ME C180 HW 8: Airfoil Analysis Using Finite Elements

Mitchell Robinson

May 8, 2015

Problem

When learning about aerodynamics and certain flow patterns around airfoils it is often difficult to see the streamlines and velocity in real time. Thin airfoil theory and mathematics are used to described these complex interactions that take place in the air. The problem I sought to solve was how to reconcile thin airfoil theory with what the finite element model tells us. By using the national Advisory Committee for Aeronautics (NACA) data base, I chose to examine a slightly cambered airfoil that could be used on an commercial aircraft, the 9412. It is worth noting that the flow velocities used in the simulations were not prescribed at velocities that the airfoil would experience in real life; however, the fluid and surface interaction, stall angle affects, and transient flow patters all observed during these simulations are still applicable as learning tools for examining airfoils. To examine the airflow characteristics the Navier-Stokes equations were used to relate pressure and velocity (1 and 2).

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla P + \nu \nabla^2 \mathbf{u} \tag{1}$$

$$\nabla \cdot \mathbf{u} = 0 \tag{2}$$

The partial differential equations listed above help to describe the motion of viscous fluid substances. For reference, $\mathbf{u}$ represents the velocity, P is the pressure, and ν is the dynamic viscosity of the material. The left most term describes the derivative of the velocity term, or the acceleration of the fluid. The second term on the left side of the equals represents the convection of the flow. The gradient for the pressure and diffusion terms on the left represent the divergent components of the flow, while the left side represents the internal qualities of the flow. The constraint equation listed second tells us that the flows divergence is 0.

For this problem, the simulation was to view the wing flying in air, therefore the top, bottom, and right side of the mesh had boundary conditions of pressure equal to 0. The inflow velocity was a specified velocity of 10 to simulate air coming at the wing. The wing itself was a large no slip boundary because it is a solid body that is not moving relative to the air. Using the above Dirichlet boundary conditions, allowed for an accurate simulation of the airfoil to be done.

Methods

This problem was more advanced than normal finite element problems as it the solution was time dependent and involved two variables: pressure and velocity. To solve flow problems involving Navier-Stokes equations, a variety of algorithms can be used. Such examples include the incremental pressure correction scheme (IPCS), Chorin's Projection method, Consistent splitting scheme, a least-squares stabilized Galerkin method (G2), and finally, A saddle point solver for a pure Galerkin discretization (GRPC). Chorin's projection method was chosen as a suitable method for it's easy implementation. For further reference of these other methods see Chapter 21 of FEniCS project online PDF resource.

Chorin's projection method begins with computing a tentative velocity step by neglecting pressure in the momentum equation. This projects the velocity onto a vector filed with no divergent restrictions as shown in the equation below where the * denotes the intermediate velocity step being taken.

$$\frac{\partial \mathbf{u}}{\partial t} = -\mathbf{u} \cdot \nabla \mathbf{u} + \nu \nabla^2 \mathbf{u} + f \tag{3}$$

Now, we will apply the first steps to solve this new equation by putting it into the weak form. We first multiple by a weight function an perform integration by parts.

$$\int \frac{\mathbf{u}^* - \mathbf{u}^{n-1}}{\Delta t} w d\Omega = \int -\mathbf{u} \cdot \mathbf{u}^{n-1} w d\Omega + \nu \int \nabla^2 \mathbf{u}^{n-1} w d\Omega + \int f w d\Omega - \int \nabla \mathbf{u}^{n-1} \nabla w d\Omega \tag{4}$$

After solving this initial step, we then add pressure back to the equation in the form shown below.

$$\frac{\partial u}{\partial t} = -\nabla P \tag{5}$$

or in discrete form

$$\frac{\mathbf{u}^* - \mathbf{u}^{n-1}}{\Delta t} = -\nabla P \tag{6}$$

From here, to solve we need to take the divergence of both sides, yielding the equation below.

$$\frac{-\nabla \cdot \mathbf{u}^*}{\Delta t} = -\nabla^2 P \tag{7}$$

Now the only unknown left to solve for is pressure. We can again implement weak form techniques by multiplying both sides by a new weight function q and then doing integration by parts. This process is shown below in integral form.

$$-\int \nabla^2 P q d\Omega = \int \frac{\nabla \cdot \mathbf{u}^*}{\Delta t} q \, d\Omega \tag{8}$$

With integration by parts being applied

$$-\int \nabla P \nabla q d\Omega = \int \frac{\nabla \cdot \mathbf{u}^*}{\Delta t} q \, d\Omega \tag{9}$$

This equation can now be easily implemented into FEniCS as done in problems before.
Lastly, the pressure term solved for is used to correct the intermediate velocity step and solve for the velocity at the full time step. This is done by using the correction term for the equation below, which reintegrates the pressure back into the momentum equation. Since pressure is known, we can calculate velocity at n.

$$\frac{\mathbf{u}^n - \mathbf{u}^*}{\Delta t} = -\nabla P \tag{10}$$

Now plugging into the the same formula as was done initially with velocity weight function w.

$$\int \frac{\mathbf{u}^n - \mathbf{u}^*}{\Delta t} w \, d\Omega = -\int \nabla P \cdot w \, d\omega \tag{11}$$

Integration by parts once more.

$$\int \mathbf{u}^n - \mathbf{u}^* w \, d\Omega = -\Delta t \int P(\nabla \cdot w) \, d\omega \tag{12}$$

This final step was implemented last in the while loop for the problem being solved in FEniCS. The code that implements this algorithm is available for reference and has been uploaded under the filename "HW8.py" along with this report.

Implementation

FEniCS software was used to implement the above derivation for the Navier - Stokes problems into the weak form. FEniCS allowed the mesh to imported directly from a MATLAB program I altered that could take the array of x and y data points for the given airfoil and make a 2D mesh from the specifications.

To build the mesh, first, the meshdemo function was saved as a different file so that it could be altered to meet the problem specifications. By altering the structure of the data that was used as mesh points, as well as the rectangular boundaries that the airfoil would lie in, a well suited mesh program could be made. This program is attached to files uploaded for further reference. The meshes for the airfoil with three varying angles of attack are shown below.

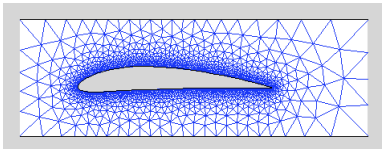

Figure 1: Mesh used for 9412 airfoil at an attack angle of 0 degrees.

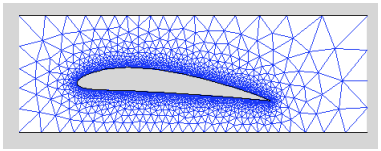

Figure 2: Mesh used for 9412 airfoil at an attack angle of 5 degrees.

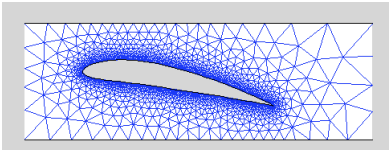

Figure 3: Mesh used for 9412 airfoil at an attack angle of 10 degrees.

The mesh was done out of triangular elements to help simplify the solving process and cover the complex geometry of the airfoil. Also, the mesh was constructed in such a way that the elements got smaller and smaller as you neared the boundary edge of the airfoil.

Once the appropriate mesh could be loaded into FEniCS, the algorithms displayed above could begin solving the flow. The first step in the FEniCS implementation was to define the the trial and test function spaces. The pressure was a simple function space since pressure is a scaler value, while the velocity function space was a vector function space. next, the test and trial functions were defined as follows: u and v as the velocity trial and test functions, and p and q as the pressure trial and test functions. Next, the parameter values describing the time step, the total time, the tolerance for the mesh, and the dynamic viscosity of the fluid were set. Then the velocity inflow for the problem was specified. Subdomains were created next in order to enforce the boundary conditions expressed earlier. These subdomains were then used along with the FEniCS function DirichletBC, to specify the no slip and pressure conditions for the problem. The functions for the velocity and pressure variables was next satisfied. With everything in place, the first tentative step of the velocity (Equation 4) was done. Following this the pressure update in Equation 9 was then implemented directly. Finally, the full velocity step was calculated by direly plugging in Equation 12. Now that all the values had been defined, the matrices for these three equations were assembled, and the program then implemented a while loop to solve these equations in the given format while the time was less than the full time that was set. The GMRES condition was used on the solver. For further reference on the Chorin method, see the FEniCS online project resource Ch. 21.

A couple things to note, the time step chosen for this problem varied depending upon the angle of attack. For the lower angles of 0 and 5, a time step of 0.0005 seconds could be used; however, they were only suitable for a total time of $t = 0.3s$ at a velocity of $v = 10$. At an attack angle of 10 degrees, the time step needed to be lowered to $t = 0.00001s$ at a velocity of $v = 3$. When prior simulations were done at larger time steps, the solver always became unstable near the trailing edge of the airfoil. This could be explained by the effect of the Kutta condition for airfoils undergoing steady flight. According to Keuthe and Schetzer's *Foundations of Aerodynamics* "A body with a sharp trailing edge which is moving through a fluid will create about itself a circulation of sufficient strength to hold the rear stagnation point at the trailing edge" [1]. This stagnation point near the edge of the airfoil is a result of a large pressure gap that causes air to be pushed against the tail. This phenomenon of a large pressure differences and large velocities explains why the simulation had trouble running at larger time steps when the parameters could not be accounted for resulting in instability. Once all the simulations were run using the appropriate methods, results could be recorded and analyzed in Paraview.

Results

Below are the flow results for the 9412 airfoil at an attack angle of 0 degrees.

Figure 4: Pressure distribution for the 9412 airfoil at 0.3 s and $\alpha = 0^o$.

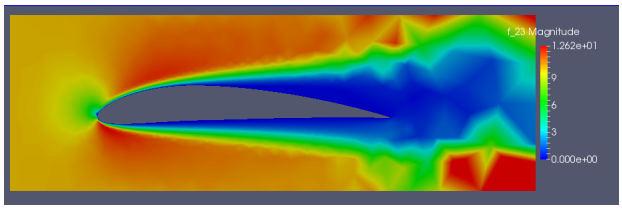

Figure 5: Velocity distribution for the 9412 airfoil at 0.30 s and $\alpha = 0^o$.

Below are the flow results for the 9412 airfoil at an attack angle of 5 degrees.

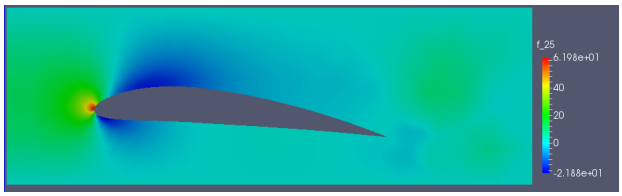

Figure 6: Pressure distribution for the 9412 airfoil at 0.34 s and $\alpha = 5^o$.

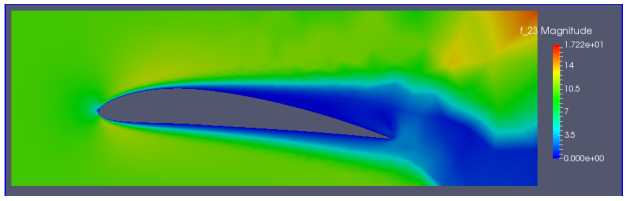

Figure 7: Velocity distribution for the 9412 airfoil at 0.34 s and $\alpha = 5^o$.

Below are the flow results for the 9412 airfoil at an attack angle of 10 degrees.

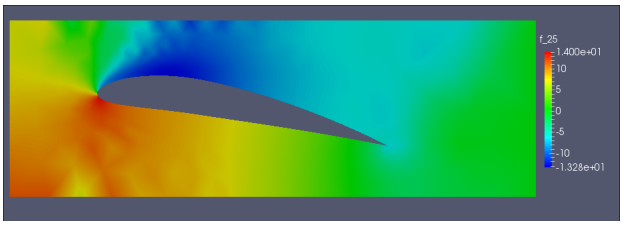

Figure 8: Pressure distribution for the 9412 airfoil at 0.34 s and $\alpha = 10^o$.

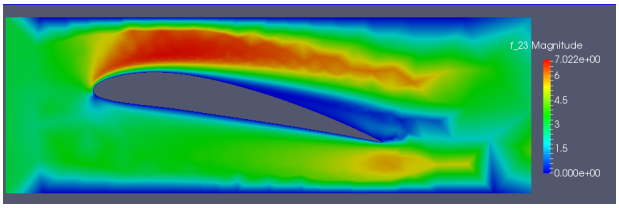

Figure 9: Velocity distribution for the 9412 airfoil at 0.34 s and $\alpha = 10^o$.

Conclusions

The figures shown in the results section display the steady characteristics of air flowing over a 2 dimensional airfoil. In the steady state conditions, we can see that the given pressure distribution for all three cases shows a positive lift force by having higher pressure below the wing and lower pressure above. In addition. the velocity of all three simulations shows how the velocity increases above the airfoil, which is a necessary factor to have a pressure decrease since they are both part of the continuum equation we are analyzing. Given that we observe these commonalities to airfoil theory, these results can be assumed to be generally correct.

As can be seen in the figures, the distribution for velocity and pressure differences increases with attack angle. At an attack angle of 0, the wing does not do much in terms of lift; however, adjust the angle slightly and we see drastic results such as in the 10 degree case.

Another conclusion worth noting is the fact that the lift increases with increasing angle of attack. this makes sense with thin airfoil theory as when the angle of attack is high, more air is directly aimed at contacting the bottom side of the wing, driving it upward with lift force; another way to think about it is that the velocity going over the top of the wing has to travel that much faster, resulting in less pressure on top of the wing. In conclusion, these simulations help to provide insight into the basics of aerodynamics and prove some basic fundamentals of airflow over rigid bodies.

References

1. Anders Logg, Kent - Andre Mardal, Garth N. Wells. *Fenics project: Lecture Note in Computational Science and Engineering* Springer. 2012. PDF.

2. A.M. Kuethe and J.D. Schetzer, Foundations of Aerodynamics, Section 4.11 (2nd edition), John Wiley and Sons, Inc., New York (1959)