ME 163	Engineering Aerodynamics	Prom Putthisri
Spring 2021	Prof. Omer Savas	Project 2

1 Abstract

The objective of this project is to use programming skills to create simple computational aerodynamic machine on a cylindrical airfoil. In this project, there will be simulation of what happened on the surface of an airfoil with simple geometry during a none accelerating flight with no angle of attack (no lift) and with 30° angle of attack (lift). Employing linearly varying vortex panels, there will be calculations coefficient of pressure, normal velocity, tangential velocity, and stagnation points on each control points using Vortex Panel Method and Kutta Conditions. There will also be calculations of separation points using the boundary layer separation criterion of Thwaites.

2 Project Materials

The main problem in both computational and analytical fluid dynamic is the chaotic behavior of the fluid itself. In this project we hope to tackle that challenge by analysing a moving airfoil. To make the problem more suitable for the current knowledge from the class, the shape of an airfoil that are going to be analyse in this project will be reduce to a circular cylinder with radius of 1 and the movement of the airfoil will be replicated by a constantly moving uniform flow U(u,v) = [1, 0]. In this project, Matlab will be used as main programming language.

2.1 Panels Setup

To replicate the circular airfoil surface layout, a function called *locateCylinderControlPanel(np, r, theta)* is created, where *np* is the number of vortex panel; *np* is the radius of the circular airfoil; and *theta* is the angle in which Kutta condition and trailing edge is placed. The function create sets of X and Y coordinate representing location of control panels and edges. After which, a function called *findPanelCenter(X, Y)* and *findPanelTheta(X, Y)* are created to take in those X and Y coordinate and find center point and angle of each control panels.
Note: As the number of *np* increases, the closer the closer it is for the simulated boundary edge to the ideal boundary edge at Kutta condition.

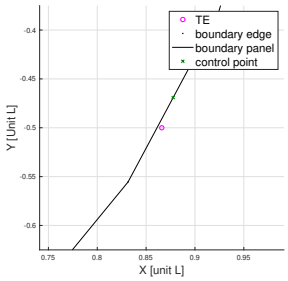

Figure 1: 32 panels model

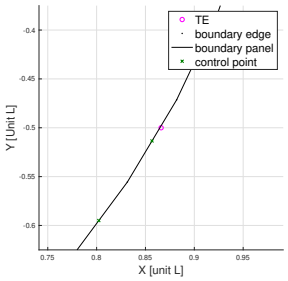

Figure 2: 64 panels model

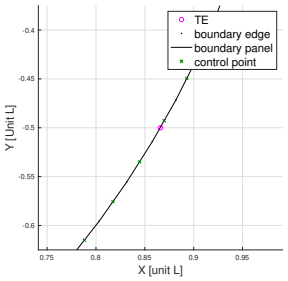

Figure 3: 128 panels model

2.2 Vortex Panel Method

The heart of this project lies within this section, the vortex panel method. The potential function of the combined fild of the uniform flow and the vortex panels is

$$\phi(x,y) = Uxcos(\alpha) + Uysin(\alpha) + \sum_j \int_j \frac{\gamma(s_j)}{2\pi} tan^{-1} \frac{(y-y_j)}{(x-x_j)} ds_j$$

where the vortex panel strength caries linearly, the vaniching normal velocity components at control points i, and the Kutta conditions are

$$\gamma(s_j) = \gamma_j + (\gamma_{j+1} - \gamma_j)\frac{s_j}{S_j}$$
$$\frac{\partial \phi}{\partial n} \mid_i = 0$$
$$\gamma_i + \gamma_j = 0$$

After solving the above equation and the twos conditions, we will be able to arrived at the following matrix.

$$\begin{bmatrix} C_{n1_{(1)(1)}} & C_{n2_{(1)(2)}} + C_{n1_{(1)(3)}} & ... & C_{n2_{(1)(m-2)}} + C_{n1_{(1)(m-1)}} & C_{n1_{(1)(m)}} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ C_{n1_{(n)(1)}} & C_{n2_{(n)(2)}} + C_{n1_{(n)(3)}} & ... & C_{n2_{(n)(m-2)}} + C_{n1_{(n)(m-1)}} & C_{n1_{(n)(m)}} \\ 1 & 0 & ... & 0 & 1 \end{bmatrix} \begin{bmatrix} \gamma_1 \\ \\ \vdots \\ \\ \gamma_{n+1} \end{bmatrix} = \begin{bmatrix} 2\pi U \sin(\theta_1 - \alpha) \\ \\ \vdots \\ \\ 2\pi U \sin(\theta_n - \alpha) \\ 0 \end{bmatrix}$$

Each Cn is a function of control points and edges spatial information. In Matlab, create function called $vortexPanelMethod(edgesInformation, controlPointsInformation, \alpha)$ where α is the rearrange angle of attack from leading edge point of view to solve the matrix equation. Once solved for the γ matrix, the tangential velocity on each control point can be found by plugging in the value of γ into the equation below.

$$U_{ti} = 2\pi cos(\theta_i - \alpha) + \sum_{j=1}^{m} (C_{t1_{ij}\gamma_j} + C_{t2_{ij}\gamma_{j+1}})$$

In Matlab, combine the above equation into $vortexPanelMethod(edgesInformation, controlPointsInformation, \alpha)$ to prevent memory lost and wasteful run-time to find Cn and Ct.
Note: The definitions and value of all Cn and Ct terms is extremely lengthy and the the actual derivation of the matrix is too long for this report. They can be found at ME163 lecture note on page 168.

2.3 Coefficient of Pressure

The Cp could be found from the two equations below. In Matlab, create $findCp(U, \alpha, \theta)$ where θ the angle of each control panels around the airfoil and U is the power of the stream or velocity of an airfoil. The function return Coefficient of pressure, Cp, against each theta around the airfoil (from Green's theorem compile to 2π despite the shape).

$$\Gamma = 4\pi a U sin(\alpha)$$
$$Cp = 1 - (2sin(\theta + \frac{\Gamma}{2\pi a U}))^2$$

2.4 Stagnation Points

Recall that the trailing edge of the airfoil will always be the place that the Kutta condition is placed, but what about the leading edge? As it turns out, the tangential velocities at the control points around the airfoil are crucial in determining the leading edge. To find this, create a function called $findStagnation(Ut)$, where

Ut is the tangential velocity around an airfoil, that return index of edge where the velocity reverses from positive to negative. The spot where the velocity changes from negative to positive is where the stagnation points (trailing and leading edge) are. This phenomenon occurs due to the fact that, at the stagnation point, the tangential velocity diverges. One moves in clockwise direction and the other moves in counter-clockwise direction until reaching the other stagnation points where the direction reverse again, leading to the changes in sign of the velocities.

2.5 Thwaites' Method

Most fluids flow in a somewhat predictable manner, until reaching a point where the flows are no longer recognize or predictable. One could imagine flow over flat plate where the first half of the flow is laminar when after a threshold it turns turbulent. An airfoil also has that threshold. In this case, it is called separation points. To determine this, one would need to implement function that solve for the boundary layer separation criterion of Thwaites.

$$K(x_s) = \frac{0.45}{U_e^6(x_s)}\left(\frac{dU_e}{dx}(x_s)\right)\int_0^{x_s} U_e^5(\xi)d\xi = -0.09$$

(U_e is the tangential velocity around the airfoil from the stagnation points and x_s is the distance travel from a stagnation points to another. From this it is clear that there are two paths created, the upper boundary layer and the lower boundary layer. To program this, create a function that takes in U_e and x_s and return the the first point where the criterion of Thwaites is met. Notice that there will be need to be two calculation one for the upper boundary layer and one for the lower boundary layer of an airfoil. Since we are integrating with respect to the boundary control points, make sure that the the distance and velocity collected from both upper and lower layer are taken at the control points.

3 Required Calculation Results

After coding the complete project of the rough direction above in detail. I have run the program on six different sets of parameter, 32 panels no-lift, 64 panels no-lift, 128 panels no-lift, 32 panels lift, 64 panels lift, and 128 panels lift, where lift angle is -150◦ or -30◦ angle of attack. Here are the result figures from the simulations.

3.1 Coefficient of Pressure Plots of the no-Lift Cases

As mentioned in lecture, the pressure around the circular airfoil will look like a person with symmetrical hat and beard with a set of symmetrical ears. The result shown below demonstrate just that in terms of magnitude of the pressure against each angle of the circular airfoil. Notice that as the number of control points increases the more smooth the plot. In reality, there are infinite infinitesimal control points making the Coefficient of Pressure a smooth function.

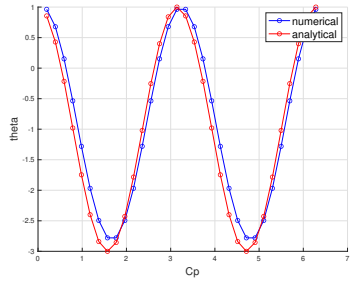

Figure 4: Cp of 32 panel model with 0 angle of attack

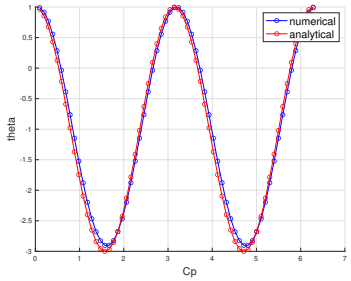

Figure 5: Cp of 64 panel model with 0 angle of attack

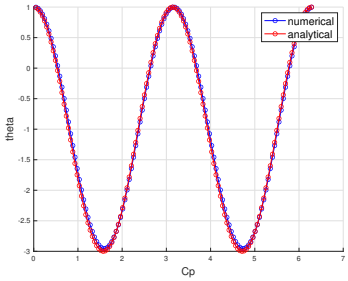

Figure 6: Cp of 128 panel model with 0 angle of attack

3.2 Coefficient of Pressure Plots of the Lift Cases

Similar to the Coefficient of Pressure Plots of the no-Lift Cases, as the number of control points increases the more smooth the plot. However, it is clear that the distribution of pressure around the airfoil is different than that of the no-lift case. There is a strong lower coefficient of pressure and the negative angle from Trailing Edge and higher coefficient of pressure at the positive side. This Show that the force acting the upper and lower boundary of the airfoil is not equal. The force at the bottom is higher, exerting lift force on the airfoil.

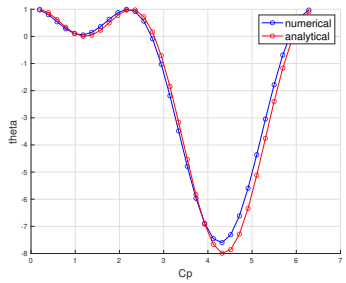

Figure 7: Cp of 32 panel model with 30 angle of attack

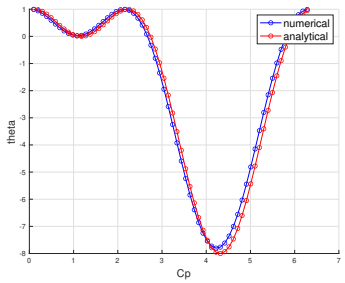

Figure 8: Cp of 64 panel model with 30 angle of attack

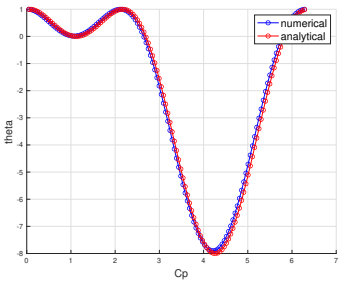

Figure 9: Cp of 128 panel model with 30 angle of attack

3.3 Stagnation and Separation Points of Each Cases

The stagnation points of the airfoil is the trailing and leading edge of its shape. For the no-Lift case, the separation points are symmetric across the x-axis, this is due to the fact that the flow coming into the airfoil is symmetric with respect to the x-axis. In reality flight, however, gravity will play an insignificant roles in the density of the flow making it non-uniform, but the different is extremely minor in most aircraft. As for the Lift case, the separation points are no longer symmetric across x-axis. The longer duration in which air particle moves along a supportive gradient field along the upper boundary layer allows air particles to move further along the boundary.
Note: As the number of control points increases the closer the separation points are toward the y-axis, which is the ideal case presented in the textbook.

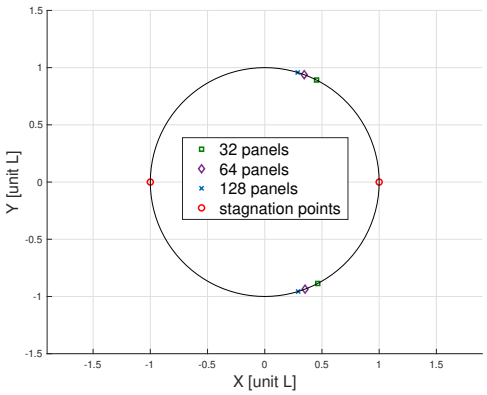

Figure 10: separation points of 0 angle of attack model

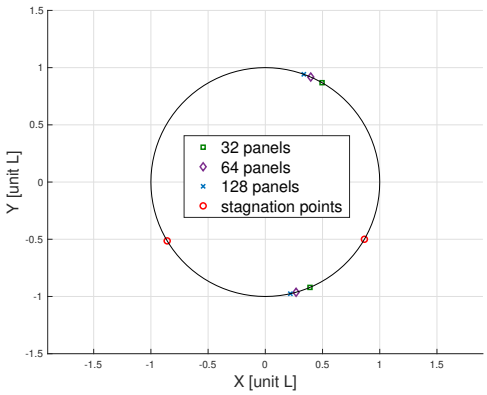

Figure 11: separation points of 30 angle of attack model