ELIMINATING FLOW SEPARATION AND REDUCING VISCOUS DRAG THROUGH BOUNDARY LAYER ANALYSIS AND MANIPULATION

Matthew D. Oser (LARSS Student)
Richard L. Campbell (Mentor)

Research and Technology Group
Aerodynamics Division
Transonic/Supersonic Aerodynamics Branch

ABSTRACT

As both computers and flow-analyzing equations have increased in sophistication, computational fluid dynamics (CFD) has evolved into a fixture for advanced aircraft design. While CFD codes have improved in accuracy and efficiency, their ability to encompass viscous effects is lacking in certain areas. For example, current CFD codes cannot accurately predict or correct for the increased drag due to these viscous effects at some flow conditions. However, by analyzing an airfoil's turbulent boundary layer, one can predict not only flow separation via the shape factor parameter, but also viscous drag via the momentum thickness. Various codes have been written which can calculate turbulent boundary layer parameters. The goal of my research is to develop procedures for modifying an airfoil (via its local pressure distribution) to eliminate boundary layer separation and/or to reduce viscous drag. The modifications to the local pressure distribution necessary to achieve these objectives will be determined using a direct-iterative method installed into a turbulent boundary layer analyzer. Furthermore, the modifications should preserve the basic characteristics of the original airfoil.

ELIMINATING FLOW SEPARATION AND REDUCING VISCOUS DRAG THROUGH BOUNDARY LAYER ANALYSIS AND MANIPULATION

As both computers and flow-analyzing equations have increased in sophistication, computational fluid dynamics (CFD) has evolved into a fixture for advanced aircraft design. While CFD codes have improved in accuracy and efficiency, their ability to encompass viscous effects are lacking in certain areas. For example, current CFD codes cannot accurately predict or correct for the increased drag due to these viscous effects at some flow conditions. However, by analyzing an airfoil's turbulent boundary layer, one can predict not only flow separation via the shape factor parameter, but also viscous drag via the momentum thickness. Various codes have been written which can calculate turbulent boundary layer parameters. My research project involved developing a procedure for modifying an airfoil (via its local pressure distribution) to eliminate boundary layer separation and/or to reduce viscous drag. The modification to the local pressure distribution necessary to achieve these goals would be determined using a direct-iterative method.

My research project required a turbulent boundary layer analyzer. I selected a code, written by Gary Warren, which utilizes Green's direct lag-entrainment method. This code solves for boundary layer parameters, such as displacement thickness, momentum thickness (Θ), and shape factor (H), based on inputted free stream conditions, local pressure (C_P) distribution, and an initial value of Θ. Unfortunately this code was written in FORTRAN, while I had learned C. I therefore borrowed a FORTRAN manual and spent my first few nights learning FORTRAN.

My first task was to edit the code to input data from a more convenient data file. The code originally input data from an archaic namelist that was time-consuming to modify. With this task accomplished, I needed to run some test cases in order to verify the code's accuracy. I found an AGARD report that contained experimental boundary layer data for various airfoil configurations. I selected three test cases: The first case exhibited completely attached flow, the second case exhibited flow that neared separation, and the third case produced separated flow. The code yielded very accurate theoretical data for the first two cases. One drawback of the direct lag-entrainment method is its inability to produce accurate boundary layer data for separated flow cases. However, my code remained fairly accurate for even the third test case. Confident in the boundary layer analyzer, I was ready to proceed with my research.

In order to examine the effects alterations to the C_P distribution would have on the H distribution, I needed an easy way to qualitatively compare my calculated data. The original code output data into a large table. It was difficult, and very time-consuming, to compare various tables from different test cases. I therefore modified my code to graph both C_P and a user specified boundary layer parameter versus x/c location. As a result, I could quickly compare multiple sets of data. I also required a convenient method for modifying the inputted C_P distributions. I chose to approximate each C_P distribution with a system of linear segments joined together at "control points." (Figure 1) I altered the code to present the user the option of inputting a C_P distribution via these control point coordinates. I also created an option which allows the user to interactively move one of these control points using a mouse.

After these modifications, I was ready to contrast H distributions produced by various C_P distributions. I chose a base-line C_P distribution composed of five linear segments as defined by six control points (Figure 1). I created dozens of test cases by varying the height and/or slope of these segments. After observing the resulting changes in the H distributions, I developed some fundamental conclusions. I observed that a change in the C_P distribution has no effect on the H distribution (or any other boundary layer parameters' distributions) upstream of the change. I also concluded that the sharp increase in H near the shock is determined by the pressure gradient across the shock. Furthermore, I observed that in front of the shock, a decrease in C_P causes an increase in H. Behind the shock, however, a decrease in C_P typically results in a decrease in H.

Armed with a qualitative understanding of the relationships between C_P and H, I sought to develop a procedure for modifying an existing C_P distribution to return a desired H distribution. Since H is often used as a separation criterion, I would then be able to construct a C_P distribution with fully attached flow. I first looked to develop a routine for reducing the rise in H due to the shock. By raising the C_P value of control point three ($C_P(3)$), I could decrease the strength of the shock, and thus diminish the rise in H due to shock. Unfortunately, adjusting only control point three would change the area under the C_P distribution, and thus alter the lift coefficient. However, by simultaneously lowering the C_P value of control point two as I raised the C_P value of control point three, I would decrease the shock strength while conserving the area under the C_P curve. Thus the rise in H due to the shock would be lowered without upsetting the lift coefficient. In order to install this procedure into my computer code, I needed an expression to quantitatively predict this increase in $C_P(3)$ requisite for a desired decrease in the maximum value of H within the shock region. After trial-and-error, I selected an expression of the form:

$$\Delta C_P(3) = A*H*\Delta H_{max} + B*H*(\Delta H_{max})^2$$

Since this expression is an approximation, I had to make the procedure iterative within the code. Despite being iterative, the procedure requires little CPU time to achieve a high degree of accuracy. (Figure 2.) It is also worthy to note that this shock-adjusting procedure can be used to increase a shock's strength if desired.

In order to remove any flow separation that occurred after the shock region, I planned to "locally" adjust the C_P value at each point where H exceeded some specified $H_{separate}$. This technique differs from the previous "shock-reducing" method where C_P values upstream of the separation point were modified. Early attempts to predict a local change in C_P that would yield a desired change in H were unsuccessful. Unfortunately, multiple C_P values exist which produce a single desired value of H. My code needed to predict a change in C_P that would not only produce a desired H value, but also preserve the characteristics of the Θ distribution. (While there are various C_P values that will produce a desired H, there exists a unique C_P value that will produce both the desired H *and* Θ values). With continued investigation and trial-and-error, I developed an expression of the form:

$$\Delta C_P = A*(\Delta H/H) + B*(\Delta H^2/H)$$

which accomplished this with reasonable accuracy. (Figure 3.) With procedures for both reducing H within and after the shock region, my code could now redesign an airfoil to avoid separation at

specific free stream conditions. I now aspired to devise a process for reducing an airfoil's viscous drag.

While H predicts flow separation, the value of Θ at the trailing edge ($\Theta_{t.e.}$) of the airfoil predicts viscous drag. In order to minimize viscous drag, one must minimize $\Theta_{t.e.}$ (viscous drag also depends on $H_{t.e.}$, but lowering $\Theta_{t.e.}$ has the greatest influence on the reduction of viscous drag). Thus, I required a procedure for redesigning a C_P distribution around a desired value of $\Theta_{t.e.}$. At the same time, I wanted the basic characteristics of my C_P distributions to remain fixed. In order to accomplish this, I first scaled the existing Θ distribution so that Θ at the trailing edge of the airfoil would equal a user specified target value. Thus, the new target Θ distribution would maintain the same characteristics of the original Θ distribution. I now only needed a procedure for modifying an existing C_P distribution to produce this desired target Θ distribution. I once again chose to use a direct iterative method. After much experimentation, I developed an expression of the form:

$$\Delta C_P = A*(\Delta\Theta/\Theta) + B*(\Delta\Theta^2/\Theta)$$

This expression produced an accurate estimation of the required change in C_P needed to produce a desired change in Θ. After installing this iterative procedure into my code and running some test cases, I was pleased with the results. Designing to a target Θ distribution is much more efficient than designing to a target H distribution: it requires fewer iterations (and therefore less CPU time) to achieve a desired tolerance. Figure 4 shows an example where I have used this new procedure to reduce the drag of an airfoil by nearly five counts.

With separate methods for redesigning a C_P distribution to produce both desired H and Θ distributions, I now sought to create a procedure for redesigning an airfoil to simultaneously produce both target boundary layer distributions. Unfortunately, it is sometimes impossible to design an airfoil to meet two stringent criteria. Therefore, one of the criteria must take precedence over the other. Then, if a C_P cannot be modified to produce both boundary layer design criteria at a given x/c location, the code will insure that at least the primary design constraint is met. With few alterations to my existing code, I installed this procedure. Basically, the code modifies the existing C_P distribution to remove any secondary criterion violations. Once this has been accomplished, the code checks for primary criterion violations. If the code encounters any such violations, it readjusts the C_P distribution accordingly.

One drawback of these various methods for reducing separation and viscous drag is the tendency for the value of C_P at the trailing edge ($C_{P\ t.e.}$) to become increasing smaller. Typically, $C_{P\ t.e.}$ for a given airfoil is fixed around a value of 0.1. I needed a procedure for redesigning an airfoil, with the added constraint of keeping $C_{P\ t.e.}$ constant. Unfortunately, this added constraint created the possibility for even more cases with "no solutions." However, I felt this constraint was too important to over-look. Therefore, I devised a fairly simple subroutine that uses triangulation to adjust the C_P distribution after the shock until this trailing edge constraint is met. (During this stage of my research, I determined that a linear C_P distribution after the shock results in the lowest value of $\Theta_{t.e.}$) I even included the option of maintaining constant lift and moment coefficient by adding or subtracting a parabolic increment to the C_P distribution as needed. Figure 5 illustrates an example of this lift coefficient conserving technique. Readjusting the C_P

distribution to constrain $C_{P\ t.e.}$ unfortunately can create more boundary layer parameter violations (H and/or Θ may once again exceed their specified limits). One would then have to redesign the C_P distribution to eliminate the new violation, which in turn could cause another $C_{P\ t.e.}$ violation. Continuing this iterative procedure will cause eventual convergence on the solution, assuming one exists.

My completed code meets all of the original project goals. It can eliminate flow separation about an airfoil, as well as reduce an airfoil's viscous drag. Despite utilizing direct-iterative procedures, the code remains efficient. However, installation of a proven boundary layer analyzer that uses the *indirect* lag-entrainment method would eliminate the need for most of the iterations. Unfortunately, no such analyzer was available during the course of my research. Regardless, the UNIX based workstations I used for my research provided more than enough processor speed to run my program swiftly.

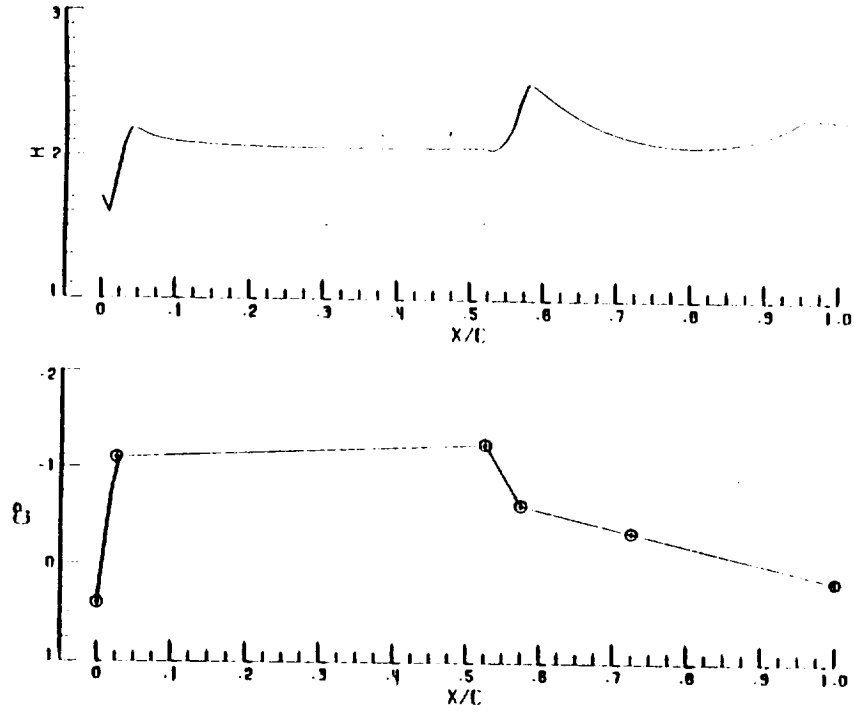

Figure 1.

(In the remaining figures, the dashed-lines represent the original distributions, while the solid-lines represent the adjusted distributions.)

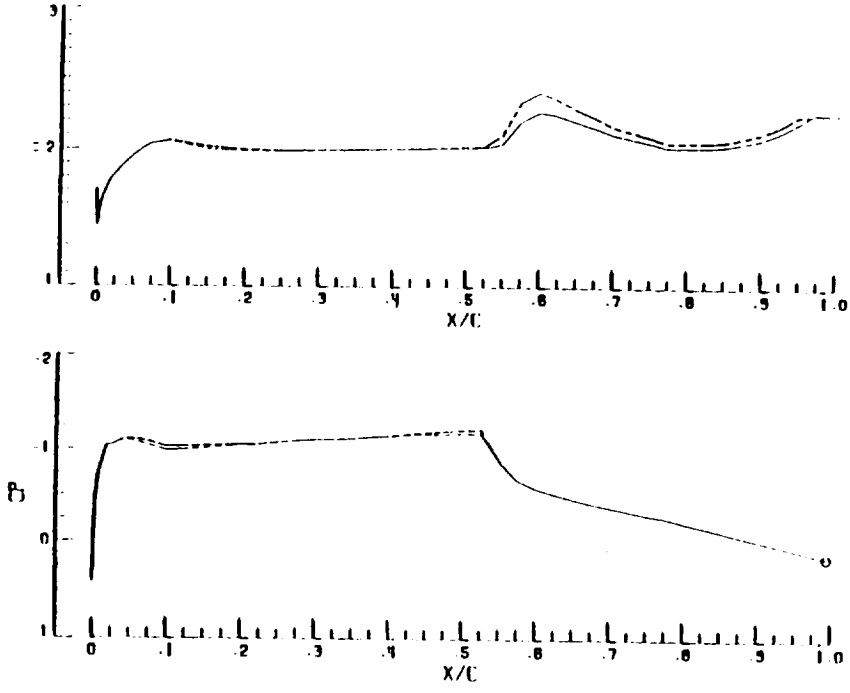

Figure 2.

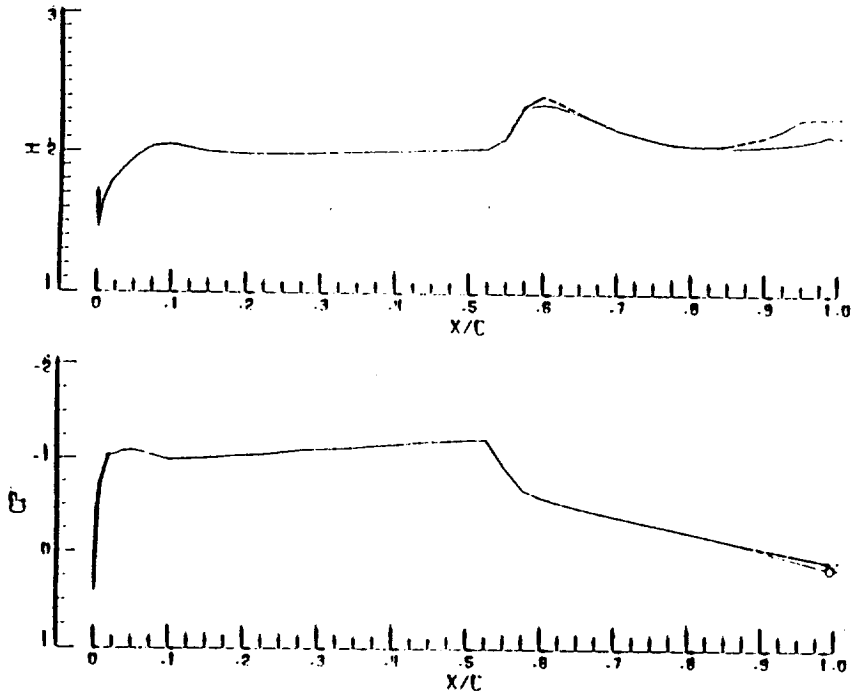

Figure 3.

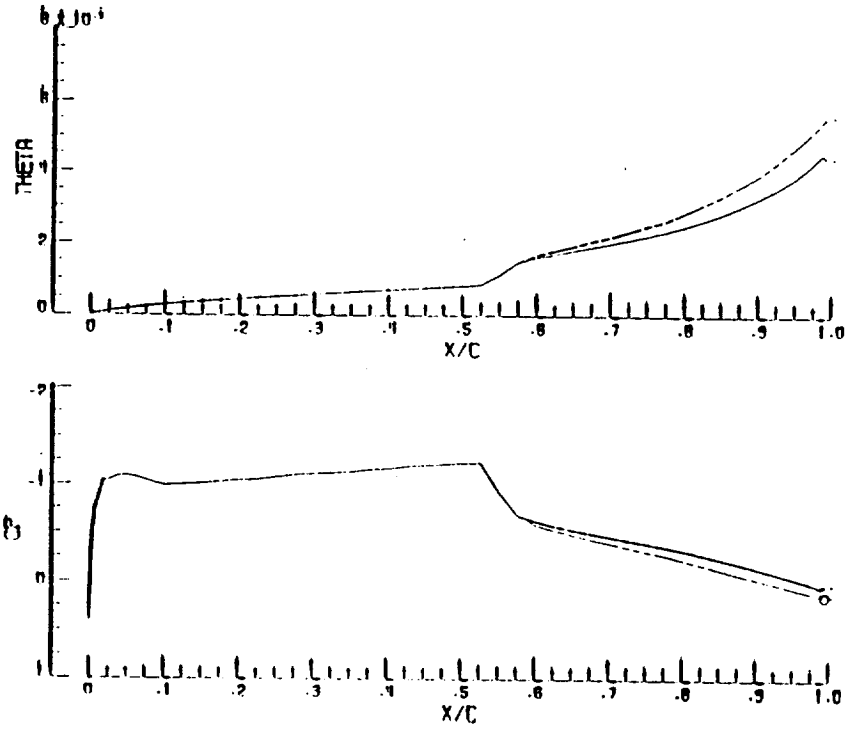

Figure 4.

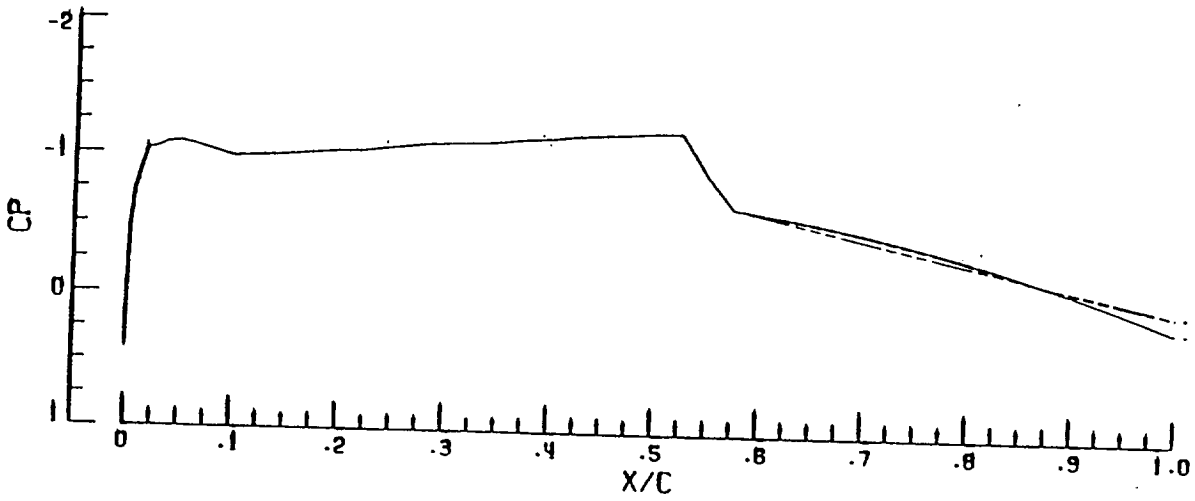

Figure 5.