Supervised Learning Program

Report on

Inverse Airfoil Design Using XFOIL

Submitted by

Micky
140010059

Under The Guidance Of

Prof. A.M. Pradeep

Aerospace Engineering Department
IIT BOMBAY

Abstract

The report describes the Inverse Airfoil Design method. Given a target pressure distribution, an airfoil has to made that gives the required pressure distribution. XFOIL is used as the solver for the flow over the airfoil.

Contents

Abstract ……...2

1. **Introduction ……...3**
2. **Available Methods ……4**
3. **Design Methodology Implemented ……...5**
4. **Interpolation And The Algorithm ……...7**
5. **Results ……..8**
6. **Conclusions ……10**

References ……11

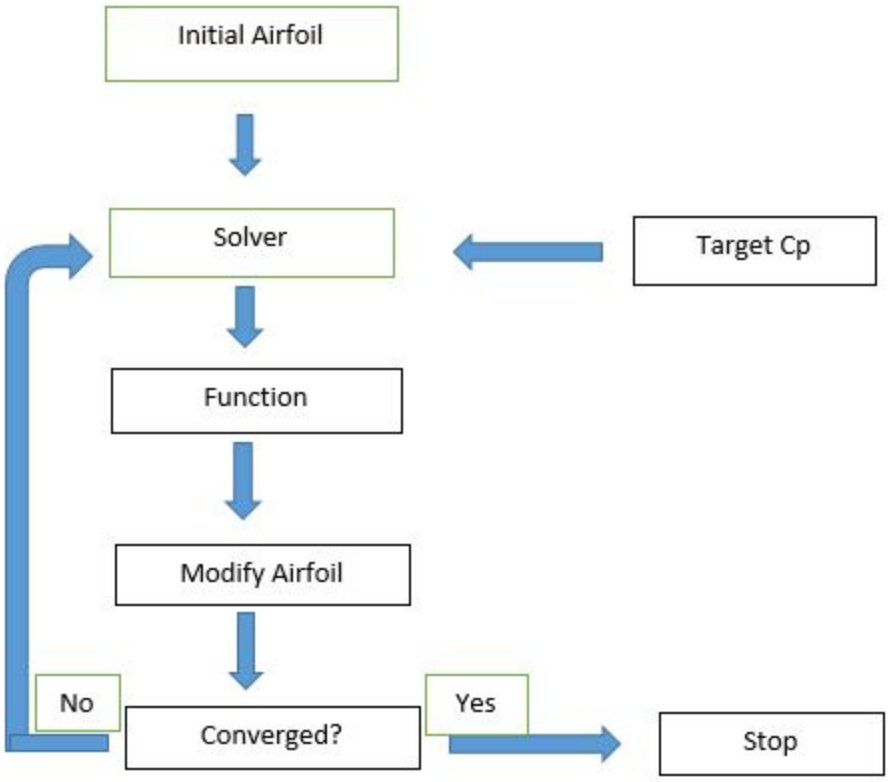
Acknowledgement ……...12

Introduction

In inverse airfoil design method an airfoil has to be generated that satisfies the criterion given by the user. There are various user specified criterion, the one that has been implemented in this report uses a target pressure distribution at a particular mach number over the airfoil. An airfoil has to be generated that matches the specified the pressure distribution at a particular mach number.

Inverse Airfoil Design Methodology

The user gives the target pressure distribution at a particular mach number. Now, the inverse design starts with an initial airfoil, the pressure distribution over the airfoil is calculated using a solver. The obtained pressure distribution is compared with the target pressure distribution. The airfoil coordinates are changed by a function that takes into account the current and the target pressure distribution. A new airfoil is generated and the pressure distribution over the new airfoil is calculated using a solver and the obtained and the current pressure distributions are compared and the airfoil coordinates are changed again. The above process is repeated till the convergence is reached.



Inverse Design Method

Available Methods

The basic methodology of the inverse design is the same as described above but the difference comes in the solver that is used and the function that is used for modifying the airfoil.

Three methods (with different solvers) were explored and are described below :-

1. Using Ansys Fluent

In this method the ansys fluent solver could be used to calculate the cp distribution over the airfoil. This method is a bit complicated as compared to the third method as a script would have to be written which controls the ansys fluent and was not implemented due to the unavailability of time.

2. Using OpenFOAM

This method is the same as the previous method but there is a benefit in using this method,i.e, in order to control OpenFOAM we may not need to write the full script as there is a package named PyFoam through which OpenFOAM could be controlled as we wish but still is was very complicated as to implement the above one needs a good command over PyFoam and that would take a lot of time and thus could not be done.

3. Using XFOIL

This is the most fastest and the least complicated as compared to the above methods and thus was implemented.

The Algorithm(Function) Used

Now the solver has been finalized, now the problem boils down to how to get the required airfoil. The airfoil coordinates needs to be changed so that the pressure distribution approaches the target pressure distribution. To accomplish this various algorithms were explored and the algorithm mentioned in the paper by *Raja Ramamurthy, Benedikt Roidl* and *Wahid Ghaly* on "*A VISCOUS INVERSE DESIGN METHOD FOR INTERNAL AND EXTERNAL FLOW OVER AIRFOILS USING CFD TECHNIQUES*".

Design Methodology Implemented

The solver and the algorithm have been finalized. Now the detailed description of the process would be given.

1. Initial Airfoil

An initial airfoil is required whose coordinates would be changed iteratively to obtain the airfoil satisfying the target pressure distribution. So, an initial airfoil is fed in the script once to start the design process.

2. XFOIL

The flow conditions are specified in the XFOIL and the pressure distribution is generated. The generated(current) pressure distribution is fed into the algorithm.

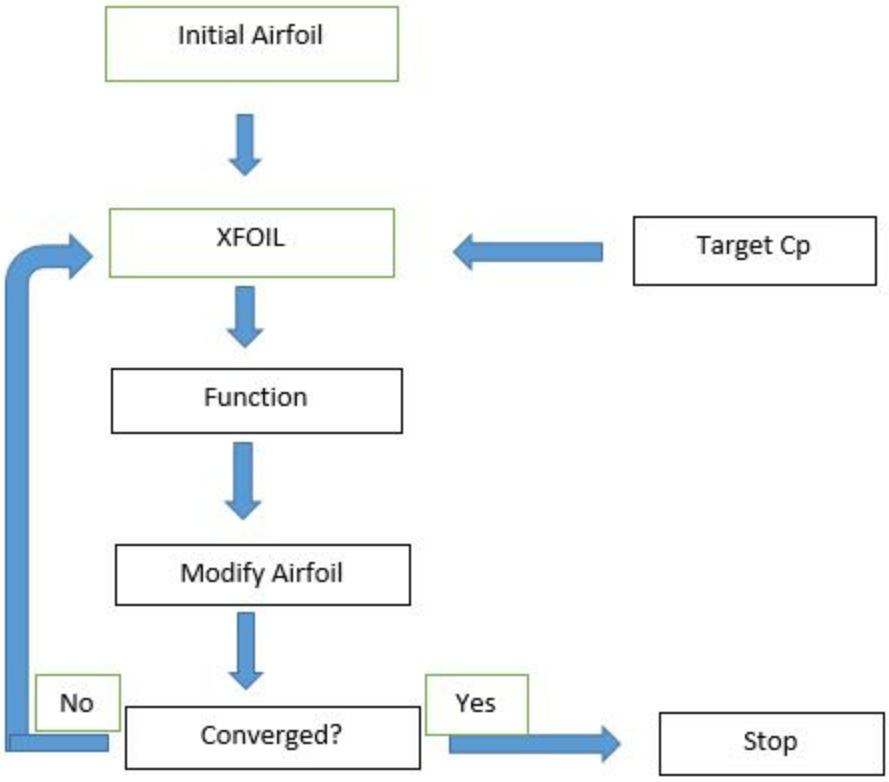
3. Changing The Airfoil

XFOIL feeds the current pressure distribution. The algorithm then compares it with the target pressure distribution and generates velocities for each point on the airfoil. The airfoil coordinates are then moved according to their corresponding velocities for some time interval(user specified).
The new airfoil is generated and is fed to XFOIL

4. Convergence

The L2 norm is calculated using the current and the target pressure distribution. If the L2 norm is less than some(user specified) number, the loop of XFOIL and the algorithm is terminated and hence the required airfoil has been generated.

The algorithm may converge faster for some initial airfoils. As XFOIL can't work well for transonic regimes, so for that regime some other solver needs to be used. The same algorithm can be used for different regimes, it is regime independent. The methodology will remain the same for different regimes but the solver needs to be changed. Usually the high-fidelity simulation softwares like OpenFOAM and Ansys are very huge(a lot of features in them) and writing a script from scratch to put them in the loop along with the algorithm is a bit tedious. So, if someone wants to go for OpenFOAM, using PyFoam would be very beneficial. In the case of XFOIL, the script to run the solver and the algorithm in a loop was written from scratch in Python.



Inverse Design Methodology Adopted

Interpolation And The Algorithm

The x coordinates of the initial airfoil may not be the same as that of the target pressure distribution. So, if that is the case(often this is the case), the x coordinates needs to be made same. So there are two approaches , either change the x coordinates of the airfoil to match with that of the target pressure distribution or change the x coordinates of the target pressure distribution to match with that of the airfoil. The later is adopted in the design methodology. Linear interpolation was used to obtain pressure at the x coordinates of the airfoil.

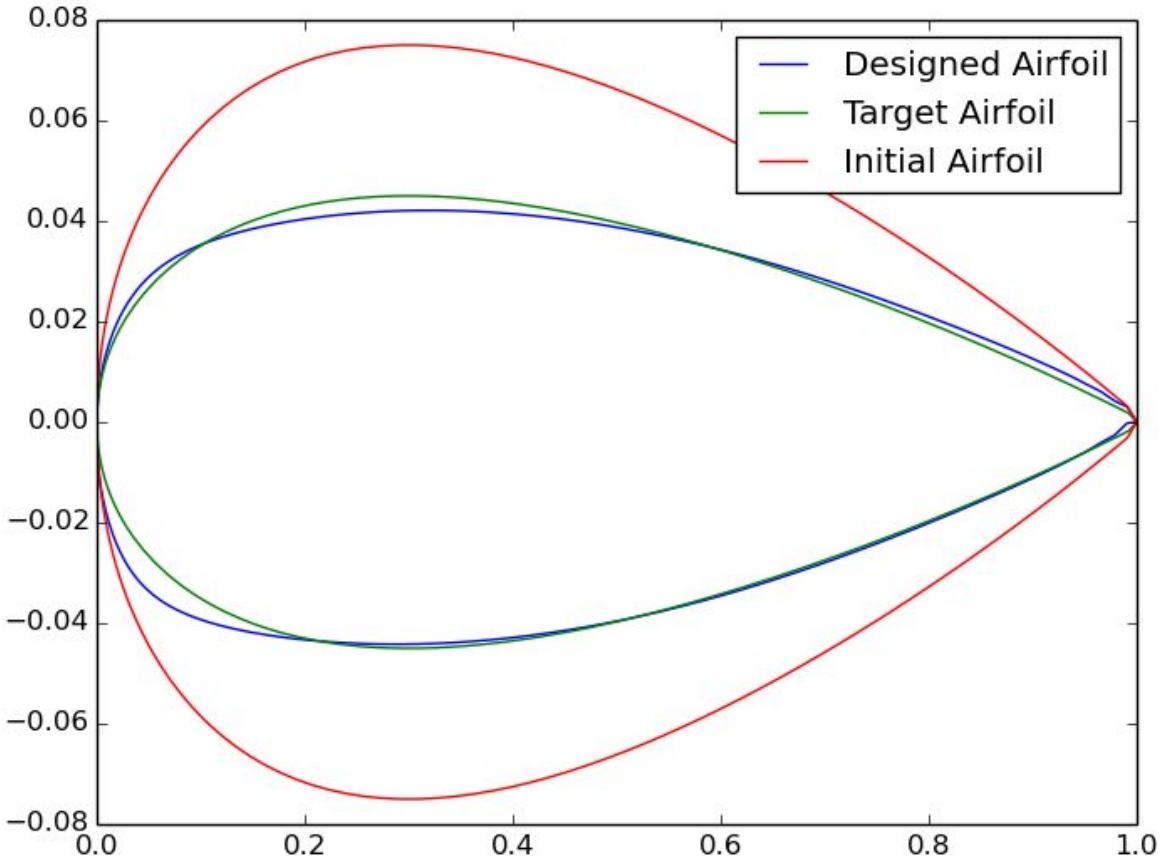
The algorithm that was written is slightly modified from that given in the paper but the physics is the same. The airfoil was just moved in the y direction(only the y coordinates of the airfoil were moved). The algorithm generates a new airfoil but it needs to smoothed out so that the calculations done by the solver gets converged. The smoothing is done to remove the abrupt jumps in the airfoil. In the implemented algorithm averaging was used but in the paper from where the algorithm is taken the smoothing is a bit different and complicated.

The algorithm and the interpolation codes were written in Python.

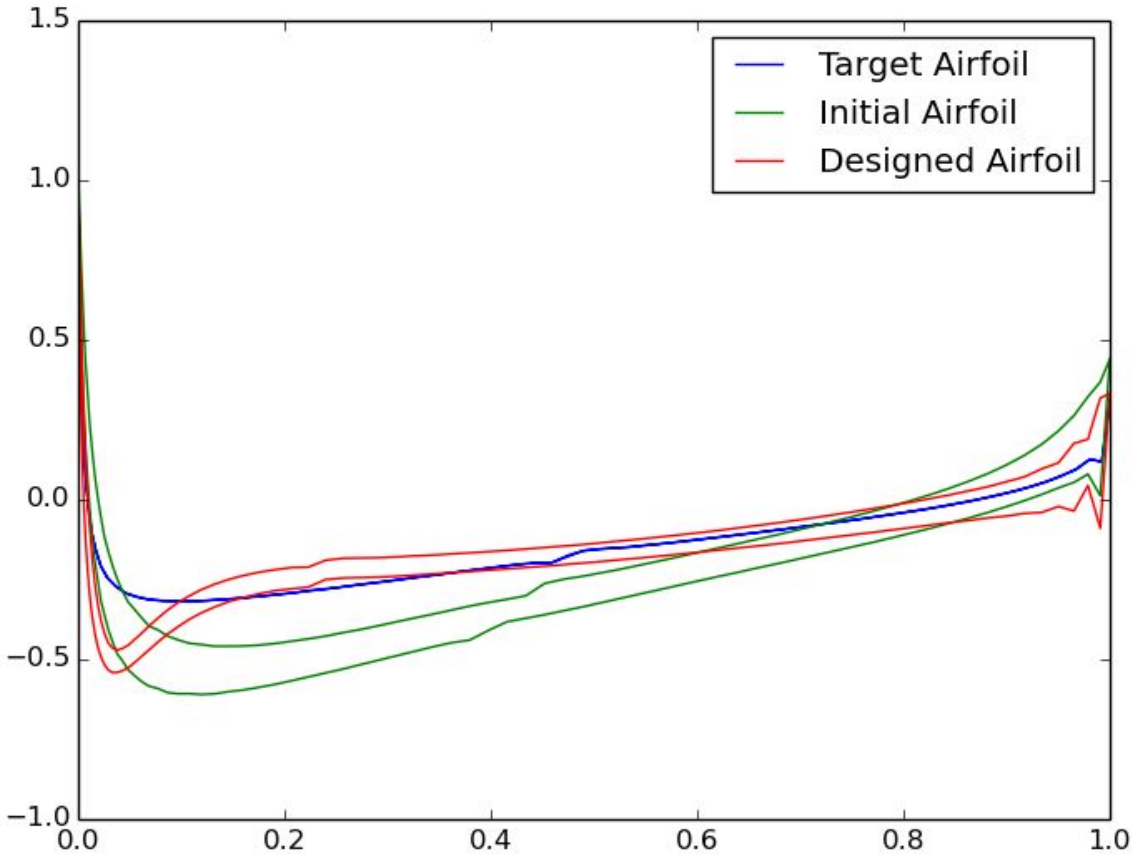
Results

For testing the code the target Cp distribution was taken that of NACA 0009 and the initial airfoil was taken NACA 0015

The following are the results generated:-



Airfoils Compared



Pressure Distributions

Conclusions

The inverse design method described in this report does not work very well if the target airfoil is highly cambered. The method described is still under development and a lot of improvements needs to be done especially in the smoothing part. So, the smoothing part would be improved further and then some other algorithm could be tested. Once that is done high fidelity solvers would be used.

The code is available on the following link:-
Inverse_Airfoil_Design

References

[1] Paper by *Raja Ramamurthy, Benedikt Roidl* and *Wahid Ghaly* on "*A VISCOUS INVERSE DESIGN METHOD FOR INTERNAL AND EXTERNAL FLOW OVER AIRFOILS USING CFD TECHNIQUES*

[2] XFOIL Software by Mark Drela MIT

[3] Ansys Software

[4] OpenFOAM Software

[5] PyFoam Software

Acknowledgement

I would like to express my sincere gratitude to **Prof A.M Pradeep** for his invaluable guidance, support and constant encouragement during the course of the project.His presence at tough times and prompt decisions has constantly truly inspired me in carrying out this work.In addition, grateful acknowledgement to **Utkarsh Chauhan,** my batchmate and my co worker in this project without whom the inverse design method code could not get complete. He has helped me to overcome the setbacks during the tasks. I greatly value his support.