j} = \alpha_{i\max,j} = 0$$
$$d_{1,j} = d_{i\max,j} = \delta_{1,j} = \delta_{i\max,j} = 1$$
$$c_{1,j} = x_{1,j} \quad c_{i\max,j} = x_{i\max,j} \quad \gamma_{1,j} = y_{1,j} \quad \gamma_{i\max,j} = y_{i\max,j}$$

The Thomas Algorithm is utilized for j=2,3,4,...,jmax−1 for each of the governing equations. This sweeping procedure is repeated until convergence is achieved.

Demonstrate your solver by generating 5 grids (each with 41x19 grid points):

Grid 1: Initial algebraic grid, non-clustered (C_y =0.001)
Grid 2: Initial algebraic grid, clustered (C_y =2.0)
Grid 3: Elliptic grid, clustered (C_y=2.0), no control terms ($\phi = \psi = 0$)
Grid 4: Elliptic grid, clustered (C_y=2.0), with control terms
Grid 5: Now, use your program to generate the best grid you can for inviscid, subsonic flow in the geometry shown. You must keep imax=41, jmax=19 and not change the size or shape of the outer and wall boundaries. You may, however, change the grid spacing along any and all of the boundaries and use different levels of grid clustering wherever you think it is appropriate.

What to Turn In
In the printed report that you turn in please provide the following:

1. A brief description of the grid generation method and show discretized expressions for the coefficients of the tri-diagonal system. (3 page maximum)
2. Plots of each of the 5 grids generated (showing all grid lines).
3. State the convergence criterion and submit plots of the RMS residual (log scale) vs. Iteration No. (linear scale) for Grids 3 - 5.
4. A discussion of the results of each of the grid generation cases and the quality of each of the grids. Describe the differences in the grids and the effects of the control terms. Describe your strategy for generating Grid 5 and why it is better than the other 4 grids. (2 page maximum)
5. A printed copy of your code **and** an electronic copy of your code. These must be accompanied with detailed instructions describing how the instructor can run your code for the various cases. Also state the machine type, and compiler type used to compile and run the code. The electronic copy of the code and instructions should put in the "Drop Box" on the T-square site for the course. The program should be clearly written, commented and be original work (except the Thomas algorithm).