Indian Institute of Technology, Madras

Design of Low Speed Aerofoils using Evolutionary Optimization Algorithms

Project Report submitted as a requirement for Summer Research Fellowship Programme 2012
21st May – 15th July, 2012

Written and Compiled By

Prashanth K R

Student

B.Tech (Third Year Completed)

Mechanical Engineering,

VIT University, Vellore

Guided By

Dr. Arul Prakash K.

Assistant Professor

Fluid Mechanics Group,

Dept. of Applied Mechanics,

IIT Madras

Abstract

The work I would like to propose involves development of an **evolutionary optimization algorithm** based technique that can help design otherwise tricky components such as aero foil blades.

For instance, in case of components such as turbine or propeller blades, I propose the development and use of a coding system that can uniquely define various sections of the aero foil (such as hub, mean, tip and intermediate ones) using a finite number of parameters. A fast flow solver would serve as a cost function. This would be developed using fundamental principles of the physical system. It would then evaluate its performance and serve the data into an optimization routine by calculating the values of the mechanical/aerospace objective function (in this case based on lift, drag, pressure distribution, etc.) that we are trying to optimize.

An evolutionary algorithm, such as a **Real Coded Genetic Algorithm**, could then be used to optimize existing designs through numerous iterations of crossovers and mutations. This process would be carried out for various sections of the blade to ultimately create a complete lofted 3D geometry. There is a lot of potential to make blades for various purposes including wind turbines, impellers and even aircraft wings.

Contents

Acknowledgements

First and foremost, I would like to convey my gratitude towards the **Dept. of Applied Mechanics**, **IIT Madras** for considering me for the **Summer Research Fellowship Programme 2012**. The time spent here has been one of the more fruitful experiences of my life. I would always be grateful for this opportunity.

I would like to thank my guide **Prof. Arul Prakash** of the **Fluid Mechanics Division** for giving me freedom to work on my own problem and guiding me throughout. His encouragement and support were essential for the completion of the project. He also connected me to other students whose assistance I could take to understand certain aspects of the project.

I would also like to thank **Vasanth**, a research scholar in the dept. who helped me a lot with the understanding of the working of a grid-less flow solver.

Lastly, I would like to thank everyone around me in lab and in the hostel who played a role in making my stay here enjoyable.

Problem Introduction

Aerofoil design has always been one of the more challenging problems in aerospace engineering. The phenomenon is highly non-linear and results often become non-intuitive. It is very hard to design aerofoils or for the matter hydrofoils that are meant to be operated at specific flow conditions. Optimal performance may demand that the aerofoil to generate maximum lift with minimum drag. Another way of saying this is maximize L/D ratio with some bounds on the pitching moment. This is a simplified **single objective function optimization problem**.

In other cases, if there is a particular desired amount of lift required, we need to minimize the square of the difference between the target lift and the lift in our aerofoil such that it becomes nearly zero. The same can be said about the pressure distribution. Simultaneously, the drag would have to be minimized independently, giving scope for **multi-objective function optimization** problems. In all problems, even the **aerodynamic moment** must be used in defining the objective function as we don't want very large values of the same as it puts torsional load at the point of attachment. If used in aircraft, it affects equilibrium.

Multi-objective problems may also be of the form where we need the foil to attain certain values of lift and drag for particular angles of attacks. This process introduces new complications into the problem where there is no single unique solution. On the other hand, some problems may force us to introduce pareto-optimality, where we need to compromise on some criterion in order to satisfy some other.

Lift, drag and aerodynamic moments **depend heavily on the shape** of aerofoil. When we take into account **viscous effects** as well, it becomes a daunting task to come up with the right shape of an aerofoil for a specific application.

In the past, engineers tackled this problem through a compromise. Thousands of 2D slices of aerofoils were catalogued into a vast database and organized into various families based on numerous traits such as flow regime, etc. The engineer would then choose the aerofoil based on how closely it matched the experimental characteristics of his application.

With the advent of affordable, powerful and easy to use computing tools, this can now take a whole new approach. This project aims to integrate an analysis as well as an optimization routine that can modify a set of randomly generated aerofoils till it matches the design criteria. Any design criterion can be specified as a function of the pressure distribution, and an evolutionary optimization algorithm will perform the required shape optimization so that its performance tends to the desired criteria.

Literature Review

1. **Design and Analysis of Low Reynolds Number Airfoils**, by *Nicholas K. Borer* from Georgia Tech provided brief but informative section on the conventional approaches to aerofoil design and the primary design criteria engineers use in the process. The report also contains an introduction to vortex panel methods.
2. **NACA Four-Digit Airfoil Section Generation using Cubic Parametric Curve Segments and The Golden Section**, by *William T. Scarbrough* from Rochester Institute of Technology gave detailed information on generating ordinates for a NACA aerofoil. This helped me greatly with the parameterization and geometry generation.
3. **NACA Airfoil Evaluation**, by *Evan Kontras et al* from University of Missouri evaluated the NACA 4212 aerofoil using both analytical and numerical methods. This gave me data to validate my panel based solver before using it for optimization.
4. **Algorithm for Calculating Coordinates of Cambered NACA Airfoils at Specified Chord Locations**, by *Ralph L. Carmichael* from Public Domain Aeronautical Software gave detailed mathematical outline on the process of writing code that generates NACA aerofoil ordinates. The mathematical basis of my code is from this paper.
5. **Panel Methods: An Introduction**, by *Larry Erickson* from NASA Ames Research Center gave a holistic view of panel methods and its capabilities as well as its limitations. A clear explanation of the distinctions between various panel codes was very helpful in choosing the approach while coding the solver.
6. **Parameterization of Airfoils and its Application in Aerodynamic Optimization**, by *J. Hajek* of Charles University gave a qualitative review of various approaches to optimizing aerofoils. The author used adopted a popular open source solver X-FOIL with BEZIER-Parsec parameterization to accomplish the same.
7. **Automatic 2D Airfoil Generation, Evaluation and Optimisation using MATLAB and XFOIL**, by *Xavier Mauclère* of Technical University of Denmark implemented a program to generate wind turbine aerofoil sections using multi-objective optimization.
8. **Shape Optimization of Low Speed Airfoils using MATLAB and Automatic Differentiation**, by *Christian Wauquiez* from Royal Institute of Technology had pretty much the same goals as this project. Interesting insights on posing an optimization problem were obtained from this paper.
9. **Bezier-PARSEC: An Optimized Aerofoil Parameterization for Design**, by *R.W. Derksen & Tim Rogalsky* from Winnipeg, Canada discusses the influence of the choice of parameterization on the time it takes for the solution to converge. Potential for this new approach as a parameterization tool in optimization is explored qualitatively in this article.
10. **Developments in the use of the Genetic Algorithm in Engineering Design**, by *G. N. Bullock et al* from the University of Plymouth discusses about how the use of Genetic Algorithms has cut across various disciplines and has tremendous potential in helping us with detailed designs of complex components.
11. **Airfoil Aerodynamics Using Panel Methods**, by *Richard L. Fearn* implements a type of Panel Method using the package Mathematica. The article discusses the thought behind each step of the process and the test results were used for solver validation.

Methodology

The project is entirely software programming based where I have divided my code into **three independent tasks**, namely:

1. **Aerofoil Parameterization**
2. **Flow Solver**
3. **Optimization Routine**

The purpose of each of these tasks is as follows:

1. Aerofoil Parameterization

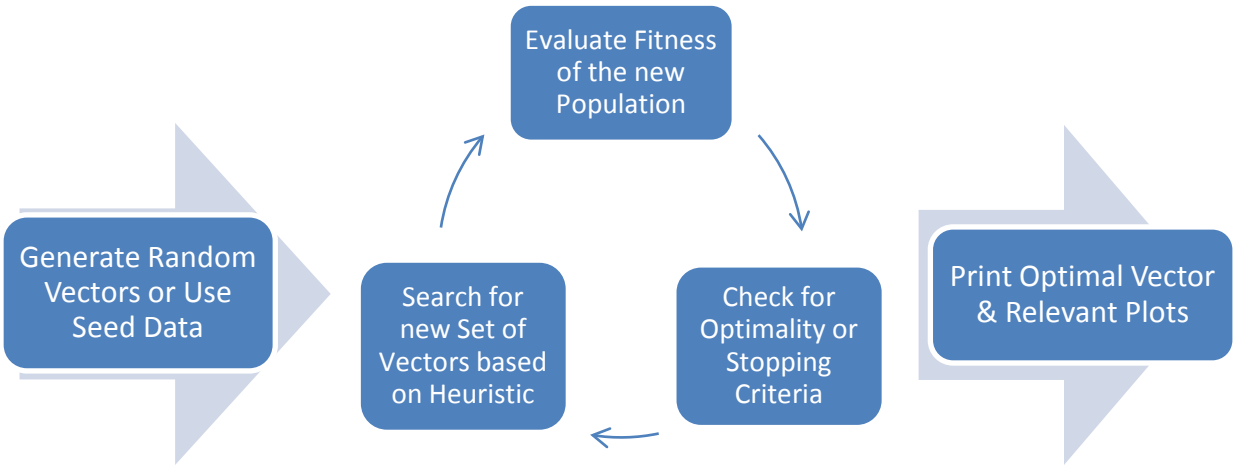
Parameterization in simple terms is a process where we **uniquely define a particular aerofoil in terms of a finite number of independent variables**. Our code should be able to generate all the required geometrical details of the aerofoil simply with the help of those numbers which we can refer to as a tuple or a vector. We are **converting the problem of finding the most optimal aerofoil geometry to the most optimal combination of those numbers** (in other words, the most optimal vector) for our desired application.

2. Flow Solver

The process of shape optimization is an iterative process where the '**fitness**' of an aerofoil vector (as previously defined) is computed and appropriate modifications are performed till an end criterion is met. Here fitness is the objective function we choose to optimize (or extremize). It could be Lift, Drag or their ratio, or even deviation from the required characteristics. Within a single iteration, one may have to evaluate numerous aerofoil vectors. Hence, we need a **generic** (applicable to all feasible aerofoils), **fast** (computation cost/time must be minimum) and **scalable** (geometrical approximations can be controlled) solver in order to accomplish the task.

3. Optimization Routine

This is the process where **the feasible space of the aerofoil vector is searched** till an appropriate candidate is found, or time runs out. A number of **Global Optimization Algorithms** can be used for this problem. Being a highly non-linear phenomenon, there is no intuitive method to obtain the right shape. In general, a global search based optimization routine works in the following manner:



Parameterization Methods

It is completely in the choice of the user to create a parametric representation of an aerofoil. The parameterization technique must be able to fit the shapes of a wide range of aerofoils. For it to work well with commonly used Global Search Techniques, it should be an n-vector. The other function of this part of the code is to generate geometrical input for the solver. However, it is necessary to make sure that the task of calculating the geometric data from the given vector should be computationally cheap. Some of the parameterization techniques developed are:

1. **Bezier-PARSEC Method**

From the source:
"A Bezier parameterization is determined by its **control points** which are **physical points** in the plane. Four curves form the airfoil, two for the camber line and two for the thickness distribution. The first and last control points of a Bezier curve are the initial and terminal point on the curve itself. However the other control points need not be on the curve even though they determine the shape of the curve. As such a Bezier parameterization of an airfoil has control points that are only indirectly determined by the underlying aerodynamics.
The superior performance of PARSEC parameterization is likely due to its **ability to minimize epistasis** – a term borrowed from biology to indicate the nonlinear manner in which the objective function is dependent on the design parameters. In epistatic functions, small changes in several variables can result in large changes in the objective function. They are difficult to minimize because they provide so few clues as to the location of the global minimum. Typically a reduction in nonlinear interaction of the parameters, by having **parameters more directly linked to the objective function**, will enable the optimizer to converge more quickly."

With the help of 3/4 degree parametric polynomial curves, we can uniquely define the thickness and the camber profile of the foil. There are two different implementations of this approach which go by the name of BP3333 & BP3434 (5 more parameters introduced).

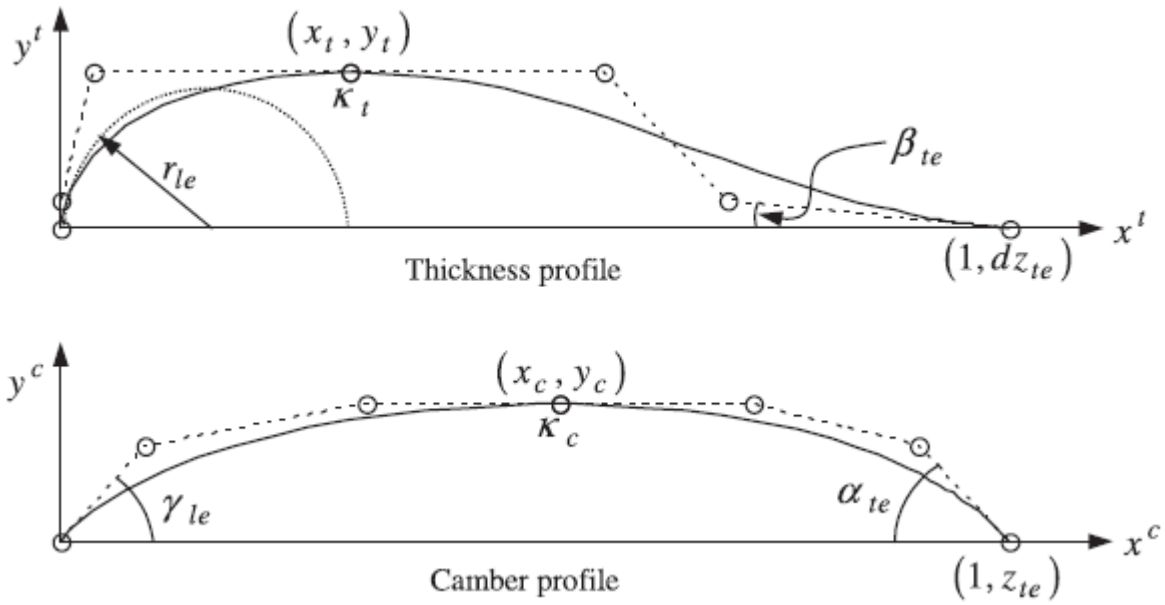


Fig.1. BP 3333 Aerofoil Geometry and Bezier Control Points Defined By Twelve Basic Aerodynamic Parameters

2. NACA 4-Digit Aerofoil Parameterization

The NACA aerofoils are aerofoil shapes for aircraft wings developed by the **National Advisory Committee for Aeronautics** (NACA). The shape of the NACA aerofoils is described using a series of digits following the word "NACA." The parameters in the numerical code can be entered into equations to precisely generate the cross-section of the aerofoil and calculate its properties.

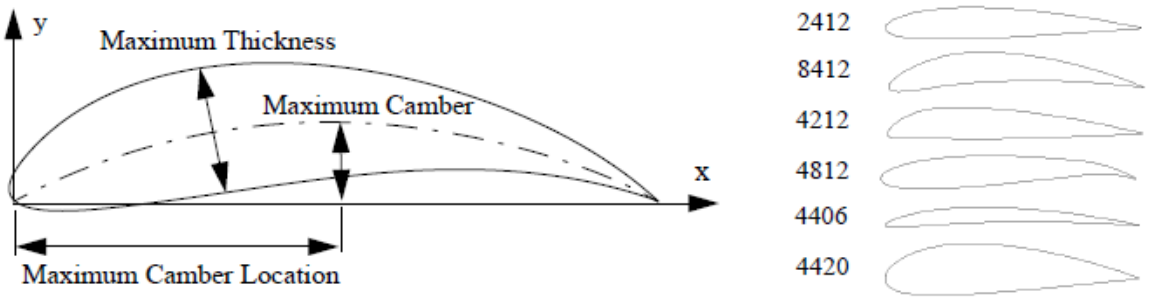


Fig. 2. NACA Parameterization by Example

The NACA four-digit wing sections define the profile by:

- First digit describing **maximum camber** as percentage of the chord. (m)
- Second digit describing the distance or **position of maximum camber** from the aerofoil leading edge in tens of percentage of the chord. (p)
- Last two digits describing **maximum thickness** of the aerofoil as percentage of the chord. (t)

For example, the NACA 4412 aerofoil has a maximum camber of 4% located 40% (0.4 chords) from the leading edge with a maximum thickness of 12% of the chord. Four-digit series aerofoils by default have maximum thickness at 30% of the chord (0.3 chords) from the leading edge.

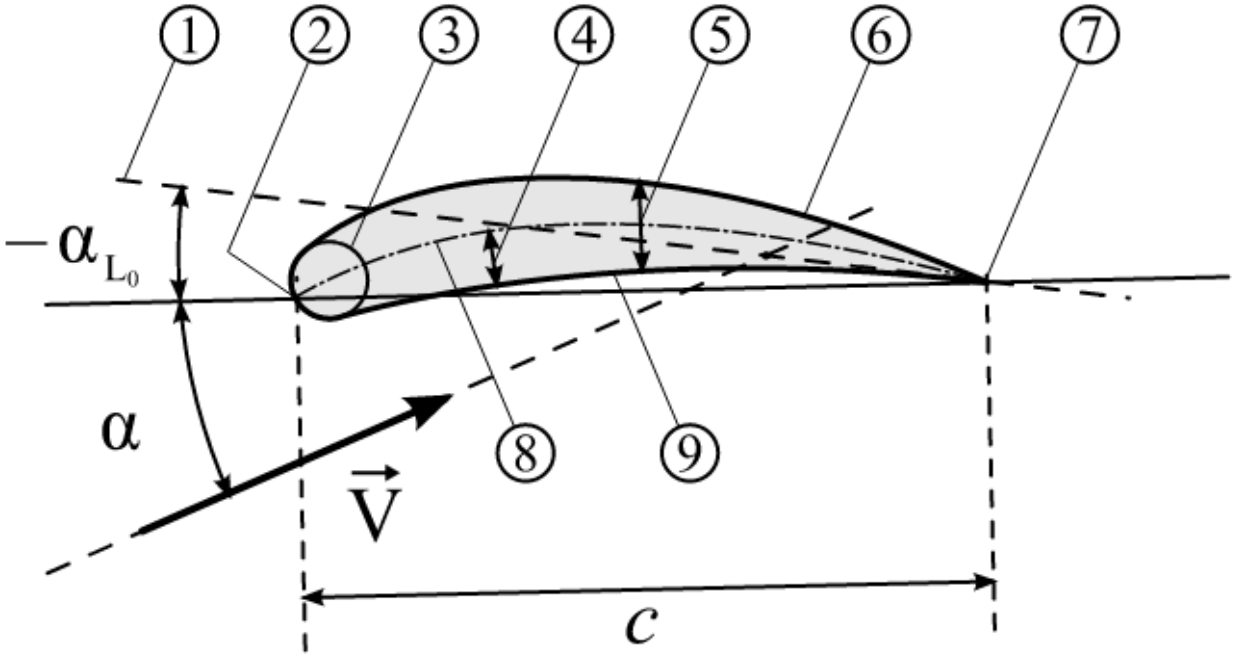


Fig. 3. Profile Geometry – 1: Zero Lift Line; 2: Leading Edge; 3: Nose Circle; 4: Camber; 5: Max. Thickness; 6: Upper Surface; 7: Trailing Edge; 8: Camber Mean-Line; 9: Lower Surface

The equations describing the thickness and camber are as follows:

- **Equation for Thickness** or Deviation from Camber Line

$$y_t = \frac{t}{0.2}c\left[0.2969\sqrt{\frac{x}{c}} - 0.1260\left(\frac{x}{c}\right) - 0.3516\left(\frac{x}{c}\right)^2 + 0.2843\left(\frac{x}{c}\right)^3 - 0.1015\left(\frac{x}{c}\right)^4\right]$$

Here, the last coefficient can be changed to -0.1036 to ensure that thickness is zero at the trailing edge i.e. when x=c. Also, c stands for **chord length**.

- **Equations for Camber Curve**

$$y_c = \begin{cases} m\,\dfrac{x}{p^2}\left(2p - \dfrac{x}{c}\right), & 0 \le x \le pc \\ m\,\dfrac{c-x}{(1-p)^2}\left(1 + \dfrac{x}{c} - 2p\right), & pc \le x \le c \end{cases}$$

The above is a piecewise continuous as well as differentiable polynomial that describes the camber of the aerofoil. For symmetric aerofoils, the x-axis itself is the camber.

- **Calculations of Aerofoil Surface Coordinates**

$$x_U = x - y_t \sin\theta, \qquad y_U = y_c + y_t \cos\theta,$$
$$x_L = x + y_t \sin\theta, \qquad y_L = y_c - y_t \cos\theta,$$

where

$$\theta = \arctan\left(\frac{dy_c}{dx}\right)$$

3. **Other Alternative Parameterization Techniques**

Besides these, engineers have come up with more parameterization techniques that can define an even wider variety of aerofoils. One might also come up with one that can generalize for any kind of a foil, or an impeller blade. Some other alternatives may include NACA 5 and 6 digit aerofoils. In addition, NACA has also introduced 1,6,7,8 Series of aerofoils for specific applications. Series 8 aerofoils are only for supercritical applications.

People have also used **Grid Parameterization** which is nothing but a dense sequence of two-dimensional (x – axis and z – axis) coordinates of **sample points along the aerofoil surface**. In principle, it is possible to use the **coordinates of these points directly as design variables** for aerofoil optimization. In this manner, a very flexible parameterization is obtained; however, great care must be taken to preserve smoothness and reasonable shape of the aerofoil. Moreover, the number of **design variables in this scheme is very high**, which makes the optimization problem quite difficult.

PARSEC Parameterization includes the variables as defined in the figure below. There are many variations of the same.

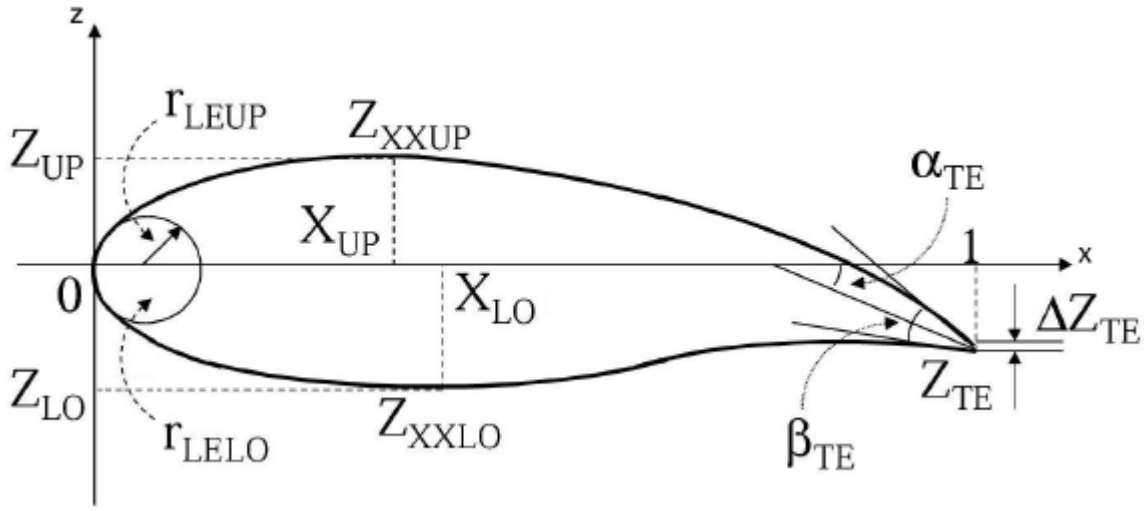


Fig. 4. Design Variables for PARSEC Shape Functions

Depending on various constrains one might have for their specific problems; it is possible to define a specific parameterization scheme for some particular class of foils. Example below shows a scheme for designing foils with a flat lower surface.

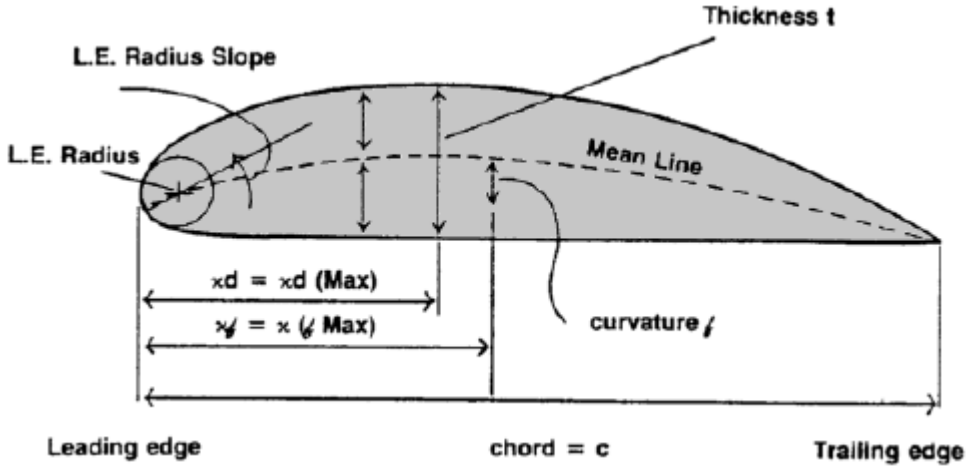


Fig. 5. Alternative Design Variables

Grid-less Flow Solvers

Flow solvers based on numerical methods such as **finite difference** or **finite volume** methods require a grid to be generated in outside the aerofoil geometry. The algorithms calculate the all the field properties by solving simplified versions of the **Navier-Stokes Equations** at the grid points. The reasons why 'field' methods **may not be suitable** for this problem is because:

- They are **computationally expensive**.
- Each run required the **construction of a new unique mesh**, where a number of considerations such as element size and grid spacing.
- The approach arrives at the solution through an **iterative approach**. The flow properties are updated with run and hence it takes a **sizable time to achieve convergence**.
- **Automating** this process, although not impossible is a **very tedious task**.
- Meshing becomes difficult to code as the **configurations get complex** like in cases of multi-element wings.

For this reason, since the 1960s, the aerospace industry has been depending on grid-less solvers. They are much faster and not iterative in process. The accuracy purely depends on how finely the test surface has been divided. The grid less flow solver I would be using in this project would be variation of the popular panel method.

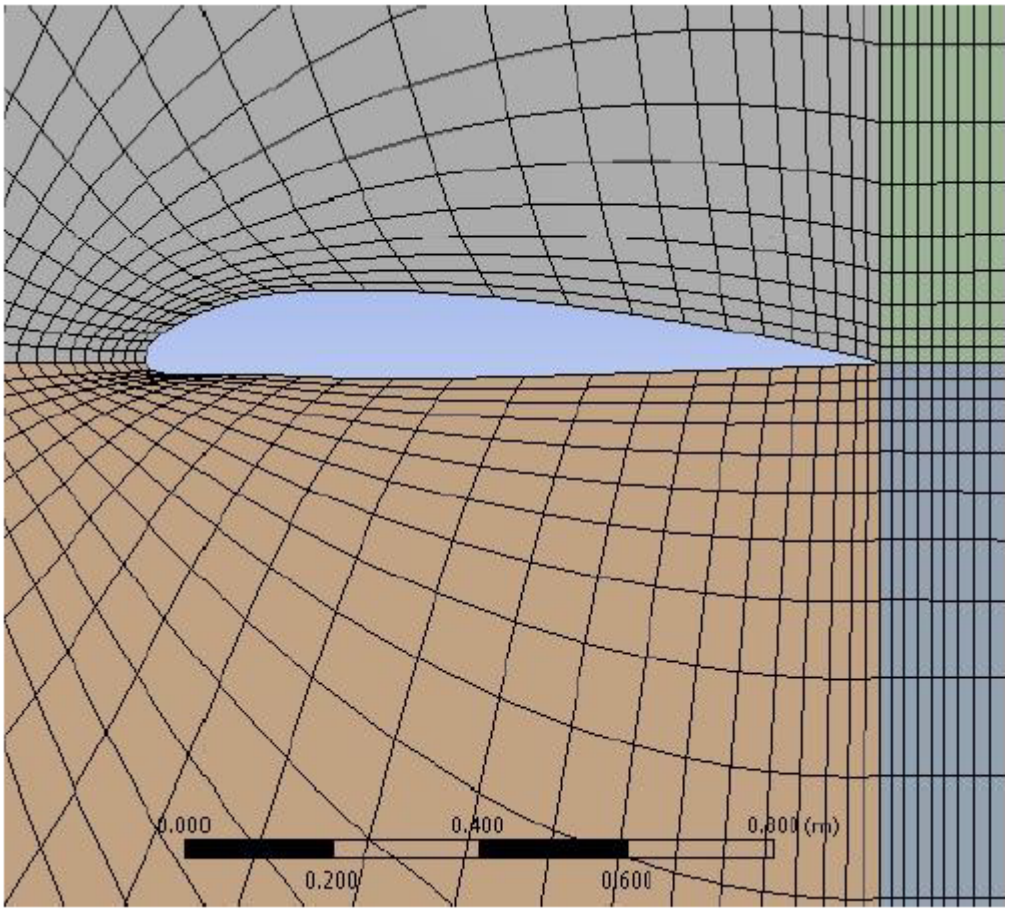


Fig. 6. Mesh Generated in a 'Field' Method

Panel Methods

These are grid-less methods of solving flow past an aerofoil where we approximate the curved surface of an aerofoil as an assembly of connected panels. For simplicity, the method applied to this problem uses only flat panels. On each flat panel in 2D, the ends of the panels are called **corner points** and in the centre of the panel, we have a control or **collocation point**.

The main **assumptions** of all panel methods are **incompressible**, **irrotational flow**, often called potential flow. These two **assumptions greatly simplify** the governing equations of fluid motion.

These methods work by stating that the governing flow equations can be solved via a superposition of elementary flows. These elementary flows can be **sources**, **sinks**, **doublets**, **vortices**, and others. The elementary flows are placed at control points along the perimeter of a body, typically at the centre of a straight segment. Lifting bodies can only be approximated with vortex flows, while non-lifting (symmetrical) flow can be approximated with sources and sinks, among others. The vortex allows for the creation of circulation, an abstraction quite necessary for lifting flow fields.

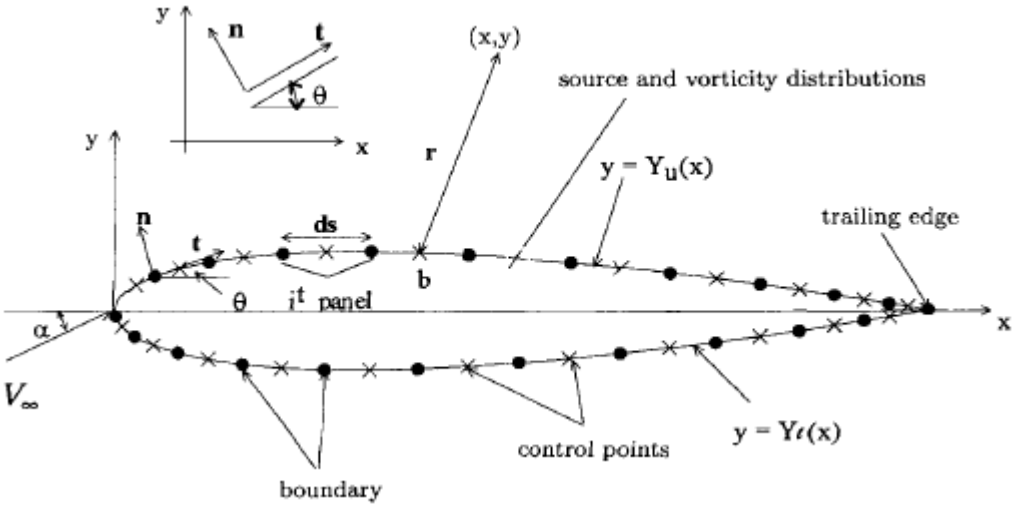


Fig. 7. Panel Representation for an Aerofoil

Panel methods basically attempt to solve the **Laplace's Equation**, which is one of the fundamental linear partial differential equations. The **basic solution procedure** for panel methods consists of discretizing the surface of the object with flat panels and **selecting singularities to be distributed over the panels** in a specified manner, but with **unknown singularity-strength parameters**. Since **each singularity is a solution to Laplace's equation**, a linear combination of the singular solutions is also a solution. The tangent-flow boundary condition is required to be satisfied at a discrete number of points called collocation points. This process leads to a system of linear algebraic equations to be solved for the unknown singularity-strength parameters. The **procedures vary depending on the singularities** used and other details of problem formulation, but the end result is always a system of linear algebraic equations to be solved for the unknown singularity-strength parameters.

In this problem, I use a particular panel method variation called the Hess-Smith Panel Method.

Hess-Smith Panel Method

In this method, each panel (j) is a straight line segment between surface contour points (j and j+1). Along the panel, a **source distribution of constant strength** (σ) is applied. This **distribution strength varies from panel to panel**. As well, along each panel is a **constant vorticity distribution** (γ). The **vorticity is the same** on each panel around the contour and produces the required circulation for the lifting section.

The spacing between the corner points is decided using **cosine** or **half-cosine spacing**.

The strengths of these distributions is unknown. Hence, if there are *n* panels, there would *n* different values for σ, one for each panel. Also, there would be one value for the vorticity distribution that is common for all the panels. Totally, we have *n+1* unknown quantities.

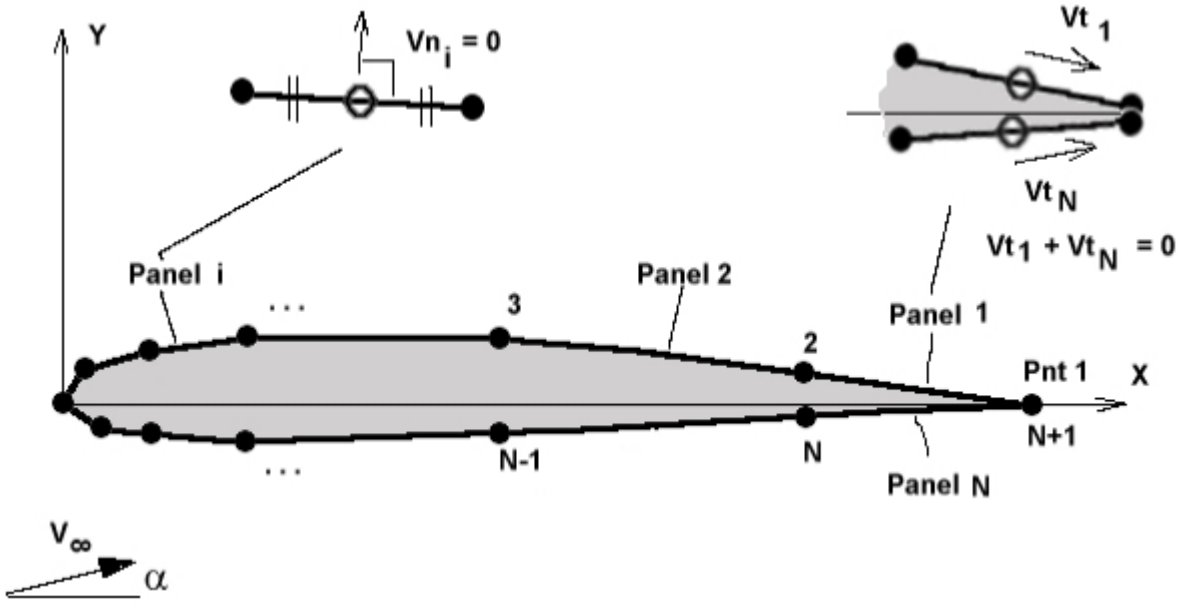


Fig. 8. Hess-Smith Panel Method Graphical Summary

For fully streamlined or attached flow, we put the boundary condition of no flow through surface at the centre of each panel. Essentially, we make the normal component of the velocity zero at the centre of each panel.

Velocity around the panels is influenced by the source and vortex distribution existing on each of the panels and also the **free stream** with an **angle of attack**. The extent to which the distribution strength located on each panel influences the velocities in others is determined by a quantity known as the **influence coefficient**. There would be n+1 influence coefficients for each panel.

For *n* panels, we obtain *n* equations of the following form:

$$V_n = \begin{pmatrix} \sum \text{velocities induced at P due to all source panels} \\ +\sum \text{velocities induced at P due to all vortex panels} \\ + \quad \text{freestream velocity component} \end{pmatrix} = 0$$

$$V_n = \sum_{j=1}^{N} A_{ij} \cdot \sigma_j + \sum_{j=1}^{N} B_{ij} \cdot \gamma + C_i \cdot V_\infty = 0 \quad \text{at Panel (i)}$$

These coefficients are evaluated only using the geometrical data obtained from the discretization of the aerofoil surfaces using the following expressions:

$$A_{ij}^{n} = \begin{cases} \frac{1}{2\pi}\left[\sin(\theta_i - \theta_j)\ln\frac{r_{i,j+1}}{r_{i,j}} + \cos(\theta_i - \theta_j)\beta_{ij}\right] & i \neq j \\ \frac{1}{2} & i = j \end{cases}$$

$$A_{ij}^{t} = \begin{cases} \frac{1}{2\pi}\left[\sin(\theta_i - \theta_j)\beta_{ij} - \cos(\theta_i - \theta_j)\ln\frac{r_{i,j+1}}{r_{i,j}}\right] & i \neq j \\ 0 & i = j \end{cases}$$

$$B_{ij}^{n} = -A_{ij}^{t} \quad B_{ij}^{t} = A_{ij}^{n}$$

Here, n and t superscripts represent normal and tangent components respectively. The A coefficients represent the influence due to a Source Strength Distribution and B coefficients represent the influence due to a Vortex Strength Distribution where,

$$r_{i,j+1} = \left[(x_{m_i} - x_{j+1})^2 + (y_{m_i} - y_{j+1})^2\right]^{1/2}$$

$$r_{i,j} = \left[(x_{m_i} - x_j)^2 + (y_{m_i} - y_j)^2\right]^{1/2}$$

$$x_{m_i} = \frac{1}{2}(x_i + x_{i+1}), \quad y_{m_i} = \frac{1}{2}(y_i + y_{i+1})$$

$$\theta_i = \tan^{-1}\left(\frac{y_{i+1} - y_i}{x_{i+1} - x_i}\right), \quad \theta_j = \tan^{-1}\left(\frac{y_{j+1} - y_j}{x_{j+1} - x_j}\right)$$

$$\beta_{ij} = \tan^{-1}\left(\frac{y_{m_i} - y_{j+1}}{x_{m_i} - x_{j+1}}\right) - \tan^{-1}\left(\frac{y_{m_i} - y_j}{x_{m_i} - x_j}\right)$$

This gives *n* equations for *n+1* unknown terms. We make another equation using the aerofoil's **Kutta Condition**. By this condition, the panels sharing the sharp trailing edge of the foil must have the same tangential velocity component. We calculate the difference between the two using the tangential influence coefficients and thus form the equation.

The free stream influence is nothing but the component (projection) of the free stream velocity in the direction of the normal of the respective panel.

Finally, the set of *n+1* equations can be represented in the matrix form by the following manner:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} & \sum_{j-1}^{N} B_{1j} \\ a_{21} & a_{22} & \cdots & a_{2N} & \sum_{j-1}^{N} B_{2j} \\ \cdots & & & \cdots & \\ a_{N1} & a_{N2} & \cdots & a_{NN} & \sum_{j-1}^{N} B_{Nj} \\ k_1 & k_2 & \cdots & k_N & k_{N+1} \end{bmatrix} \cdot \begin{Bmatrix} \sigma_1 \\ \sigma_2 \\ \cdots \\ \sigma_N \\ \gamma \end{Bmatrix} = \begin{Bmatrix} C_1 \\ C_2 \\ \cdots \\ C_N \\ C_k \end{Bmatrix} \cdot (-V_\infty)$$

This system of equation can simply be solved by inverting the matrix on the LHS and multiplying it with the vector on the RHS.

The next step of the process is to find the **tangential velocity component** magnitude on each of the panels. Using the values of the source and vortex strengths, we can evaluate the tangential velocities using the tangential influence coefficients. Using these values of V_T, we then obtain the value of the coefficient of pressure at the centre of each panel using the formula:

$$C_p = 1 - \left(\frac{V^t}{V_\infty}\right)^2$$

These contours can be plotted for **solver validation**.

Finally, the **dimensionless pressure** in the appropriate directions is integrated to compute the aerodynamic force and the coefficients for **lift** (C_L) and **pitching moment** (C_M) about the leading edge of the aerofoil. Similarly, one can also obtain the value of the **pressure drag** coefficient (C_L) as well.

A simple set of functions were coded in MATLAB to implement the same routine.

Solver Validation

The results of the solver were matched with those from literature and a good amount of agreement was found. Dimensionless Pressure distribution from exact conformal mapping and panel method from literature was matched with that from the coded solver.

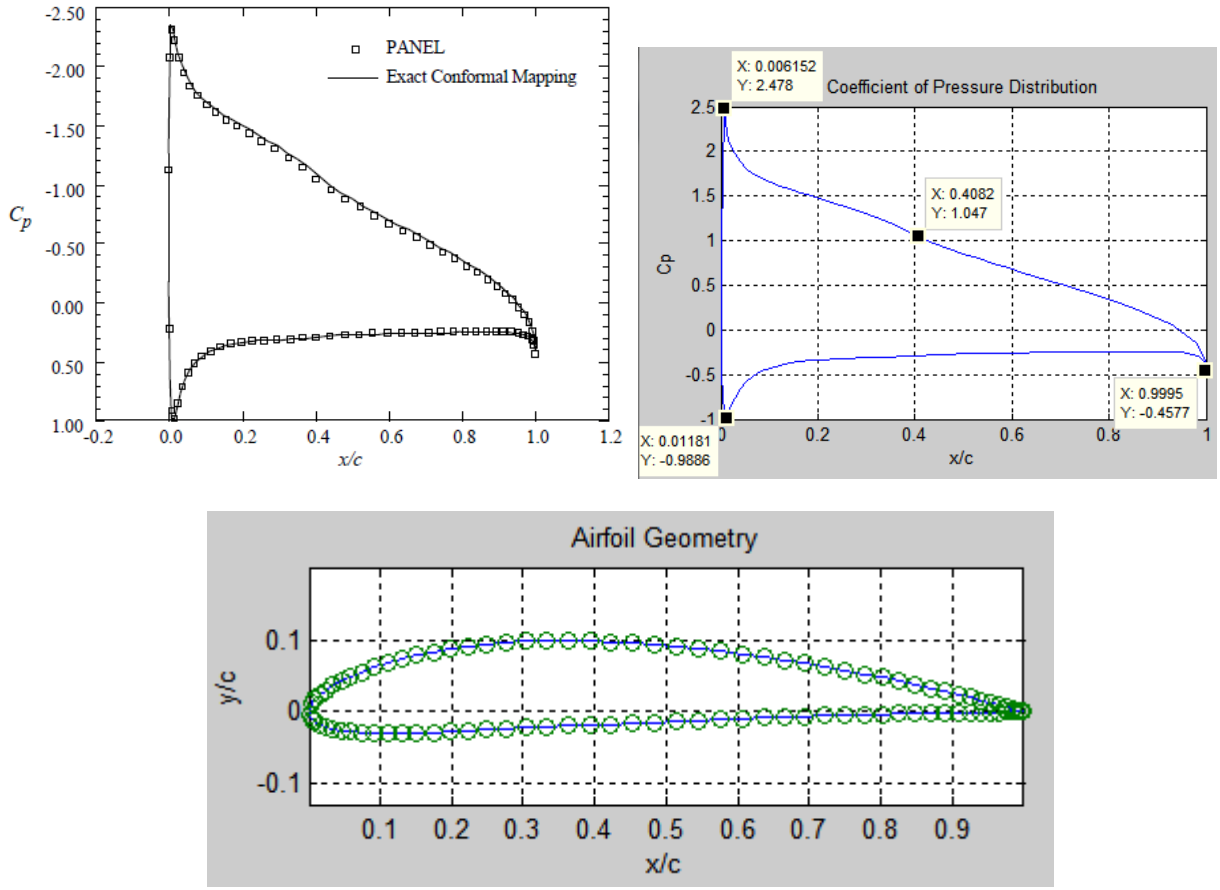


Fig. 9. Top-Left figure shows the plot from literature and Top-Right figure shows the generated plot in MATLAB for the simulation of NACA 4212 aerofoil with 6° of Angle of Attack. Bottom figure shows the aerofoil with its panels and control points. (The y-axis quantities have reversed signs in the top-right figure)

Global Search Techniques

These are a special class of optimization algorithms that **search the entire feasible set** of possible solutions to provide the most optimal set of parameters for the system. Since it is not computationally possible to search the entire feasible set by repeatedly plugging in every possible value into the objective function (in this case, say balancing efficiency) and then selecting the parameters that give the best value of the objective function, global search methods are usually also a **set a heuristics that give a direction to search** for optimal solutions.

In other words, these methods **narrow the search area for possible solutions**, thereby reducing computational loads compared to **brute-force methods**. These are very efficient techniques of solving multivariable as well as combinatorial problems as they provide avenues for testing the right set of combinations that lead to the optimal solution.

MATLAB Global Optimization Toolbox is a very powerful tool for making use of these techniques in cross disciplinary problems.

Evolutionary Optimization Algorithms

Many Global Search Techniques are inspired from physical and biological phenomena and are therefore referred to as **Evolutionary Optimization Algorithms**.

Some examples of such algorithms are:

- **Genetic Algorithm**, inspired from Darwin's theory of Natural Selection (Evolution of Species) that successfully explains the level of sophistication different living creatures have achieved depending on the natural constraints on the species.
- **Particle Swarm Optimization**, inspired from the searching/hunting techniques of swarm creatures such as honeybees, locusts or hornets. They work by constantly communicating locations of high food availability to the rest of the swarm and maintaining a memory of previous locations for the same.
- **Ant Colony Optimization** is inspired from the way social terrestrial insects such as ants and termites search for food locations. The worker ants leave behind a trail of pheromones which indicates to the next batch of workers locations of high food availability. Greater workforce is allocated to such locations.
- **Simulated Annealing**, inspired from the physical process of annealing, where random motion of particles inside a mass of near molten metal tries to achieve a state of minimum energy through slow cooling.

Genetic Algorithms in Engineering

Genetic Algorithms are extremely versatile tools for global optimization for any problem that can be posed in a way such that the solution lies in finding the **ideal vector** for a given objective function. The GA is one of a class of evolutionary design techniques in which successive generations of the product or process under consideration are defined by **different combinations of the design parameters** thereby producing results that tend to be increasingly fit for their purpose.

The process mimics natural evolution with biological processes like **inheritance**, **selection**, **crossover**, and **mutation**.

The basic necessity of GA is a **chromosome** which is nothing but a string of numbers of a vector that in itself has managed to uniquely define all the required characteristics of a design. A set of randomly generated chromosomes or a seed pool is used as an initial set of solutions to the problem.

A fitness function, which is nothing but the objective function calculates the performance index of each solution. This is used to select the best solutions in the population to be to the offspring that will comprise the next generation. **The fitter the parent, the greater its probability of selection**.

Offsprings are produced by selecting parent chromosomes for breeding, and crossing over the genetic material. For real numbers in each gene of the chromosome, operations such as weighted means are used for crossovers and they are governed by the fitness of each parent.

Small amount of mutation is encouraged so that the algorithm explores new region of state space. This is basically a randomized change in a few offspring where one or more genes attain values different from those attainable from their parents.

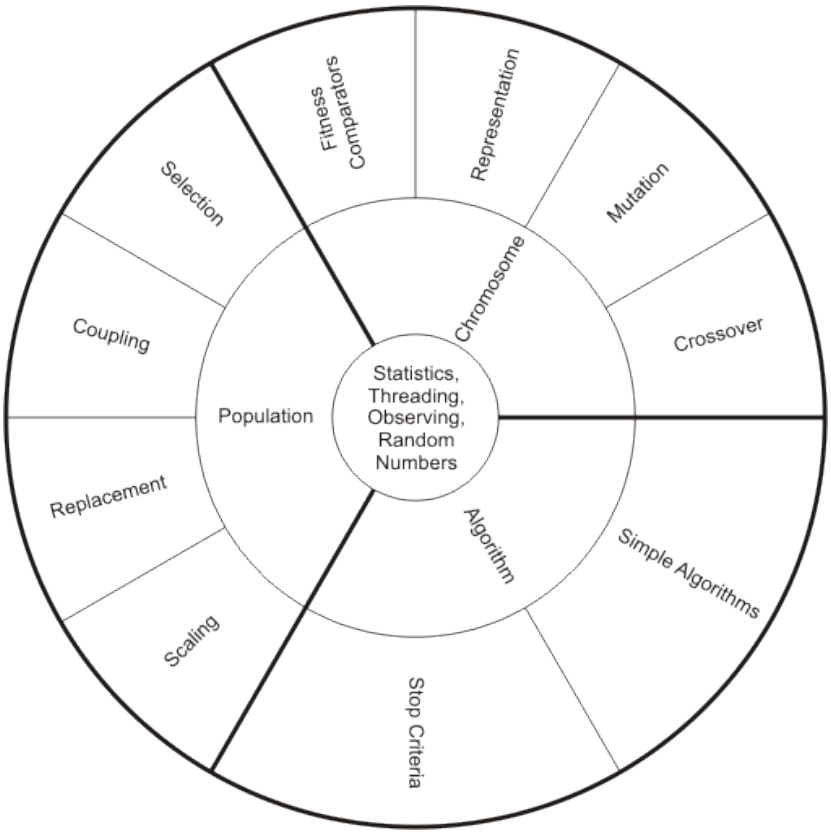


Fig. 10. A chart of the major operations involved in Genetic Algorithms

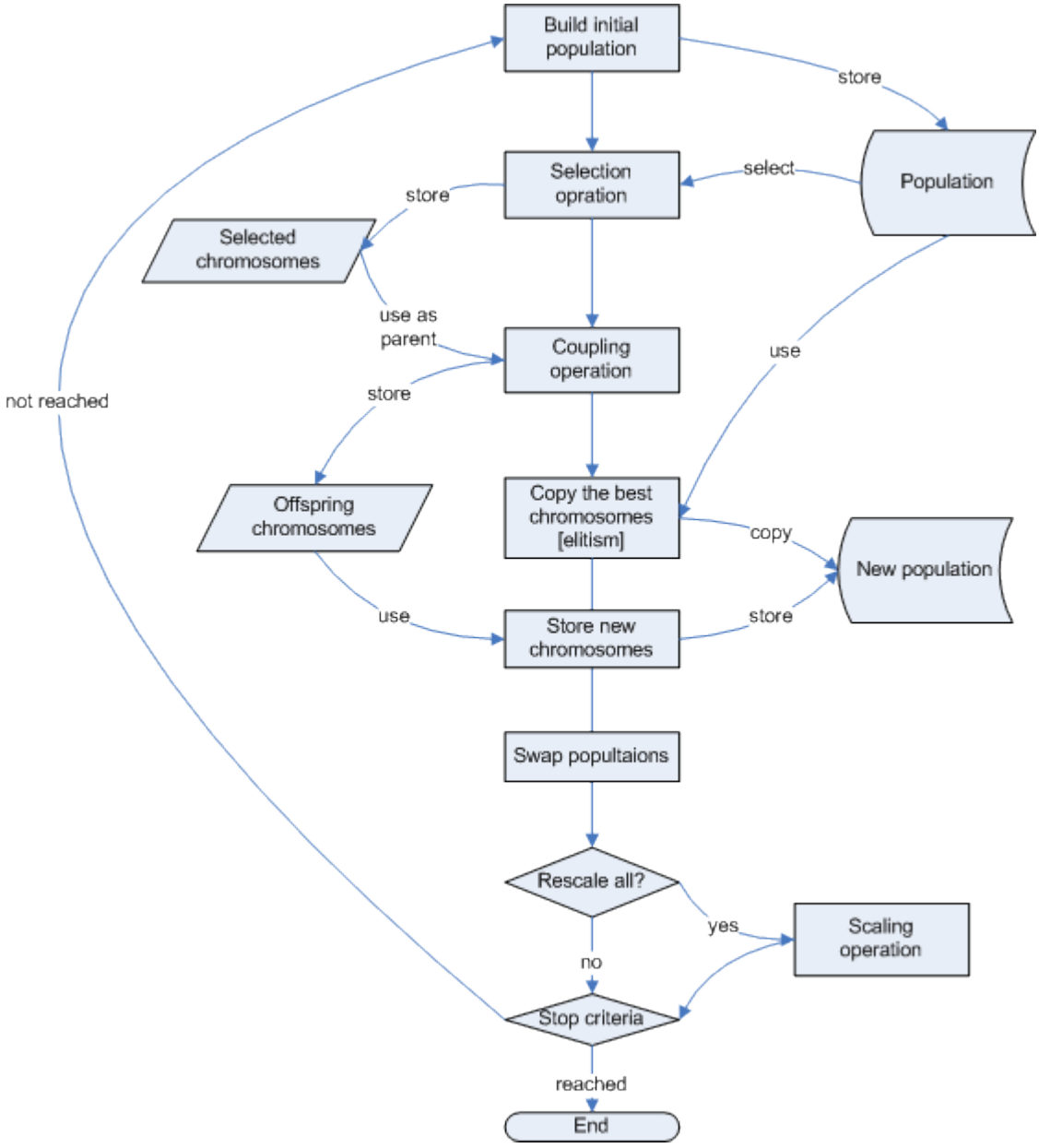


Fig. 11. Flow chart of a general Genetic Algorithm

In engineering, GAs are being used to **solve diverse interdisciplinary problems** as they are capable of finding satisfactory solutions without going into the physics of the problem. Although **computationally heavy** and **stochastically driven**, GAs have empirically proven themselves in solving problems otherwise harder without these soft computing techniques.

For this reason, some of the premier institutes in the world work on Genetic Algorithms research. This includes **IIT Kanpur**, which has its own Genetic Algorithms Laboratory headed by **Prof. Kalyanmoy Deb**. **University of Illinois, Urbana-Champagne** also has its own lab called **IlliGAL**. **University of Sydney** also has a research group called ADVENT in its Aerospace Division which focuses on multi-disciplinary research with Evolutionary Optimization Algorithms.

Results

A basic single-objective problem of **minimizing the ratio between lift and pressure drag** for a given **angle of attack of 4°** was posed and tested. With a limit of 30 iterations, an instance from the MATLAB Global Optimization Toolbox was started and here is the result:

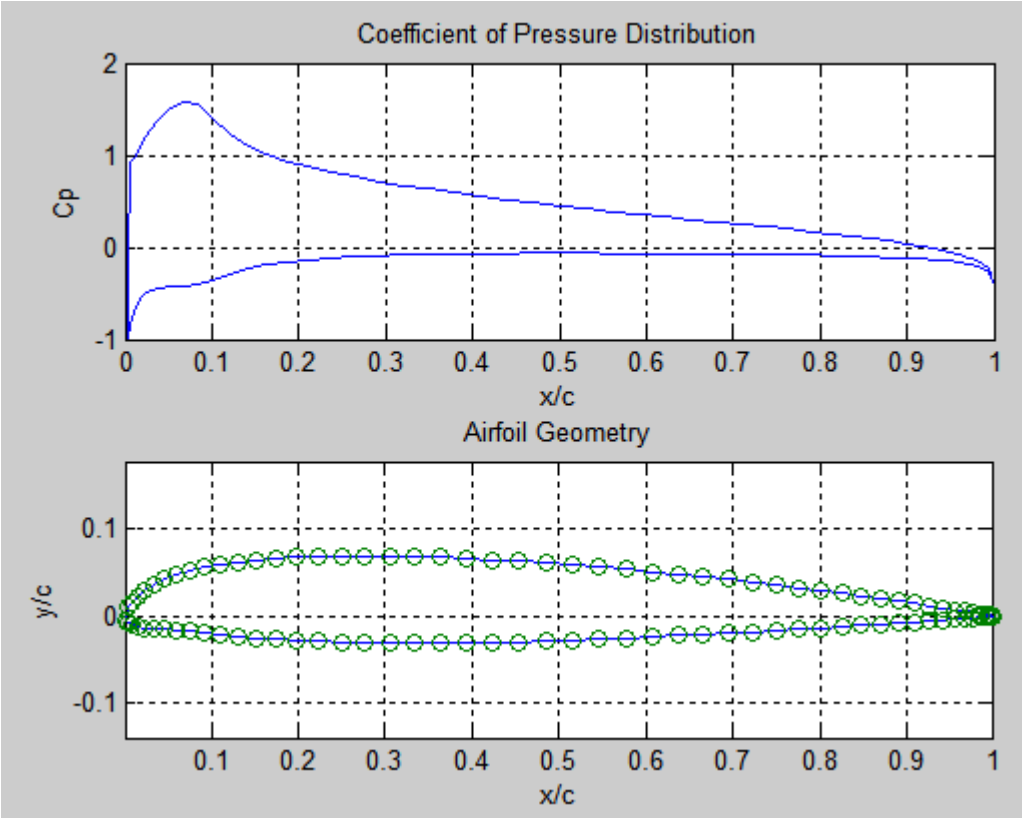


Fig. 12. Results from a preliminary test run

The aerofoil can be defined as NACA [1.992 1.064 10.004].
The solution converged in about 11 iterations with only minor (2nd and 3rd decimal) changes in the subsequent iterations.

Please note that the result above is carried out only for the inviscid case. The viscous effects are yet to be coded fully. We also need to pose our problems better if the generated aerofoil needs of be relevance when it comes to practical use.

Conclusions and Future Scope

The article presents a novel method for design new aerofoils using an unconventional technique. The final application although is far from complete. There are a number of tasks necessary to be performed before we can expect usable results from this application. They are:

- The Viscous-Boundary Layer routine must be fully implemented for physically correct values of lift, drag and aerodynamic moment
- Compressibility Effects may be accounted for slightly higher Mach numbers of 0.4-0.6
- Multi-Objective Optimization needs a lot of testing and tweaking
- The solver itself needs a lot of testing using results from analytical as well as numerical techniques

Additional Features:

- If the foil is being used in an environment where free stream velocity is a function of the position in the z-axis, then the panel method should be capable of evaluating the aerodynamic load coefficients simply using the velocity profile of the chamber/duct.
- In order to construct aircraft wings, a way of accounting the chord length into the optimization routine must be established. Various sections from the hub to the tip can be designed and lofted in a CAD software to generate the final wing.

This application need not limit itself to just aerofoils. Infact, turbomachinery **impeller blades** and **hydrofoils** could just as easily be designed using this. A diverse variety of geometrical routines must be made to take advantage of the modular and independent nature of the code created.

In addition, **alternative optimization methods** such as **Differential Evolution** can readily be implemented. A **comparative study** on various evolutionary methods is due.

This presents a **design methodology abstraction** where **analogous applications** could be made for **design of tricky components** from various disciplines of **engineering**.

Bibliography

- **Low Speed Aerodynamics** by Katz & Plotkin
- **Computational Fluid Dynamics for Engineers** by Tuncer Cebeci
- **Fundamentals of Aerodynamics** by J D Anderson
- **Aerodynamics4Students Online Textbook** by Prof. Douglas Auld from University of Sydney
- **MATLAB Help Documentation** by MathWorks Inc.

Summer Fellowship Outcomes

- It was an amazing opportunity to interact and work amongst the best of the students and professors in an **academically stimulating environment** like **IIT Madras** from the learning point of view.
- The project engaged me productively in out-of-the-box thinking and taught me a lot about **aerodynamics**, a discipline I was not introduced to being a Mechanical Engineering Student.
- Learned a lot about **coding custom solvers** for specific problems with interactions from the research scholars and my guide.
- The **seminars** conducted by the department to share progress opened windows for a lot of ideas and concepts I had previously not thought about.
- The fellowship gave me enough time to **explore Chennai**, a city new to me in my own way and **meet a lot of interesting people** on the way. I have a lot of memories to take home in that experience.
- The **institute conducted numerous talks** and seminars by experts which brought light to a lot of things. I really value the **insight** I gained in these sessions.
- Managed to produce my own solver for solving simple aerodynamics problems. The solver was then coupled with a geometry generation routine and a simple optimization routine to assist the **design of aerofoils**.
- The stay at **Cauvery Hostel** was also a memorable one. I made good use of the high speed internet connection here to download lectures that helped me deepen my understanding.
- Made friends with other **summer research fellows** from some of the best colleges across the country. **Exchange of ideas** and **discussions** on each other's project **benefitted everyone mutually**. Their presence in the vicinity lightened the mood whenever things weren't working as per plans.
- Experienced the **student culture of IIT** through close interactions with various professors and students from diverse backgrounds in order to gain clarity of my intentions for my future.
- The **strong research atmosphere**, **freedom and flexibility of work**, **quality/level of problems** that students work on inspired me a great deal to now actually **consider a career in academia** seriously.
- Met professionals at **IITM Research Park** to learn about **R&D prospects** in the industry for my own career.
- Gained a lot of **clarity** on the possible directions to **pursue higher education**.
- On a lighter note, the stay here multiplied the respect I had for the mess in my university! :D

Appendix – MATLAB Source Code

Optimized Code for Load Coefficient Calculations

NACA 4-Digit Aerofoil Coordinates (naca4.m)

```
function [x,y] = naca4(naca,npanel,x,y)


m=naca(1)/100;
p=naca(2)/10;
t=naca(3)/100;

% compute thickness and camber distributions

if mod(npanel,2) ~= 0
    sprintf('Please choose an even number of panels');
    sprintf('Exiting...');
    exit;
end

nside = npanel / 2 +1;

% camber distribution

for i=1:nside
    xx(i) = (1-cos(i*pi/nside))/2; %full cosine spacing
    %xx(i) = 1-cos(i*pi/(2*nside)); %half cosine spacing
    yt(i) = ( 0.29690*sqrt(xx(i)) -0.12600*xx(i)     ...
             -0.35160*xx(i)^2      +0.28430*xx(i)^3  ...
             -0.10360*xx(i)^4) * t / 0.20;
    if xx(i) < p
        yc(i) = m/p^2 * (2*p*xx(i) -xx(i)^2);
    else
        yc(i) = m/(1 -p)^2 * ((1 -2*p) + 2*p*xx(i)-xx(i)^2);
    end
end

% airfoil shape = camber + thickness

for i=1:nside
    x(nside+i-1) = xx(i);
    x(nside-i+1) = xx(i);
    y(nside+i-1) = yc(i) +yt(i);
    y(nside-i+1) = yc(i) -yt(i);

end

return
```

Panel Geometry (panel_geometry.m)

```
function [l,st,ct,xbar,ybar] = panel_geometry(x,y,npanel)

% compute various geometrical quantities
%[x y]

for i=1:npanel
    l   (i) = sqrt((x(i+1) -x(i))^2 +(y(i+1) -y(i))^2);
    st  (i) = (y(i+1) -y(i))/l(i);
    ct  (i) = (x(i+1) -x(i))/l(i);
```

```
    xbar(i) = (x(i+1) +x(i))/2;
    ybar(i) = (y(i+1) +y(i))/2;
end

return
```

Influence Coefficient (infl_coeff.m)

```
function ainfl = infl_coeff(x,y,xbar,ybar,st,ct,ainfl,npanel)

pi2inv = 1 / (2*pi);

% set vn = 0 at midpoint of ith panel

for i=1:npanel

    % find contribution of the jth panel

    for j=1:npanel

        log_term = 0.0;
        tan_term = pi;

        if i ~= j
            dxj    = xbar(i) -x(j);
            dxjp   = xbar(i) -x(j+1);
            dyj    = ybar(i) -y(j);
            dyjp   = ybar(i) -y(j+1);

            log_term = 0.5*log((dxjp*dxjp+dyjp*dyjp)/(dxj*dxj+dyj*dyj));
            tan_term = atan2(dyjp*dxj-dxjp*dyj,dxjp*dxj+dyjp*dyj);

        end

        ctimtj        = ct(i)*ct(j) +st(i)*st(j);
        stimtj        = st(i)*ct(j) -st(j)*ct(i);
        ainfl(i,j)    = pi2inv*(tan_term*ctimtj +log_term*stimtj);
        add           = pi2inv*(log_term*ctimtj -tan_term*stimtj);
        ainfl(i,npanel+1) = ainfl(i,npanel+1) + add; %last column

        if (i == 1 | i == npanel)

            % if the ith panel touches trailing edge
            %  add contribution to kutta condition

            ainfl(npanel+1,j)        = ainfl(npanel+1,j) - add;
            ainfl(npanel+1,npanel+1) = ainfl(npanel+1,npanel+1) + ainfl(i,j);
        end
    end
end

if rank(ainfl) ~= npanel+1
    exit
end

return
```

Velocity Distribution to calculate Dimensionless Pressure Distribution (veldis.m)

```
function vt = veldis(qg,x,y,xbar,ybar,st,ct,al,npanel)

% flow tangency boundary condition - source distribution

for i=1:npanel
    vt(i) = ct(i)*cos(al) +st(i)*sin(al);
    for j=1:npanel
        rp     = sqrt((xbar(i) -x(j+1))^2 +(ybar(i) -y(j+1))^2);
        rm     = sqrt((xbar(i) -x(j)  )^2 +(ybar(i) -y(j)  )^2);
        betaij = atan2 ((ybar(i) -y(j+1))*(xbar(i) -x(j)) ...
                -(xbar(i) -x(j+1))*(ybar(i) -y(j)),  ...
                    (xbar(i) -x(j+1))*(xbar(i) -x(j))   ...
                       +(ybar(i) -y(j+1))*(ybar(i) -y(j)));

                  %calculation of tangent influence coefficients
        if i==j
            log_term  = 0.0;
            beta_term = pi;
        else
            log_term  = log(rp/rm);
            beta_term = betaij;
        end

        ctimtj = ct(i)*ct(j) +st(i)*st(j);
        stimtj = st(i)*ct(j) -st(j)*ct(i);

        vt(i) = vt(i) + (qg(j)/(2*pi))*(stimtj*beta_term -ctimtj*log_term) ...
                + (qg(npanel+1)/(2*pi))*(stimtj*log_term +ctimtj*beta_term);
    end
end

return
```

Aerodynamic Load Coefficient Calculations (aero_coeff.m)

```
function [cl,cd,cm] = aero_coeff(x,y,cp,al,npanel)

cl = 0;
cd = 0;
cm = 0;

for i=1:npanel
    dx  = x(i+1) -x(i);
    dy  = y(i+1) -y(i);
    xa  = 0.5*(x(i+1) +x(i)) -0.25;
    ya  = 0.5*(y(i+1) +y(i));
    dcl = -cp(i)*dx;
    dcd = cp(i)*dy;
    cl  = cl +dcl;
    cd  = cd +dcd;
    cm  = cm +dcd*ya -dcl*xa;
end

dcl = cl*cos(al) -cd*sin(al);
cd  = cl*sin(al) +cd*cos(al);
cl  = dcl;


return
```

Hess Smith Solver routine combining the above (hess_smith.m)

```
function lbyd = hess_smith(naca)
clc
%
%specify inlet conditions
%
alpha=6; %in degrees
npanel=100; %total number of panels on the airfoil surface
%
% allocate all necessary arrays
%

x  = zeros(npanel+1,1);
y  = zeros(npanel+1,1);
cp = zeros(npanel+1,1);

%
% generate airfoil surface panelization
%

[x,y] = naca4(naca,npanel,x,y);

%
% generate panel geometry data for later use
%

l    = zeros(npanel,1);
st   = zeros(npanel,1);
ct   = zeros(npanel,1);
xbar = zeros(npanel,1);
ybar = zeros(npanel,1);

[l,st,ct,xbar,ybar] = panel_geometry(x,y,npanel);

%
% compute matrix of aerodynamic influence coefficients
%

ainfl = zeros(npanel+1);

ainfl = infl_coeff(x,y,xbar,ybar,st,ct,ainfl,npanel);
%ainfl
%
% compute right hand side vector for the specified angle of attack
%

b  = zeros(npanel+1,1);

al = alpha * pi / 180;

for i=1:npanel
    b(i) = st(i)*cos(al) -sin(al)*ct(i);
end

b(npanel+1) = -(ct(1)     *cos(al) +st(1)     *sin(al)) ...
              -(ct(npanel)*cos(al) +st(npanel)*sin(al));

%
% solve matrix system for vector of q_i and gamma
%
```

```
qg = inv(ainfl) * b;

%
% compute the tangential velocity distribution at the midpoint of panels
%

vt = veldis(qg,x,y,xbar,ybar,st,ct,al,npanel);


%
% compute pressure coefficient
%

cp = 1 -vt.^2;
%mincp=min(cp)
%
% compute force coefficients
%

[cl,cd] = aero_coeff(x,y,cp,al,npanel);

%
% plot the output
%

subplot(2,1,1),plot(xbar,-cp),xlabel('x/c'),ylabel('Cp'),title('Coefficient of
Pressure Distribution'),grid

subplot(2,1,2),plot(x,y,x,y,'o'),xlabel('x/c'),ylabel('y/c'),title('Airfoil
Geometry'),axis('equal'),grid
lbyd=-cl/cd;
%cd
%cl
return
```