2007

APPLICATION OF GENETIC ALGORITHMS AND CFD FOR FLOW CONTROL OPTIMIZATION

Narendra Beliganur Kotragouda
University of Kentucky, narendra.bk@gmail.com

ABSTRACT OF THESIS

APPLICATION OF GENETIC ALGORITHMS AND CFD FOR FLOW CONTROL OPTIMIZATION

Active flow control is an area of heightened interest in the aerospace community. Previous research on flow control design processes heavily depended on trial and error and the designers' knowledge and intuition. Such an approach cannot always meet the growing demands of higher design quality in less time. Successful application of computational fluid dynamics (CFD) to this kind of control problem critically depends on an efficient searching algorithm for design optimization. CFD in conjunction with Genetic Algorithms (GA) potentially offers an efficient and robust optimization method and is a promising solution for current flow control designs. Current research has combined different existing GA techniques and motivation from the two-jet GA-CFD system previously developed at the University of Kentucky propose the applications of a real coded Continuous Genetic Algorithm (CGA) to optimize a four-jet and a synthetic jet control system on a NACA0012 airfoil. The control system is an array of jets on a NACA0012 airfoil and the critical parameters considered for optimization are the angle, the amplitude, the location, and the frequency of the jets. The design parameters of a steady four-jet and an unsteady synthetic jet system are proposed and optimized. The proposed algorithm is built on top of CFD code (GHOST), guiding the movement of jets along the airfoil's upper surface. The near optimum control values are determined within the control parameter range. The current study of different Genetic Algorithms on airfoil flow control has been demonstrated to be a successful optimization application.

KEYWORDS: Flow Control, Genetic Algorithm, CFD, Optimization.

Narendra Beliganur Kotragouda

05/24/2007

APPLICATION OF GENETIC ALGORITHMS AND CFD FOR FLOW CONTROL OPTIMIZATION

By

Narendra Beliganur Kotragouda

Dr. Raymond .P. LeBeau
Director of Thesis

Dr. L. S. Stephens
Director of Graduate Studies

RULES FOR THE USE OF THESIS

Unpublished thesis submitted for the Master's degree and deposited in the University of Kentucky Library are as a rule open for inspection, but are to be used only with due regard to the rights of the authors. Bibliographical references may be noted, but quotations or summaries of parts may be published only with the permission of the author, and with the usual scholarly acknowledgements.

Extensive copying or publication of the thesis in whole or in part also requires the consent of the Dean of the Graduate School of the University of Kentucky.

A library that borrows this thesis for use by its patrons is expected to secure the signature of each user.

Name	Date

THESIS

Narendra Beliganur Kotragouda

The Graduate School

University of Kentucky

2007

APPLICATION OF GENETIC ALGORITHMS AND CFD FOR FLOW CONTROL OPTIMIZATION

THESIS

A thesis submitted in partial fulfillment of the
requirements for the degree of Master of Science in Mechanical Engineering
in the College of Engineering at the University of Kentucky

By

Narendra Beliganur Kotragouda
Lexington, Kentucky

Director: Dr. Raymond .P. LeBeau

Assistant Professor of Mechanical Engineering

University of Kentucky Lexington, Kentucky

2007

To My Parents

ACKNOWLEDGEMENTS

I would like to express my gratitude to Dr. Raymond P. LeBeau for being an outstanding advisor and an excellent professor. His constant encouragement, support, and invaluable suggestions have made this work successful. My sincere thanks go to Dr. Johne' M. Parker and Dr. Jian Sheng for serving on my thesis committee. I would also like to thank Dr. Thomas Hauser of Utah State University for his useful inputs and Kentucky NASA EPSCoR for providing financial support for this project.

I am deeply and forever indebted to my parents for their love, support and encouragement throughout my entire life. I am also very thankful to my brother Dinesh and sister Rajeswari for their continuous encouragement and love without which none of this was possible. Last but not the least, I would also like to thank my friends for their valuable discussions and cooperation.

TABLE OF CONTENTS

LIST OF TABLES vi

LIST OF FIGURES vii

LIST OF FILES ix

1. INTRODUCTION 1
   1.1 OVERVIEW 1
   1.2 BACKGROUND 2
   1.3 FLOW CONTROL 3
      1.3.1 PASSIVE FLOW CONTROL 4
      1.3.2 ACTIVE FLOW CONTROL 5
      1.3.3 FLOW CONTROL AND CFD 5
   1.4 OPTIMIZATION TECHNIQUES 7
      1.4.1 GENETIC ALGORITHMS 8
   1.5 OBJECTIVES 8
   1.6 ORGANIZATION OF THESIS 9

2. LITERATURE REVIEW 11
   2.1 FLOW CONTROL 11
      2.1.1 FLOW CONTROL - CONVENTIONAL 11
         2.1.1.1 PASSIVE CONTROL 13
         2.1.1.2 ACTIVE CONTROL 15
            2.1.1.2.1 SUCTION AND BLOWING 16
            2.1.1.2.2 SYNTHETIC JETS 17
            2.1.1.2.3 MORPHING WING 18
      2.1.2 FLOW CONTROL AND CFD 19
   2.2 OPTIMIZATION ALGORITHMS 23
      2.2.1 REVIEW OF GENETIC ALGORITHMS 26
      2.2.2 GENETIC ALGORITHMS AND CFD 28
   2.3 SUMMARY 32

3. GENETIC ALGORITHMS 33
   3.1 OVERVIEW 33
   3.2 CONVENTIONAL GENETIC ALGORITHM 37
   3.3 EARND GENETIC ALGORITHM 39
   3.4 CONTINUOUS GENETIC ALGORITHM 43
      3.4.1 COMPONENTS OF CONTINUOUS GA 43
         3.4.1.1 VARIABLE ENCODING, PRECISION, AND BOUNDS 44
         3.4.1.2 INITIAL POPULATION 45
         3.4.1.3 NATURAL SELECTION 47
         3.4.1.4 PAIRING 48
         3.4.1.5 MATING/CROSSOVER 49
         3.4.1.6 MUTATION 50
         3.4.1.7 THE NEXT GENERATION 51
   3.5 SUMMARY 53

4. COMPUTATIONAL TOOLS 54
   4.1 GRID GENERATION 54
      4.1.1 FLEXGRID.F90 54
      4.1.2 G.F90 58
      4.1.3 INPUT FILE – “INPUT” 59
   4.2 GHOST 59
   4.3 GA-CFD SYSTEM 61
   4.4 COMPUTATIONAL PLATFORMS – KENTUCKY FLUID CLUSTERS 62
   4.5 SUMMARY 64

5. STEADY FOUR JET RESULTS 65
5.1 GRID AND BOUNDARY CONDITIONS 65
5.2 PARAMETER SELECTION 67
5.3 GENETIC PARAMETERS 69
5.4 OPTIMIZED CONFIGURATION - CGA 71
5.5 FLOW CONTROL PHYSICS 82
5.6 EARND GA 89
5.7 COMBINATION OF CGA AND EARND GA CONFIGURATION 97
5.8 SUMMARY 98

6. UNSTEADY SYNTHETIC JET RESULTS 100
6.1 SYNTHETIC JETS 100
6.2 CASE SETUP 101
6.3 PARAMETER SELECTION 105
6.4 GENETIC PARAMETERS 108
6.5 HYBRID UNSTEADY TWO JET RESULTS 109
6.6 SUMMARY 116

7. GA-NEURAL NETWORKS-CFD 117
7.1 NEURAL NETWORKS 117
7.2 GA-NN-CFD SYSTEM AND INITIAL RESULTS 119
7.3 SUMMARY 122

8. CONCLUSIONS AND FUTURE WORK 123
8.1 CONCLUSIONS 123
8.2 FUTURE WORK 125

APPENDIX 127
A1. FLEXGRID.F90 MODIFICATION FOR STEADY CASE 127
A2. A TYPICAL 'INPUT' FILE 128
A3. FLEXGRID.F90 MODIFICATION FOR UNSTEADY CASE 131

REFERENCES 139

VITA 147

LIST OF TABLES

Table 3.1: Initial population arranged according to fitness 47
Table 3.2: Chromosomes which survived the selection process..... 48
Table 3.3: Variables and fitness at the end of 1st generation 51
Table 3.4: Variable and fitness values after convergence..... 52
Table 4.1: *i* and *j* points of the 16 grid blocks 58
Table 5.1: Parameter range for the four jet case 71
Table 5.2: Best configuration obtained from CGA simulation..... 72
Table 5.3: Best 10 individuals from CGA simulation 73
Table 5.4: Separation point for various configurations 85
Table 5.5 Best configuration from the EARND GA..... 90
Table 5.6 Best 10 configurations obtained using EARND GA 91
Table 5.7 Jet configuration and fitness using both GA configurations..... 97
Table 6.1: Parameter range for the pure oscillating case 106
Table 6.2: Modification of variables for hybrid unsteady case 108
Table 6.3: Best 5 individuals of the unsteady 2-jet case – CGA 110
Table 8.1: Improvements in parameters for steady case..... 123
Table 8.2: Improvements in parameters for hybrid unsteady case 123
Table 8.3: Best configuration for the steady four-jet case 124
Table 8.4: Best configuration for the hybrid unsteady two-jet case 124

LIST OF FIGURES

Figure 1.1 Classification of flow control techniques, Gad-el-Hak [13] 4
Figure 1.2 Predetermined, open-loop control, Gad-el-Hak [15] 5
Figure 1.3: The "three dimensions of fluid dynamics" 6
Figure 2.1: Interrelation between flow control goals [15] 12
Figure 3.1: Process flow of Conventional flow chart. 39
Figure 3.2: Process flow chart of the EARND GA [20] 42
Figure 3.3: Contour plot of the example problem 46
Figure 3.4: Contour plot showing the initial population 48
Figure 3.5: Contour plot showing region of high fitness (minimum cost) in blue 52
Figure 4.1: Multi-zone grid setup 55
Figure 4.2: A: old two-jet grid (multi jet blocks), B: new grid (single jet block) 57
Figure 4.3: KFC6-I and KFC5 63
Figure 5.1 (a): Grid and boundary conditions for four jet case 66
Figure 5.1 (b): Jet parameters for steady case [20] 68
Figure 5.2 (a): Fitness of all configuration from CGA 74
Figure 5.2 (b): Location sorted by fitness - CGA 74
Figure 5.2 (c): Amplitude sorted by fitness – CGA 75
Figure 5.2 (d): Angle sorted by fitness – CGA 75
Figure 5.3 (a): Fitness of best 500 individuals – CGA 77
Figure 5.3 (b): Location of best 500 individuals - CGA 77
Figure 5.3 (c): Amplitude of best 500 individuals - CGA 78
Figure 5.3 (d): Angle of best 500 individuals – CGA 78
Figure 5.4 (a): Best 500 leading suction jet configurations (fixed amplitude) 79
Figure 5.4 (b) Best 500 trailing suction jet configurations along with fitness 79
Figure 5.4 (c): Best 500 leading blowing jet configurations along with fitness 80
Figure 5.4 (d) Best 500 trailing blowing jet configurations along with fitness 81
Figure 5.5: Streamline and vorticity plot using the CGA configuration 82
Figure 5.6: C_p (coefficient of pressure) using the CGA configuration 83
Figure 5.7: Pressure plots using the CGA configuration 84
Figure 5.8: C_f (skin friction) using the CGA configuration 85
Figure 5.9: C_l and C_d using the CGA configuration 86
Figure 5.10 (a) to (d): Jet parameters for configurations having fitness within 1.5% of the maximum fitness 88
Figure 5.12 (a): Location of jets from EARND GA (top) and CGA (bottom) 93
Figure 5.12 (b): Amplitude of jets from EARND GA (top) and CGA (bottom) 94
Figure 5.12 (c): Angle of jets from EARND GA (top) and CGA (bottom) 95
Figure 5.13 (a): Vorticity and streamline plot with EARND GA optimum jet configuration 96
Figure 5.13 (b): Vorticity and streamline plot with CGA optimum jet configurations 96
Figure 5.14: Vorticity and streamline comparisons 98
Figure 6.1: A 2D synthetic jet interacting with a laminar boundary layer [108] 101
Figure 6.2: Lift and drag convergence for the unsteady synthetic jet case 103
Figure 6.3: Comparison of unsteady (a) and steady (b) grid setup 104
Figure 6.4: Vorticity and streamline plot for pure unsteady two-jet case 107
Figure 6.5: Results of the pure oscillating unsteady two-jet case (24 generations) 107
Figure 6.6: Fitness plots of unsteady simulations 111
Figure 6.7: Scatter plots for parameters of the unsteady case – CGA (both simulations) 114

Figure 7.1: A typical neural network 118
Figure 7.2: Fitness comparison GA-CFD and GA-NN system 120
Figure 7.3: Lift and drag comparisons of GA-CFD and GA-NN system 121

LIST OF FILES

1. Beliganur_Thesis.pdf: ~ 5 MB (File Size)

CHAPTER – 1

1. INTRODUCTION

1.1 OVERVIEW

The ability to move with freedom seems basic to man's nature. Man's desire to travel and explore has driven innovation in the field of transportation. The desire to travel around the world and to get there faster has made air travel a very important means of transportation. Regional, national, and cross-continental flights have therefore become hallmarks of air transportation throughout the world, personifying a free society and the pursuit of happiness to most human wishes.

However, the demands concerning the performance of aircraft are increasing. Increases in fuel prices together with the high fuel consumption of large airplanes have heightened the research interest of the aerospace community in the area of flow control. Active flow control has been increasingly used by the aerospace community to enhance flow control efficiency through alteration of flow field. Examples of active flow control techniques are projectile maneuvering with pulsating jets, changing/oscillating the shape of the wing (morphing wing), steady suction and blowing jets, and plasma actuators. With the advancements in technology and availability of relatively inexpensive and fast computers, Computational Fluid Dynamics (CFD) has been used to simulate such active flow control setups [1], [2]. Most active flow control techniques have a complicated parameter space and therefore require an efficient search system to predict a near optimum configuration of the control parameters. Previously this process heavily relied on designers' knowledge and intuition or trial and error methods. Such an approach is necessarily limited and cannot always meet the demands of high design standards. One possible solution to this problem is a combination of CFD and evolutionary search algorithms, such as a Genetic

Algorithm (GA) [3]. Genetic Algorithms have been successfully used as an optimization tool in diverse fields such as operational research [4], multi disciplinary design optimization [5] [6], management [7], logistics [8], and aerospace applications [9]. Work done as part of this thesis aims to study flow control by means of separation control using suction, blowing, and synthetic jets, and optimizing the design parameters of these jets using a Continuous Genetic Algorithm. A steady four jet control system and an unsteady two jet control system are setup on a NACA0012 airfoil and the control effects of location, amplitude, frequency and angle of jets are optimized. Optimum parameters are searched using two different GAs and results are compared to decide the best GA for a particular problem. Optimized results from both the GAs have achieved the design goal of high lift and low drag coefficients within the available limits of the parameter space. Finally, an attempt is made to substitute the time consuming, complex CFD calculations with rapidly converging non-linear interpolation methods, viz. Neural Networks [10] and an estimation of the potential overall reduction in the computation time is discussed.

1.2 BACKGROUND

Flow control, which is the ability to manipulate a flow field to effect a preferred change, is of immense interest in the aerospace community. Ever since the first flight of the Wright Brothers, flow control has been a much pursued area of research by engineers and scientists around the world. The numerous potential benefits range from saving billions of dollars by reducing fuel consumption of commercial aircraft to improving the ease of maneuverability of military aircraft. Conventionally this was achieved by altering the shape (passive flow control) of the aircraft, mainly aircraft wings. This method has come to a near saturation and further progress calls for a more sophisticated and complex means of flow control, in other words active flow control. These complex flow control systems require an efficient search algorithm to predict the configuration of the control parameters. This research focuses on two such approaches where

suction/blowing jets and synthetic jets are used to control separation and a Genetic Algorithm is employed as the optimization algorithm. The basis of this research is the previous [3] steady two jet (one suction and one blowing) control system which was extensively studied and optimized. In present work, a steady four jet control system and an unsteady two jet control system are optimized using an advanced Genetic Algorithm.

1.3 FLOW CONTROL

Flow control can be defined as a process used to alter a natural flow state or development path (transient between states) into a more desired state (or development path; e.g. laminar, smoother, faster transients) [11]. In the context of present research, it could be more precisely defined as modifying the flow field around the airfoil to increase lift and decrease drag. This could be achieved by using different flow control techniques such as blowing and suction, morphing wing, plasma actuators, and changing the shape of the airfoil [12]. Still all the techniques mentioned here essentially do the same job, i.e. reduce flow separation so that the flow is attached to the airfoil and thus reduce drag and increase lift. Flow control techniques can be broadly classified (Figure 1.1) as active and passive flow control which can further be classified into more specific techniques [13]. However, with reference to the research at hand, only active and passive flow control techniques are discussed. The terms "*active*" or "*passive*" do not have any clearly accepted definitions, but nonetheless are frequently used. Typically, the classification is based either on energy addition, on whether there are parameters (such as oscillating frequency) that can be modified after the system is built, or on whether the control system is steady or unsteady. In present context it will be distinguished based on energy addition, i.e. active flow control can be either steady or unsteady, but requires external energy, while passive flow control does not require external energy.

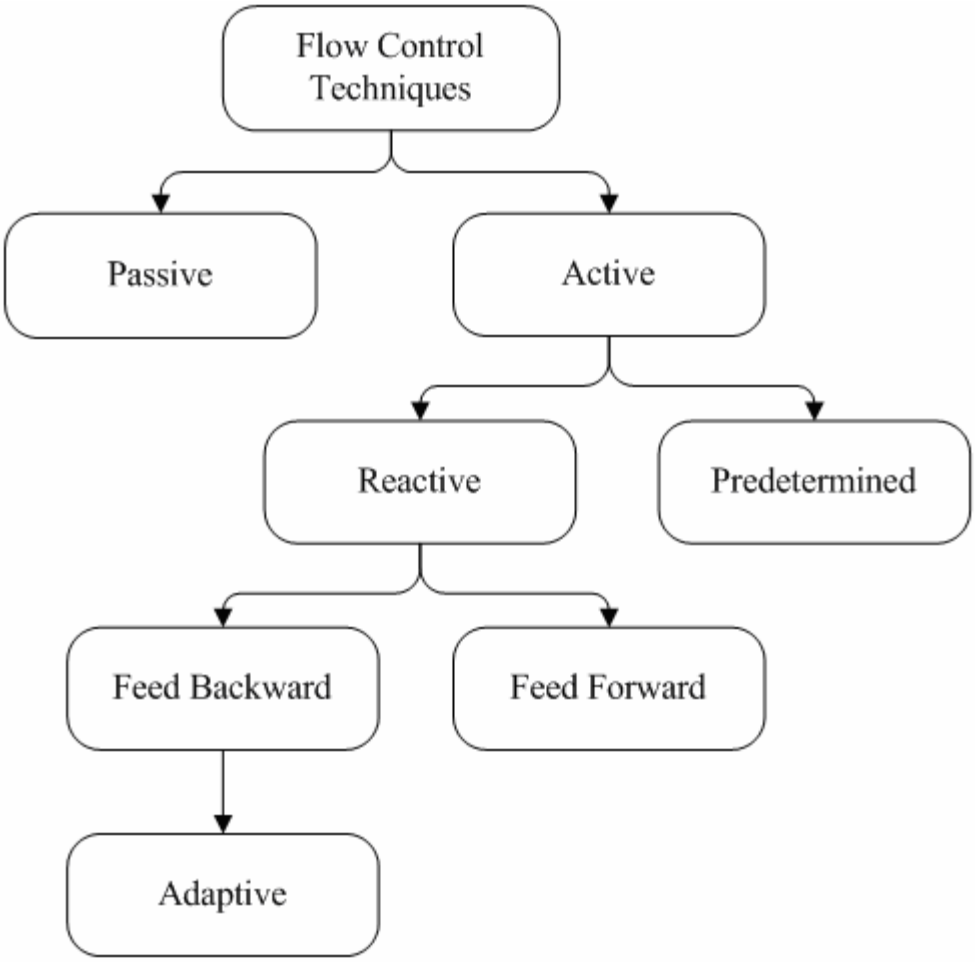


Figure 1.1 Classification of flow control techniques, Gad-el-Hak [13]

1.3.1 PASSIVE FLOW CONTROL

Passive flow control is a flow control technique which does not require auxiliary power or energy to be added to the flow. Most common forms of passive flow control are modifying the wing geometry, flaps on aircraft wings, and similar shape modifications, all to reduce drag and increase lift. The fundamental principle of this technique is boundary layer control, which most commonly involves suppression or delay of separation. Apart from the common forms and techniques, many other passive techniques have been successful in reducing skin friction in a turbulent flow, such as polymers, particles, and vortex generators or riblets, which appear to act indirectly through local interaction with discrete turbulent structures; particularly small-scale

eddies within the flow. Common characteristics of all of these passive methods are increased losses in the near-wall region, thickening of the buffer layer, and lowered production of Reynolds shear stress [14].

1.3.2 ACTIVE FLOW CONTROL

Active flow control is a scheme which involves energy expenditure and a control loop. As shown in Figure-1.1, it can be further classified into predetermined and reactive flow control. Predetermined control includes the application of steady or unsteady energy input without regard to the particular state of the flow. The control loop in this case is open, and no sensors are required (Figure 1.2). Because no sensed information is being fed forward, this open control loop is not a feedforward type. Often, this is misunderstood and treated as reactive, feedforward control. Active technique is a special form of flow control technique which uses dynamic data during the control process and regulates the input parameters. The control loop here could be open or closed and depending on that the techniques could be further classified.

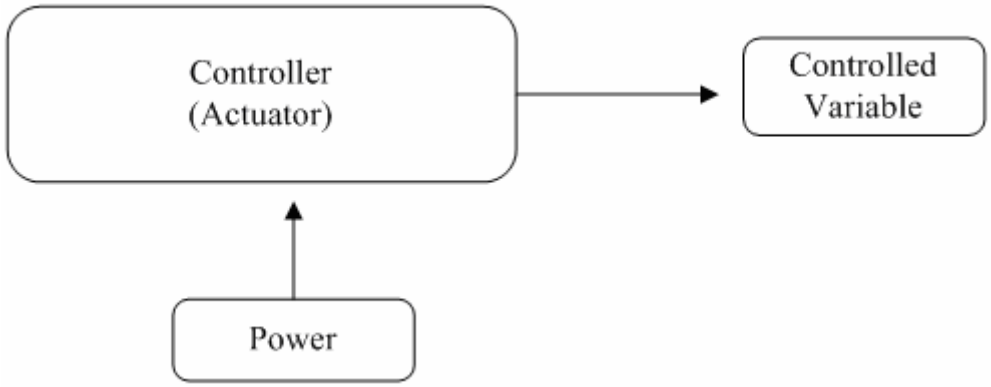


Figure 1.2 Predetermined, open-loop control, Gad-el-Hak [15]

1.3.3 FLOW CONTROL AND CFD

The potential benefits of realizing efficient flow-control systems range from saving billions of dollars in annual fuel costs for land, air, and sea vehicles to achieving economically and environmentally more competitive industrial processes involving fluid flows. Unfortunately,

current experimental setups (wind tunnels) do not have the capability of testing numerous new and efficient flow control setups. It may be possible to build such an experimental setup, but optimization would involve testing numerous configurations of the flow control setup being investigated.

In early 20th century fluid dynamics, we were operating in the "two-approach world" of theory and experiments [16]. However, the advent of high speed digital computers combined with development of accurate numerical algorithms has introduced a "third approach" (Figure-1.3) in fluid dynamics, viz. Computational Fluid Dynamics (CFD).

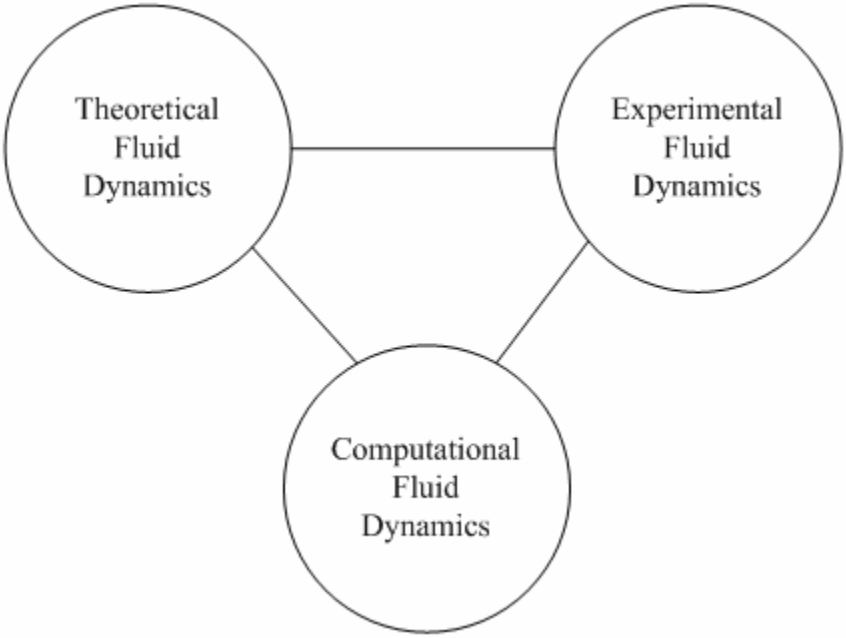


Figure 1.3: The "three dimensions of fluid dynamics"

Considering the sensitivity and large parameter space of the proposed flow control problems, it would take years of time and require advanced experiments and equipment to simulate all the configurations generated by the GA. Thus, CFD plays a vital role in successful implementation of current research. It is systematic, relatively inexpensive, and practical; in addition, the results are reliable and reasonably resemble the experimental data in many cases. CFD still cannot

completely replace experiments, but once validated with some simple setups of the problem, could be used to test the large number of configurations generated by the Genetic Algorithm.

1.4 OPTIMIZATION TECHNIQUES

Optimization is the process of making something better. Engineers and scientists present a new idea and optimization improves that idea. It involves trying variations on an initial concept and using the information gained to improve the idea. Over years optimization techniques have been applied in diverse fields ranging from operations and economics to engineering and medicine. Optimization techniques have undergone a great deal of change in recent times which allows us to apply these techniques to the most complex problems of today's world.

'Would you tell me, please, which way I ought to go from here?'
'That depends a good deal on where you want to get to', said the Cat.
'I don't much care where...', said Alice.
'Then it doesn't matter which way you go', said the Cat.
'So long as I get somewhere', Alice added as an explanation.
'Oh, you're sure to do that', said the Cat, 'if you only walk long enough.'
(Lewis Carroll: Alice in Wonderland, p.33)

This conversation between Alice and the Cat gives a perfect depiction of the tortuous path, full of dead locks, sharp curves, and hurdles that one has to face while dealing with a highly complex and non-linear optimization problem [17]. Optimization of the flow control technique used in this research is one of the examples of this kind of problem. Conventional optimization tools such as gradient based, enumerative, and random search tools fail to do a good job with such problems, mainly because of lack of robustness and/or because they tend to get stuck in local minima/maxima.

Evolutionary algorithms (EA's), which are fundamentally different from traditional approaches, are perhaps the most suitable alternative. Evolutionary algorithms use ideas and get inspiration from natural evolution and adaptation. Genetic Algorithms are a class of EA's which are most widely used for optimizing highly non-linear and complex problems.

1.4.1 GENETIC ALGORITHMS

The Genetic Algorithm is an optimization technique based on the principles of genetics and natural selection. GAs were originally developed by John Holland [18] at the University of Michigan in 1975. GAs are considered the most powerful evolutionary technique and are the most broadly applicable stochastic search technique for optimization problems than any other similar technique. In general, a GA has five basic [19] components,

- A genetic representation of solutions to the problem
- A way to create an initial population of solutions
- An evaluation function rating solutions in terms of their fitness
- genetic operators that alter the genetic composition of offspring during reproduction
- Values of the parameters of the GA

Genetic Algorithms are better than other techniques to solve intricate engineering problems because of the large population of individuals; it gives the GA a diverse search space which reduces the likelihood of converging to a non-global solution. Additionally, as a GA could be easily parallelized, it can be simply integrated into existing evaluation software (such as CFD and FEA solvers) and each set of individuals can be solved simultaneously on different processors. Thus, a GA is optimally suited for the optimization problem that is considered in this research.

1.5 OBJECTIVES

Active flow control using suction and blowing jets, although studied by various researchers in depth, still leaves much to be explained, especially the optimization part of the mechanism. This research is a continuation of work done by Liang Huang [20] at the University of Kentucky in 2004. Liang Huang studied the effects of suction and blowing jets on a

NACA0012 airfoil and used an EARND GA to optimize the various jet parameters of a two-jet control system. In the current research, a four jet control system is developed and optimized using a Continuous GA (CGA) [21] and the results are compared with the EARND GA results. Secondly, a more challenging unsteady (synthetic jets) two jet control system is setup and optimized using the Continuous GA.

Various grid changes have been performed during the course of the research to ease the positioning of jets and also to accommodate the finer synthetic jets. A total of about 9000 simulations have been performed and studied.

In summary, this research involved the following,

- Developing a better airfoil grid to ease the process of jet positioning for a four jet control system, which could be easily extended to an array of jets.
- Developing a real coded GA called the Continuous GA.
- Optimizing the four jet control system using CGA and comparing the results with the EARND GA results.
- Setting up an unsteady (synthetic jet) two jet control system.
- Modifying the grid generation codes and GHOST to accommodate synthetic jets.
- Optimizing the unsteady synthetic jet control system using the CGA

1.6 ORGANIZATION OF THESIS

Chapter 1 introduces the research and provides an overview along with background information. A brief introduction of the problem being investigated and the proposed solution methods are also presented in this chapter. Chapter 2 deals with the literature review, where the recent developments in the field of active flow control and Genetic Algorithms as an optimization tool are discussed. In Chapter 3 we talk about the GA as an optimization tool and discuss in detail the GAs developed for current research. Chapter 4 deals with the computational

tools and the grid generation process along with a discussion of the basic case setup. Chapter 5 and 6 presents the results of steady and unsteady jet control system respectively followed by a discussion of the GA-Neural Network- CFD system in chapter 7. Conclusions and future work are put forth in chapter 8.

CHAPTER – 2

2. LITERATURE REVIEW

The objective of this chapter is to present a review of the various flow control and optimization techniques relevant to the current research; therefore, importance will be given to flow control techniques involving steady (suction and blowing) and synthetic jets, and optimization methods involving evolutionary algorithms.

2.1 FLOW CONTROL

The art of flow control has roots in prehistoric times when streamlined spears, sickle shaped boomerangs, and fin-stabilized arrows were designed empirically by early Homo sapiens. Modern man has likewise applied flow control methods to achieve many technological goals [15].

The science of effective flow control however, originated with Prandtl (1904), who introduced boundary layer theory, explained the physics of separation phenomena, and described several experiments in which the boundary layer was controlled. Prandtl also pioneered the modern use of flow control [22] - he introduced the idea of self-similarity, explained the mechanics of steady two-dimensional separation, and opened the way for understanding the motion of real fluids. Subsequently in the late 1950's, Thwaites [23], Stratford [24] and Curle et al. [25] defined the various methods for predicting laminar and turbulent boundary layers, which broadened the 'way' that was opened by Prandtl.

2.1.1 FLOW CONTROL - CONVENTIONAL

Today, among the various types of flow control methods, separation control or boundary layer control (BLC) is the most common and economical method used in the aerospace and automobile industry. The primary goal of this approach is to increase lift and decrease drag.

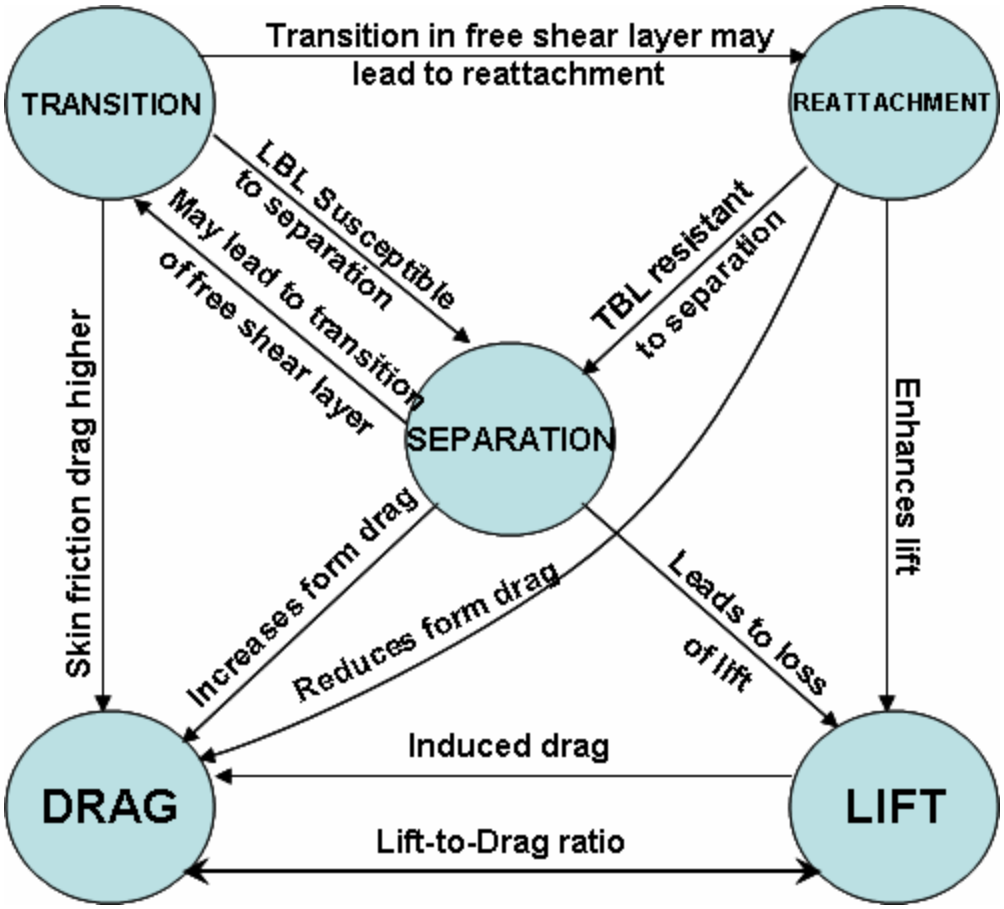


Figure 2.1: Interrelation between flow control goals [15]

The general goals of a flow control technique are summarized in Figure 2.1. Separation of flow is governed by two factors, adverse pressure gradient and viscosity. In order to remain attached to the surface, the stream must have sufficient energy to overcome the adverse pressure gradient, the viscous dissipation along the flow path, and the energy loss due to the change in momentum. This loss has a more pronounced effect in the neighborhood of the wall where momentum and energy are much less than in the outer part of the shear layer. If the loss is such that further advancement of the fluid is no longer possible, then the flow leaves the surface, i.e., the flow separates. In two dimensional flows, the criterion of separation is formulated by zero velocity gradients at the wall,

$$\left.\frac{\partial u}{\partial y}\right|_{y=0} = 0$$

or zero wall friction. Therefore, conceivable techniques for separation control are

1) to design the body surface configuration in such a way that a sufficient high energy level is maintained along the flow path in the neighborhood of the wall or

2) to augment the energy level by an auxiliary device placed at a suitable position along the flow path.

The first method is called passive flow control and the second one active flow control.

2.1.1.1 PASSIVE CONTROL

Early work on separation control mainly concentrated on passive flow control methods, i.e. using methods which do not require auxiliary power to operate. These include but are not limited to modifying the geometric shape, using riblets, changing the surface condition, and vortex generators.

Boundary layer control is divided into laminar separation control and turbulent separation control. With the advent of new technologies emphasis has been on reducing separation, thereby considerably increasing the L/D (lift to drag) ratio. Numerous researchers have done extensive research in the field of turbulent separation control as the airfoils in this region are used in the general aviation industry.

The main goal of laminar flow control is to increase lift and reduce drag by controlling separation or controlling the point of reattachment, or delaying the transition [15]. There are many interdependencies in these control objectives as depicted by Gad-el-Hak [15] in Figure 2.1. The present research mainly emphasizes on increasing the lift to drag ratio by reducing the separation.

The criterion for transition to turbulence was studied by several researchers such as Crabtree [26]. Since turbulence was not fully understood, many approximate methods, based on semi-empirical theories for the criteria of turbulence separation, had been devised, such as the methods by Thwaites [23] and Maskell [27]. The effects of compressibility on separation were also studied and tested by Reshotko et al. [28], Allen et al. [29] and Stack [30]. But all analytical studies were limited to simple conditions and assumptions; hence the predictions did not agree with the experiments in most cases. The flow control setups proposed in this research primarily focus on the flow over an airfoil, therefore a few experimental examples of airfoil related flow control methods are presented here.

Streamlining considerably reduces the separation by reducing the pressure rise. McCullough et al [31] conducted their experiments on three different airfoil sections NACA 63_3-018, NACA 63-009, and NACA 64A006 which have different thickness values and different leading edge radii. NACA 63_3-018 showed maximum lift when plotted with angle of incidence at Re = 5.8 x 10^6 when compared to other airfoil sections. The maximum thickness and leading edge radius of NACA 63_3-018 were large when compared to the other two airfoils and this made the transition to take place at minimum pressure point thereby increasing the lift. Laminar separation bubbles were seen in the other two airfoil sections thereby decreasing the lift values when plotted with respect to angle of incidence.

Experiments of Mueller et al [32] also showed increase in the lift values for Eppler-61 airfoil which has almost the same thickness as NACA 64A006 but is highly cambered. Sunada et al. [33] performed research on the different airfoil section characteristics by changing the parameters such as camber, thickness and roughness at a Reynolds number of 4 x 10^3. They deduced that low Reynolds number airfoils have less thickness when compared to airfoils with sharp leading edge at high Reynolds numbers. Optimal airfoils at this low Reynolds numbers

have a camber of about 5% and maximum camber occurs at mid-chord. They also found that leading edge vortices play a major role in deciding the characteristics of these airfoils.

The above theory states that streamlining greatly increases lift by reducing the steepness of the pressure rise and thickness is also one of the major factors effecting the separation.

While the above techniques seem like a sound idea, the end results are not always adequate as these methods are limited by the geometrical constraint of the airfoil. Therefore, other passive approaches were tried, such as passive suction and passive vortex generators. The idea of passive suction is to use a passive porous surface [34] [35] to mitigate the local pressure gradients and obviate separation to reduce drag. The vortex generators [36] use passive momentum adding to the near wall boundary to conquer the adverse pressure gradient, and this approach was widely used for airfoil flow control [37] [38] [39].

Passive methods have thus far reached a near saturation and further research does not seem to yield much improvement in the performance. Also, because passive methods are usually limited to certain working conditions, they can not be adjusted to work under wider conditions. Therefore active methods that can meet wider requirements have started to receive greater interest.

2.1.1.2 ACTIVE CONTROL

While passive methods have played an important role in the early years of flow control and will continue to do so, these methods are usually limited to certain working conditions and are not always the best way of controlling the flow field. This calls for more advanced methods of flow control, i.e. active flow control, where the control parameters change dynamically with the change in flow field to augment favorable flow control.

Suction, blowing, and synthetic jets are among the most common methods of active flow control techniques for high Re and for commercial and military aircraft. Morphing wings on the

other hand are more common for low Re regimes. As the current research deals with suction, blowing, and synthetic jets, some background information about these are presented here.

2.1.1.2.1 SUCTION AND BLOWING

Gu et al (1993) used leading edge suction on a delta wing to control the vortices. Experimental investigation of flow past a half delta wing at high angle of attack was performed using steady suction, steady blowing and alternate suction and blowing in the tangential direction along the leading edge of the wing. It was shown that this substantially retards the onset of vortex breakdown and stall. As a result of this type of control, the vortex structure in the crossflow plane is modified from a fully stalled condition to a highly coherent leading-edge vortex [40].

Saeed and Seliq (1996) presented a generalized multipoint method for the inverse design of airfoils with slot suction in incompressible potential flow. The design tool was validated against experimental data and was used interactively to perform rapid trade studies to examine the potential payoff for boundary-layer control as applied to the advanced-concept wings. Design changes in the airfoil were proposed as a result of slot injection [41].

Wright and Nelson (2001) conducted wind tunnel experiments to optimize distributed suction for laminar flow control. The experiments involved reducing the energy consumption to perform suction, without compromising on drag reduction. A large (2 m chord length and 1.6 m span) airfoil model was tested at various angles of attack. The effect of pressure gradient on the efficiency of suction was observed, and a relationship between transition and drag was also presented [42].

Wong and Konstantinos (2006) performed experimental investigation of spanwise blowing at different positions (0%, 25% and 100% of chord length) on a NACA 0012 airfoil. Lift, drag and pitching moment was measured for a range of angles of attack (from -20 degrees

to +20 degrees) and at *Re* 1.25×10^5. It was experimentally proved that lift was considerably improved as a result of blowing at 0.25c as compared to the baseline (no blowing) case [43].

Greenblatt and Wygnanski [44] provide an excellent review of the various periodic excitation methods, mainly steady suction and blowing. This review gives a detailed discussion of the mechanism and also the recent developments in the field. Previous reviews that provide a detailed discussion of the subject include Bushnell and McGinley [45], Fiedler and Fernholz [46], Gad-el-Hak and Bushnell [47], Moin and Bewley [48], and Gad-el-Hak [49].

2.1.1.2.2 SYNTHETIC JETS

Synthetic jets [50] have recently received a great deal of attention as a potential method for active flow control. Synthetic jets, in general, consist of an enclosed cavity with one side of the cavity having an opening or openings to the freestream flow. A synthetic, or zero-mass, jet derives its name from the total mass flow into and out of the cavity. During the first phase of the jet's operation, entrained fluid is drawn into the enclosed cavity. This same fluid is then expelled through the opening back into the freestream flow. Therefore, the net mass through the cavity opening is zero. However the net momentum transferred into the fluid is non-zero which enables flow control. Candidate designs of synthetic jets include piezoelectric ceramics [51], fluidics [52], and linear and rotary electromechanical motors [53]. Experimental studies [54], [55] and designs are actively carried by the Georgia Institute of Technology and Texas A&M University.

Synthetic jets have been actively applied to separation control to generate virtual shapes on solid walls. They can efficiently provide periodic forcing for dynamic separation control and completely suppress the separation by sufficient momentum injection when oscillating at higher levels. The applications of synthetic jets are numerous, such as shear flow control using fluidic actuator technology and aerodynamic flow control of bluff bodies using synthetic jet actuators.

The abilities of synthetic jets are so versatile that they also apply to other areas such as the mixing enhancement in combustion [20].

Perhaps the most influential work in synthetic jets has been performed at Georgia Tech by Glezer and colleagues. Their work was the first to characterize the basic performance of the synthetic jets and their ability to affect the flow over aerodynamic surfaces. Several papers [56], [57], [58] written by this group experimentally characterize the small-scale effects of synthetic jets. During their efforts, this group has employed several methods of experimentally measuring the flow field including phase-locked Schlieren imaging, hot-wire anemometry, and smoke visualization [59].

In addition to characterizing the performance of a single synthetic jet, Smith and Glezer investigate the performance of two adjacent synthetic jets [57]. Interestingly, they note that by phasing the timing of the jet actuation the direction of the resulting jet can be modified.

In spite of the work by Glezer and other researchers, synthetic jets still have not been exploited to their full potential. Similarly, although flow control using suction and blowing has been in use for quite some time now and there has been considerable research in the field, most of the suction research was concentrated on leading edge suction and the control parameters of the setup were decided based on design engineers intuition and experience. Likewise, with blowing jets, the majority of the research was concentrated on tangential trailing edge blowing and attempts to systematically select (optimize) the control parameters have not been extensively undertaken.

2.1.1.2.3 MORPHING WING

Morphing wing studies have been performed by various researchers in the past few years. This technique is most commonly used in regimes of low Reynolds number flights, such as the

Micro Aerial Vehicles (MAV's) and UAV's. A few recent examples of the application of morphing wing technique are discussed below.

Munday and Jacob (2002) experimentally investigated a wing with a conformal camber. The wing used an adaptive actuator mounted internally to alter the shape of the suction surface which resulted in a change in the effective camber by increasing the maximum thickness and moving the location of maximum thickness aft. They tested various oscillation frequencies at Reynolds numbers of 25,000 and 50,000 and several angles of attack. These oscillating modes showed a pronounced reduction in separation, hence the drag [60].

Kota et al. (2003) applied the morphing wing technology in designing morphing aircraft structures. Here, simple inputs are provided using actuators and the structures are deformed according to the input. In addition, these synthesis methods seek to optimize the stiffness of the structure to minimize actuator effort and maximize the stiffness with respect to the environment (external loading) [61].

Martin et al. (2005) performed experimental investigation of the technique. Using Combined Proper Orthogonal Decomposition (POD) and Linear Stochastic Estimation (LSE) technique, they developed flow induced vibrations on the wing of the micro aerial vehicle [62].

2.1.2 FLOW CONTROL AND CFD

The tremendous increase in CFD capability that have occurred as a direct result of increase in computer storage capacity and speed are transforming flow separation control from an empirical art to predictive science. Control techniques such as blowing/suction, morphing wings, and plasma actuators are all readily parameterized via viscous CFD. Current inaccuracies in turbulence modeling can severely degrade CFD predictions once separation has occurred; however, the essence of separation control is the calculation of attached flows, estimation of separation location, and indeed whether or not separation will occur. These tasks can in fact be

performed reasonably well via CFD within the uncertainties of the transition location estimation [14]. This latter uncertainty has been significantly reduced for low-disturbance freestreams and smooth surfaces using CFD [45].

In 1991, NASA in collaboration with various industries carried out substantial work in the field of supersonic laminar flow control (SLFC). The program utilized a balanced mix of computational efforts, ground facility experiments, and flight testing. Advanced Computational Fluid Dynamics methods and boundary-layer stability codes were used, which offered the opportunity to analyze flow phenomena to a greater level of accuracy than in the past. Swept-wing model experiments were carried out in a low-disturbance supersonic tunnel to provide data on leading-edge transition physics and flow mechanisms. Also, F-16XL-1 flight tests were performed using CFD to obtain laminar-flow data that will reduce the risk for the NASA experiments on the F-16XL-2. Flight tests on the F-16XL-2 provided attachment-line design criteria, code calibration data, and an improved understanding of the flow field over the wing that improved the design process for the suction panel [63].

A numerical study of blowing/suction type control (counting synthetic jets) mainly aims at qualitatively capturing the flow physics and the underlying control mechanisms. There are several different approaches from different perspectives. From the numerical methods perspective, some use RANS, and others use DNS or LES; from the computation geometry perspective, one could use 2-D grids or 3-D grids; and with respect to the simulation of membrane motion condition perspective, it could be either moving grid boundary, or directly applying velocity profiles at the boundary [20].

Kral et al. (1997) applied a 2-D RANS approach to solve a boundary value problem for the incompressible, unsteady 2-D Reynolds Averaged Navier-Stokes equations with the Spalart-Allmaras (SA) turbulence model. Their computational domain encompassed only the region

external to the jet, excluding the cavity or actuating membrane. The jet presence was simulated by forcing an analytical velocity profile on the boundary region corresponding to the jet orifice [64].

Rizzetta et al. (1999) numerically investigated the flowfields surrounding a synthetic-jet actuating device. A 3-D Direct Numerical Simulation (DNS) approach was used to solve the unsteady, compressible Navier-Stokes equations for both the interior of the actuator cavity and for the external jet flowfield. The external region, the cavity itself and the throat were calculated on separate grids and linked through a chimera methodology. The membrane motion was represented by varying the position of appropriate boundary points. These 3-D simulations showed that the internal cavity flow becomes periodic after several cycles. Therefore, it is appropriate for Kral et al. [64] to use the velocity profile as a boundary condition to simplify the computation [65].

Rumsey et al (2003) carried out CFD simulations and experimental validation of flow over a three-element McDonnell Douglas 30P-30N airfoil configuration at high lift. The experiment explores several different side-wall boundary layer control venting patterns, documents venting mass flow rates, and looks at corner surface flow patterns. The experimental angle of attack at maximum lift is found to be sensitive to the side-wall venting pattern: A particular pattern increases the angle of attack at maximum lift by at 2°. A significant amount of spanwise pressure variation is present at angles of attack near maximum lift. A CFD study using three-dimensional (3-D) structured-grid computations, including the modeling of side-wall venting, is employed to investigate 3-D effects on the flow. Side-wall suction strength is found to affect the angle at which maximum lift is predicted. Maximum lift in the CFD is shown to be limited by the growth of an off-body corner flow vortex and consequent increase in spanwise pressure variation and decrease in circulation. The 3-D computations with and without wall

venting predict similar trends to experiment at low angles of attack, but either stall too early or else overpredict lift levels near maximum lift by as much as 5%. Unstructured-grid computations demonstrated that mounting brackets lower the lift levels near maximum lift conditions [66].

At the University of Kentucky, Katam et al. (2005) used a modified NACA4415 with an adaptive actuator mounted internally. The camber of the airfoil could be changed in a static or oscillatory fashion. A series of simulations were performed in static mode for Reynolds numbers of 25,000 to 100,000 and over a range of angles of attack (AoA) and the characteristics of the flow separation and the coefficients of lift, drag, and moment were predicted. Preliminary simulations were performed for dynamic mode and it demonstrated a definitive ability to control separation across the range of Re and AoA. Numerical simulation results were compared with the previous experimental results which were performed on the airfoil in like flow conditions and these comparisons allowed determining the accuracy of both systems [12].

Since the proposed flow control setup explores the control parameters of steady and unsteady jets, some examples of CFD application to study control parameters is presented next, with focus on studies relating to different jet locations and angles of attack.

Wu et al (1998) studied control effects on a NACA 0012 airfoil using a Reynolds-averaged two-dimensional computation of a turbulent flow over an airfoil at post-stall angles of attack; they show that the massively separated and disordered unsteady flow can be effectively controlled by periodic blowing/suction near the leading edge with low-level power input. With a local unsteady forcing located at 5% from the leading edge, the angle of attack from 18^0 to 35^0 were tested using SA turbulence model approach [67].

Hassan et al (1998, 2005) studied the effect of an array of zero-mass `synthetic' jets on the aerodynamic characteristics of the NACA-0012 airfoil. Flowfield predictions were made using a modified version of the NASA Ames `ARC2D' (a 2-D RANS Baldwin-Lomax

turbulence model) unsteady, two-dimensional, compressible Navier-Stokes flow solver. Effects of the jet peak suction and blowing velocities, oscillation frequency, and jet surface placement on the time histories of the sectional lift, drag and moment were investigated for two angles of attack (0^0 and 5^0) and a free stream Mach number of 0.60 [69], [70].

Duvigneau (2006) proposed application of gradient based optimization algorithm to optimize the location of a synthetic jet on a NACA 0012 airfoil. Unsteady Reynolds-Averaged Navier-Stokes Equations (URANSE) were solved to simulate the flow over the airfoil (including the synthetic jet) at an angle of attack of 18^0 and at a Reynolds number of 2×10^6. It was numerically shown that maximum lift is generation when the jet is placed at 23% of the chord length [71].

All the above studies find that the synthetic jets and forcing/non-forcing (oscillatory/ steady) suction/blowing jets, when positioned on the airfoil leading edge can increase lift and decrease drag at certain angles of attack, but systematic studies of all the control parameters, such as jet angle, amplitude and frequency (in case of synthetic jets), are rarely performed. As mentioned at the end of previous section, CFD plays an extremely important role in successfully implementing such an approach.

2.2 OPTIMIZATION ALGORITHMS

Optimization occupies a fundamental position in engineering design and applications since the typical function of an engineer is to design better, more efficient and less expensive systems. The application of optimization methods to engineering problems requires the selection of the problem decision *variables* that are adequate to characterize the possible *individuals'* designs or operating conditions of the system, the definition of the *objective function* (fitness function) on the basis of which *individuals* will be ranked to determine the best solution and the definition of the *model* that describes the manner in which the problem variables are related and

the way in which the performance criterion is influenced by the by the variables. The problem's model normally includes a set of equality constraints, a set of inequality constraints, and some bounds for the variables. In its most general case, the optimization problem involves the determination of the optimal set of decision variables of a given objective function in the presence of some constraints. In the context of optimization, the *"best"* will always mean the individual set with either the maximum or minimum value of the fitness function.

Optimization problems could be categorized into many classes, depending on the linearity of the fitness function, the modality of the fitness function, the number of fitness functions, the availability of constraints, the number of decision variables, and the linearity of the constraints (discrete or continuous) [72]. A general optimization problem is usually described as a combination of these classifications. Optimization problems may also be classified into various categories according to the following:

- The use of some random components that test the solution space while the algorithm converges – deterministic or stochastic methodologies [73].
- The guarantee of the optimal solution obtained – exact or heuristic methods [74].
- The locality of the solution obtained. According to this classification, methods are classified as global or local techniques [75].

Local optimization techniques such as conjugate gradient methods, quasi-Newton methods, and simplex methods show great dependence on the initial guess and tend to be tightly coupled to the solution domain [75]. This tight coupling enables the local methods to take advantage of the solution space characteristics, resulting in relatively fast convergence to a local maxima or minima. However, the tight solution space coupling also places constraints on the solution domain, such as differentiability and/or continuity constraints. As a result, these methods are generally restricted to smooth and unimodal objective functions. It comes as no

surprise that these methods are unsuitable for a limited problem domain since real world research has to be fraught with discontinuities, multimodal, and noisy search spaces [76].

Exact techniques guarantee the optimal solution to a given optimization problem, while heuristic techniques seek optimal solutions without assuring either feasibility or optimality, or even in many cases to state how close to optimality a particular feasible solution is [74]. The distinguishing feature of a heuristic approach is the way they attempt to simulate some naturally occurring process. This idea of simulating natural processes has considerable value in solving complex engineering problems. Genetic Algorithms are formulated as an analogy to genetic structures and simulated annealing was in fact originally introduced as an analogy to thermodynamic processes, while tabu search finds some of its motivation in attempts to imitate intelligent processes by providing heuristic search with a facility that implements a kind of memory. Heuristics are rather more flexible and are capable of coping with more complicated and realistic objective functions and/or constrains than exact algorithms. They try to find an approximate solution of an exact model rather than an exact solution of an approximate model, as in the case of exact methods. Therefore, it may be possible to model real-world problem rather more accurately using heuristics than is possible if an exact algorithm is used.

Global optimization techniques such as Genetic Algorithms, simulated annealing and Monte Carlo are largely independent of and place few constraints on the solution domain [76]. This absence of constraints means that global methods are much more robust when faced with ill-behaved solution space. In particular, global techniques are much better at dealing with solution spaces having discontinuities, constrained variables, non-linear relations, or large number of dimensions with many potential local optimums. Global techniques yield either an optimum or near-optimum solution instead of a local optimum and often find useful solutions where local techniques fail. The downside though is that they either cannot or, at least usually,

do not take advantage of local solution space characteristics, such as gradients during the search process, resulting in generally slower convergence than local techniques.

Parallel and distributed computing plays an increasingly important role in computer science, engineering, and many other disciplines due to their ability in real time implementations of physical systems. Simulated annealing is a naturally serial algorithm, while Genetic Algorithms and Monte Carlo methods are of parallel nature [77]. This fact has caused a great deal of interest in these methods, especially Genetic Algorithms. Genetic Algorithms are considered more efficient as compared to Monte-Carlo and random walk methods as they provide faster convergence [78]. Also, Genetic Algorithms are particularly suitable for implementation under a parallel environment due to their inherent parallelism. Parallel Genetic Algorithms result in great improvements in terms of two commonly used performance measures: efficiency and speed [79]. This attribute of Genetic Algorithms can potentially provide remarkable speedups while retaining better solutions.

2.2.1 REVIEW OF GENETIC ALGORITHMS

Genetic Algorithms (GAs) were developed by John Holland and are modeled on the Darwinian concepts of natural selection and evolution [18]. In Genetic Algorithms, a population of potential solutions is caused to evolve towards a global optimal solution which occurs as a result of pressure exerted by the selection process and exploration of the solution space accomplished by crossover and mutation operators.

Genetic Algorithms are a relatively new class of optimization techniques, which are generating a growing interest in the engineering community. They are well suited for a broad range of problems encountered in science and engineering. As mentioned previously, GAs have performed efficiently in a number of diverse applications. The principal motives for researchers and practitioners in Genetic Algorithms are as follows:

- Limitations of traditional methods: The practitioners' motives in Genetic Algorithms are rooted in the limitations of traditional optimization and operations research methods. A certain optimization method is well tuned to a particular class of optimization problem, but when a problem comes along that violates the assumptions of such method, the solution results can be particularly disappointing.
- Method Investment: The wide spectrum of traditional narrow band algorithms implies that the practitioner should master a collection of techniques rather than a single broadly competent method. For example, for a linear problem with linear constraints, one can use linear programming. For a stage-decomposable problem, dynamic programming can be employed while for a non-linear problem with non-linear constraints, nonlinear programming can be utilized. The Genetic Algorithm, on the other hand, is an optimization procedure that works well over a broader class of optimization problems because the evolution of such a natural system takes place via mechanisms that are in many ways invariant across species.
- Model Investment. Method investment costs can be significant, but for many users the lion share of investment is tied up in modeling or simulation. Most complex optimization involves a fairly sophisticated objective function that may itself rely on models. Prior to using such models for optimization or design, users expend considerable time and effort inputting data and then using the models for analysis. After such a large investment in modeling, no user likes to be told that in order to perform an optimization, the model must be shoehorned into a form preferred by a particular optimization method; but many optimization methods require exactly this kind of model transformation. Genetic Algorithms, on the

other hand, take their function evaluations as they come, thereby respecting the significant investment that users may have in analyzing a model, using that model without substantial modification or transformation such as linearization. However, because GAs make relatively few assumptions about the solution space, and because the interface between GAs and evaluation involves only passing function evaluation values, a GA solution may require hundreds or thousands of function evaluations [72].

The construction of a Genetic Algorithm for the solution of any optimization problem can be separated into five distinct yet related tasks [76]:

1. The genetic representation of potential problem solutions.
2. A method for creating initial population of solutions.
3. The design of the genetic operators.
4. The definition of the fitness functions.
5. The setting of system parameters.

Each of the above components greatly affects the solution obtained as well as the performance of the Genetic Algorithm and the factors mentioned above have resulted in the availability of numerous variants of GAs reported in literature.

2.2.2 GENETIC ALGORITHMS AND CFD

For many years, flow computations have been taken into account by engineers to improve the designs, for example in aerodynamics or hydrodynamics. Modifications were performed manually first, and then using automated tools to lead the search towards optimality. Thanks to the progress in computational fluid dynamics (CFD) and computer hardware during the last few years, automated design optimization procedures are now expected to solve problems including complicated flows and realistic configurations. Consequently, the physical and geometrical

configurations encountered in industrial applications make necessary the recourse to viscous flow solvers based on sophisticated turbulence modeling, dealing with realistic geometries.

In this framework, the naive use of a standardized toolbox optimization software connected to a flow solver and an automated grid generator cannot be a sensible strategy, since the peculiarities of the flow solver should be taken into account. Otherwise, some limitations may be quickly encountered. For instance, the constraints on the grid linked to the use of near-wall turbulence models will have serious consequences on the mesh update procedure. If a particular strategy taking into account the high stretching of the volumes near the wall is not included, the mesh update will fail. This observation illustrates why the recourse to automatic grid generation software may not be wise in such a context.

Another reason is related to the mandatory use of parallelization strategies, such as domain decomposition, as soon as three-dimensional problems are considered. The mesh update, as well as the parameterization of the shape, should be adapted to this multi-block partition to work within each block independently and to send information for updating the overall domain. Otherwise, the parallelization becomes useless [81].

Concerning the optimization methods, the high computational costs implied by three-dimensional calculations, as well as the possible occurrence of numerical noise of various origins during the evaluations, should be taken into account for the choice of an optimization strategy. Finally, if differently connected software are employed in the design loop, some practical difficulties may arise, when distant computers are involved in the optimization process. All these remarks justify the development of an optimization procedure in which all numerical tools are adapted to the flow solver and integrated into a single code.

In the past, considerable research efforts were focused on the development of techniques for evaluating the sensitivity of the cost function with respect to the shape by using gradient-based optimization methods.

Jameson et al (1998) described the formulation of optimization techniques based on control theory for aerodynamic shape design in viscous compressible flow, modeled by the Navier-Stokes equations. Here, the Fréchet derivative of the cost function is determined via the solution of an adjoint partial differential equation, and the boundary shape is then modified in a direction of descent. The method was successfully used to design wings and wing-body combinations for long range transport aircrafts [82].

Anderson and Venkatakrishnan (1999) developed and analyzed a continuous adjoint approach for obtaining sensitivity derivatives on unstructured grids. A novel finite-difference gradients method was proposed for modifying inviscid and viscous meshes during the design cycle to accommodate changes in the surface shape [83]. The coupling between the optimizer and the flow solver is strong and a low number of evaluations are required to reach an optimal design. This approach was successful and cheap when rather simple flows were considered, but many limitations were noted when this approach was applied to more complex and realistic problems. First, the evaluation of the derivatives of the cost function with respect to the design variables is cumbersome when sophisticated flow solvers and highly non-linear turbulence models are considered [84]. Then, the presence of a numerical noise related to the complexity of the flow was reported [85], [86], which generates spurious local minima and inhibits the capabilities of gradient-based strategies. Moreover, using such methods, a local optimization is performed, involving only one criterion, which is not satisfactory in an industrial framework. Lastly, one may think that the difficulty in evaluating the derivatives when different physical

fields of applications are coupled makes quite unlikely the development of gradient-based multi-disciplinary optimization strategies.

To overcome these limitations, some authors proposed the employment of more powerful optimization strategies, such as Genetic Algorithms (GAs). These stochastic methods are known for their robustness, even when the cost function is noisy or discontinuous, and their ability to perform global optimization [76]. Moreover, they have the capability to solve multi-criteria problems.

One of the first attempts to systematically select the control parameters of suction and blowing setup were carried out by Huang [20]. Here, a Genetic Algorithm (EARND) was developed to optimize the various design parameters of a two-jet suction/blowing type flow control system on a NACA 0012 airfoil. It would be an extremely expensive and time consuming job to test all the configurations generated by the GA an using experimental setup. Also, it would be a tedious job to setup each case being investigated in the research experimentally. One solution to this problem is use of CFD to narrow down the solution space. In the next section we will discuss some previous applications of CFD to complex flow control setups which justify the use of CFD for current research.

Peigin and Epstein (2004) proposed an approach to the robust handling of non-linear constraints for GAs (Genetic Algorithms) optimization. A Real-coded GA was applied to aerodynamic shape design and care was taken (by reducing number of CFD computations) to reduce the overall computation time to optimize the parameter space [87].

Sengupta et al (2007) used GA in combination with CFD to optimally control incompressible viscous flow past a circular cylinder for drag minimization by rotary oscillation. At Re = 15000, using a flow solver with full 2D Navier-Stokes solver and fourth order Runge-

Kutta for time integration and a real coded GA, drag (a function of the maximum rotation rate and the forcing frequency of the rotary oscillation) was minimized [88].

However, a Genetic Algorithm in general does not use derivative information to lead the search. Therefore, a weak coupling between the optimizer and the flow solver is observed, yielding an expensive strategy which requires a high number of evaluations. Moreover, active flow control such as jet control, each individual requires large amount of computation time. Therefore, it is very important to design an efficient Genetic Algorithm, which has the capability of converging to the optimum or near optimum solution in minimum number of generations.

2.3 SUMMARY

In this chapter we discussed the previous work related to flow control, techniques of flow control, application of CFD and Genetic Algorithms to flow control and the need of current research. In chapter 3 we will discusses the conventional GA, briefly discuss the EARND GA and the detailed components of the approach used to develop another Genetic Algorithm, viz. a Continuous Genetic Algorithm (CGA).

CHAPTER – 3

3. GENETIC ALGORITHMS

3.1 OVERVIEW

Genetic Algorithms are based on principles inspired from the genetic and evolution mechanisms observed in natural systems. Their basic principle is the maintenance of a population of solutions to the problem that evolve towards the global optimum. They are based on the triangle of genetic reproduction, evaluation, and selection [76]. Genetic reproduction is performed by means of two basic genetic operators: crossover and mutation. Evaluation is performed by means of a fitness function that depends on the specific optimization problem. Selection is the mechanism that chooses parent individuals with probability proportional to their relative fitness. The selected individuals go through the mating process and mutation is performed on the resulting individuals.

GAs can be distinguished from calculus based and enumerative methods for optimization by the following characteristics [76]:

- GAs search for an optimal solution using a population of individuals, not a single individual. This important characteristic gives GAs much of their search power and also points to their parallel nature.
- GAs use only objective function information and no other auxiliary information is required. Much of the interest in Genetic Algorithms is due to the fact that they belong to the class of efficient domain-independent search strategies that are usually superior to traditional methods without the need to incorporate highly domain specific knowledge.
- GAs use probabilistic transition rules and not deterministic rules, in contrast with the calculus based and enumerative methods.

The population-based nature of Genetic Algorithms gives them two major advantages over other optimization techniques. First, it identifies the parallel behavior of Genetic Algorithms that is realized by a population of simultaneously moving search individuals or candidate solutions [76]. Implementation of GAs on parallel computers, which significantly reduces the CPU time required, is a major benefit of their implicit parallel nature. Second, information concerning different regions of solution space is passed actively between the individuals by the crossover procedure. This information exchange makes a Genetic Algorithm an efficient and robust method for optimization, particularly for the optimization of functions of many variables and nonlinear functions.

On the other hand, the population-based nature of Genetic Algorithms also results in two drawbacks. First, more memory space is occupied; i.e. instead of using one search vector for the solution, N_p search vectors are used which represent the population size. Second GAs normally suffer from a high computational burden when applied on sequential machines.

The fact that GAs use only objective function information without the need to incorporate highly domain-specific knowledge points to both the simplicity of the approach from one side and its versatility from the other. This means that once a GA is developed to handle a certain problem, it can easily be modified to handle any type of problems by changing the objective function in the existing algorithm. This is why Genetic Algorithms are classified as general-purpose search strategies.

In relation to the genetic representation of potential problem solutions, GAs are mainly classified into two categories: the binary-coded GA and the real-coded GA [80]. In a conventional binary-coded Genetic Algorithm, binary sub-strings corresponding to each design variable are stacked head to tail to yield a binary string that represents a particular design. In contrast, the real-coded Genetic Algorithm does not make use of the binary representation,

allowing for gene transformation operations to be conducted on the original real-valued representations of the design variable. A real-coded Genetic Algorithm is usually used to solve continuous design variables. While handling such optimization problems, real variants of the algorithm offer a number of advantages over binary-coded schemes, a few of which are listed below:

- It increases the efficiency of the GA.
- Less memory is required as efficient floating-point internal computer representations can be used directly.
- There is no loss in precision by discretization to binary or other values.
- There is greater freedom to use different genetic operators.

Existing crossover schemes are mainly divided into two categories: genotype crossover and phenotype crossover. In natural systems one or more chromosomes combining to form a total genetic prescription for the construction and operation of an organism are called the *genotype;* on the other hand, organisms formed by the interaction of the total genetic package with the environment are called the *phenotypes*. Genotype crossover is performed by swapping partial strings between the two parents. It is divided into three classes depending on the number of crossing points, single-point crossover, multi-point crossover, and uniform crossover schemes. The phenotype crossover is the well-known arithmetic crossover and is performed by interpolating the phenotype values of the two parents. Similarly, the mutation process has two main variants that include the genotype and phenotype mutations. For binary-coded GAs, the genotype mutation is the bitwise complement mutation operator. The phenotype mutation, on the other hand, has two variants that include static and dynamic mutation. The dynamic mutation is applied by performing random displacements for the selected variable from its original values, while the static mutation is carried out by assigning a completely new random value to the

selected variable. Dynamic mutation is particularly useful in fine-tuning the population in later stages of GA evolution when static mutation might cause too great perturbation and may lead to structures with very low fitness.

Real-coded GAs whose operators are applied at the variable level are useful when dealing with continuous optimization problems that do not have any requirement on the continuity and/or smoothness of the resulting solution curves, which is generally the case with most optimization problems. However, when dealing with continuous optimization problems that require the continuity and/or smoothness of the solution curves, the performance of such GAs will be poor. To solve this problem, Gutowski [89] introduced a special type of real coded GA, smooth Genetic Algorithms, which are well suited for continuous optimization problems with continuous and/or smooth solution curves. The main area of application of this algorithm is the reconstruction of unknown, continuous, and perhaps smooth distributions of various physical quantities derived from the experimental data. Smooth Genetic Algorithms as proposed by Gutowski depend on the evolution of curves in one-dimensional space.

The use of smooth Genetic Algorithms in continuous optimization problems with continuous and/or smooth solution curves needs some justification. First, the initialization phase in binary-coded GAs and other real-coded GAs result in neighboring variables that have opposite extreme values within the given solution range. This problem is overcome by the use of continuous curves that eliminate the possibility of highly oscillating values among the neighboring variables.

Second, the crossover operator in binary-coded GA results in a jump in the value of the variable in which the crossover point lies while keeping other variables the same or exchanged between the two parents. Other real-coded GAs whose crossover operator is applied at the variable level also result in an oscillatory behavior among neighboring variables. Smooth

Genetic Algorithms, on the other hand, result in smooth transition in the variable values during the crossover process.

Third, the mutation process in binary-coded GAs and other real-coded GAs change only the value of the variable in which mutation occurs while global mutations are required in such problems, which affect a group of neighboring variables. As a result, the operators of binary-coded GAs and the other real-coded GAs result in a step-function-like jump in the variable values, on the other hand, smooth Genetic Algorithms result in smooth transitions.

3.2 CONVENTIONAL GENETIC ALGORITHM

A conventional Genetic Algorithm as described by Salem [72] consists of the following steps (Figure 3.1):

1. ***Initialization:*** An initial population comprising of N_p individuals are generated in this phase at the genotype level by filling the bit strings randomly by 1 or 0 values. The coding process is then used to find phenotype values of the population.
2. ***Evaluation:*** The fitness, a nonnegative measure of quality used as a measure to reflect the degree of accuracy of the individual is calculated for each individual in the population according to its phenotype structure.
3. ***Selection:*** In the selection process, individuals are chosen from the current population to enter a mating pool devoted to the creation of new individuals for the next generation such that the chance of given individual to be selected to mate is proportional to its relative fitness. This means that the best individuals receive more copies in subsequent generations so that their desirable traits may be passed on to the offspring. This step ensures that the overall quality of the population increases from one generation to the next.

4. ***Crossover:*** Crossover provides the means by which valuable information is shared among the population. It combines the feature of two parent individuals to form two children individuals that may have new and possibly better phenotype structures compared to those of their parents and play a central role in the GA optimization process.
5. ***Mutation:*** Mutation is often introduced to guard against premature convergence. Generally, over a period of several generations, the gene pool tends to become more and more homogeneous. The purpose of mutation is to introduce occasional perturbations to the variables to maintain the diversity in the population. In conventional mutation operator, the bitwise complement mutation is applied at the gene level with some low probability of mutation, P_m. It is realized by performing bit inversion (flipping) on some randomly selected bit positions of offspring bit strings.
6. ***Replacement:*** After generating the offspring population through the application of the Genetic Algorithm operators to the parent population, the parent population is totally or partially replaced by the offspring population depending on the replacement scheme used. This completes the "life cycle" of the population.
7. ***Termination:*** Termination is defined by the user, it could be either the difference in fitness value of few subsequent generations or a fixed number of generations which the user thinks, would provide a reasonably acceptable solution.

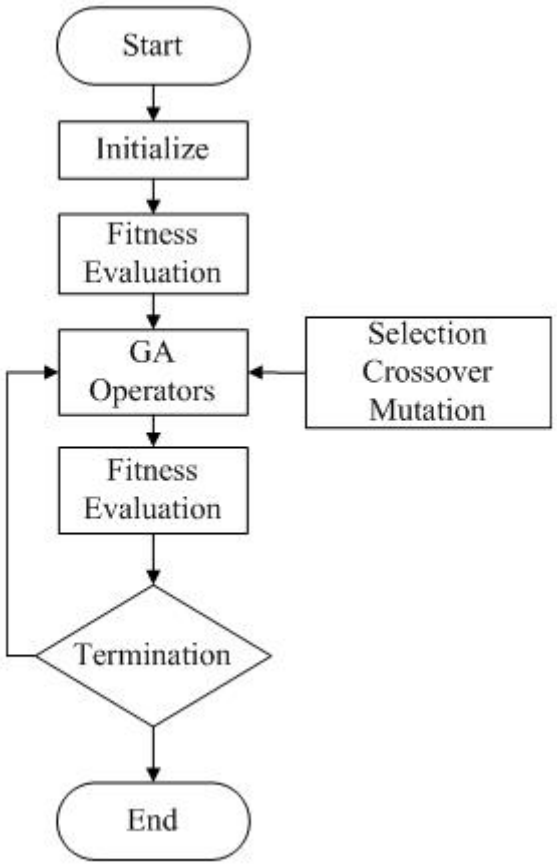


Figure 3.1: Process flow of Conventional flow chart.

3.3 EARND GENETIC ALGORITHM

The EARND GA was developed as part of the PhD dissertation [20] by Liang Huang. It is worth while to mention few important characteristics of the EARND GA here, as we use the EARND GA to compare our results obtained using the Continuous GA.

Figure 3.2 shows the flow process chart of the EARND GA. Some of the characteristics of this GA which make it an advanced GA are listed below:

- The traditional binary strings for the variables are replaced by real coded variable representation.
- Roulette wheel selection is employed for selection process.

- Crossover and mutation operators are employed and an advanced random number generation scheme is used for this process.
- The main advancement or change that was made to this GA was the use of normal distribution and explicit update of boundary along with a diversity control criteria which help in convergence and to maintain diversity of the solution space respectively.

A detailed description of this genetic algorithm approach is present in the dissertation [20], so the discussion here will focus on the key modifications [90] to the standard GA approach. Some key variables used in this description are:

- Number of total generations, *NGeneration*
- Number of individuals (population size) per generation, *NPopSize*
- Number of function variables (design parameters), *NVariable*

The first modification is that at selected generations (every *NUpdate* generations, where *NUpate* is set at 5-10% of *NGeneration*), the next generation will be born according to a normal distribution rule based on the statistics of a set of the previously-generated best individuals. The set size is equal to the number of individuals in each generation times the number of generations between the normal distribution generations plus one (*NUpdate* X *NPopSize*). As an example, consider the creation of an individual consisting of *NVariable* variables using the normal distribution approach. For the j^{th} variable of this new i^{th} individual, first randomly generate a value r_{ji}

$$r_j^i \in [0,1], \begin{matrix} 1 \le i \le NPopSize \\ 1 \le j \le NVariable \end{matrix} \tag{3.1}$$

Find the corresponding value p_{ji} as

$$r_j^{\,i} = \int_{-\infty}^{p_j^{\,i}} N(0,1)(z)dz \tag{3.2}$$

where $N(z)$ is the normal distribution. If μ_j and σ_j are the mean and deviation of the j^{th} variable calculated from the previous best (*NUpdate* X *NPopSize*) individuals, then the value of the j^{th} variable of the new i^{th} individual, x_{ji}, can be calculated as

$$x_j^{\,i} = \mu_j + \sigma_j \cdot p_j^{\,i} \tag{3.3}$$

The explicit adaptive range or boundary updates occur each *NUpdate* generation over the second 50% of the evolution. The boundaries for each variable design space are explicitly updated according to the statistics of the best (*NUpdate* X *NPopSize*) individuals up to that generation. The new ranges are chosen as

$$\begin{aligned} x_j^{Upper} &= \min(x_j^{oldUpper}, \mu_j + \kappa\sigma_j) \\ x_j^{Lower} &= \max(x_j^{oldLower}, \mu_j - \kappa\sigma_j) \end{aligned} \tag{3.4}$$

with the previous range boundaries designated by the old superscript. These new bounds are used until the next *NUpdate* generation is reached, at which point this process is repeated. Typically we use κ=5.0, which yields a conservative but robust searching process. The scale up factor, s_j, defines the precision of each variable and is likewise updated to maintain a consistent resolution with the new variable range

$$s_j = s_j^{\,old} - 0.5 \cdot \log_{10}\left(\frac{x_j^{\,oldUpper} - x_j^{\,oldLower}}{x_j^{\,Upper} - x_j^{\,Lower}}\right) \tag{3.5}$$

The diversity control is added to the selection function typically during the initial 20% of the evolution by adding a denominator *d* to the calculation of the scale fitness. The value of *d* is proportional to the number of similar individuals in that generation as determined by subdividing

the parameter space into cells and counting the number of solutions that fall into each cell. This penalizes a given individual's scale fitness if multiple individuals fall in the same section of the parameter space. If the search is to get the maximum (as opposed to minimum) fitness, then the fitness scale with diversity control for the birthing of the next generation is given by

$$fit_{scale} = \frac{1}{d} \frac{fit - fit_{\min} + \gamma}{fit_{\max} - fit_{\min} + \gamma} \tag{3.6}$$

Successful application of EARND GA with diversity control has been demonstrated previously in two-jet simulations [20].

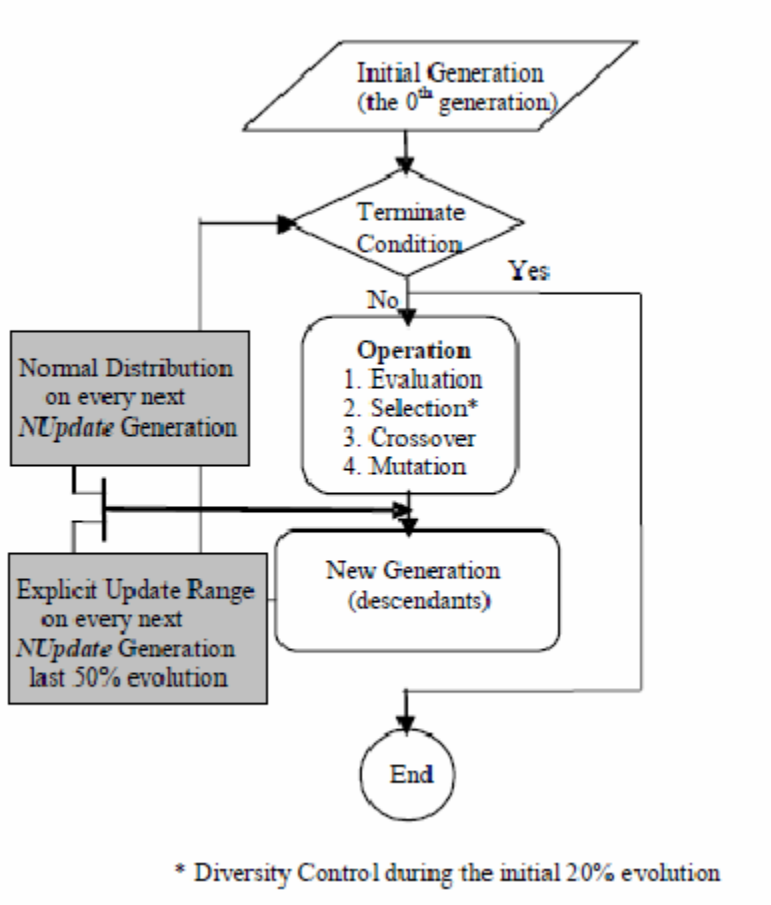


Figure 3.2: Process flow chart of the EARND GA [20]

3.4 CONTINUOUS GENETIC ALGORITHM

The continuous GA [21] is very similar to the conventional GA presented in the previous section. The primary difference of both CGA and EARND GA from the conventional GA is the fact that variables are no longer represented by bits of zeros and ones, but instead by floating-point numbers over the allowed range of the problem at hand. However, this simple fact adds some nuances to the application technique that must be carefully considered. In particular the GA operators i.e. crossover and mutation are different from the conventional GA.

The next obvious question is, why CGA instead of EARND GA? While the EARND GA is an advanced GA, it has some drawbacks; the most important one being early convergence. The EARND GA is forced to converge after a certain number of generations and it is possible that the GA has not found or is not in a region of optimum solution space by that generation, forcing it to converge to a local minima/maxima. Second, the CGA evolves in a manner where the most-fit individuals are carried unmodified to the next generation. This potentially reduces the number of simulations required since individuals that are carried unmodified from generation to generation do not have to be recalculated.

3.4.1 COMPONENTS OF CONTINUOUS GA

As previously stated, the goal of the GA is to solve the optimization problem at hand, where we search for an optimal solution in terms of the input variables. We begin the process by defining a chromosome as an array of variable values to be optimized. If the chromosome has N_{var} variables (an N dimensional optimization problem given by $p_1, p_2, ..., p_{N_{\text{var}}}$ then the chromosome is written as an array with $1 \times N_{\text{var}}$ elements so that,

$$chromosome = \left[p_1, p_2, ..., p_{N_{\text{var}}} \right] \tag{3.1}$$

In this case, the variable values are represented as floating-point numbers. Each chromosome has a cost function attached to it, found by evaluation the cost function f at the variables $p_1, p_2, ..., p_{N_{\text{var}}}$.

$$\text{cost} = f\left(\textit{chromosome}\right) = f\left(p_1, p_2, ..., p_{N_{\text{var}}}\right) \tag{3.2}$$

Equations (3.1) and (3.2) along with applicable constraints constitute the problem to be solved.

We will now consider a simple two variable continuous function minimization problem, to explain in detail the Continuous GA components [21].

Consider the cost function,

$$\begin{aligned} &\textit{function} = f(x,y) = x\sin(4x) + 1.1y\sin(2y) \\ &\text{Subject to the constraints: } 0 \le x \le 10 \text{ and } 0 \le y \le 10 \end{aligned} \tag{3.3}$$

Since f is a function of x and y only, the clear choice for the variable chromosome is

$$\textit{chromosome} = \left[x, y\right] \tag{3.4}$$

with $N_{\text{var}} = 2$. A contour map of the cost function is shown in Figure 3.3. This cost function, as evident from the figure, is a complex one, with peaks and valleys (color coded) of the cost function clearly evident in the contour plot. The plethora of local minima overwhelms traditional optimizing methods and our goal here is to find the minimum value of function $f(x,y)$.

3.4.1.1 VARIABLE ENCODING, PRECISION, AND BOUNDS

At this point of the GA process, we clearly see the differences and advantages of a continuous GA over a traditional GA. We no longer need to consider how many bits are necessary to accurately represent a value in binary. Instead, x and y have continuous values that fall between the bounds listed in equation (3.3). Although the values are continuous, a digital computer represents numbers by a finite number of bits. Therefore, when we refer to the continuous GA, we mean the computer uses its internal precision and roundoff to define the

precision of the value. Consequently, the algorithm is limited in precision to the roundoff error of the computer.

Since the GA is a search technique, it must be limited to exploring a reasonable region of variable space. Sometimes this is done by exploring a constraint on the problem such as equation (3.3). If one does not know the initial search region, there must be enough diversity in the initial population to explore a reasonably sized variable space before focusing on the most promising regions.

3.4.1.2 INITIAL POPULATION

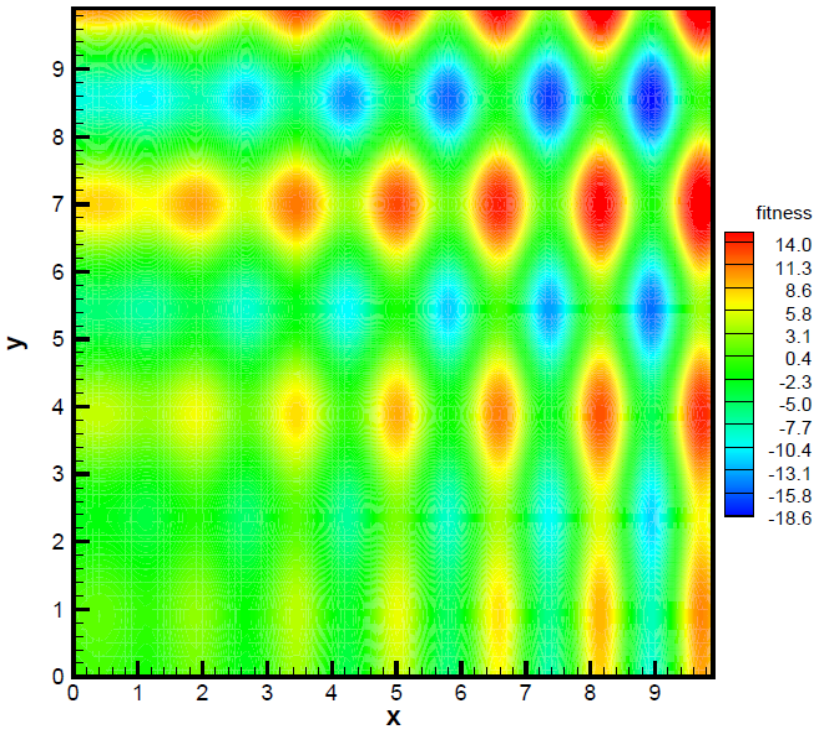
To begin the GA, we define an initial population of N_{pop} chromosomes. A matrix represents the population with each row in the matrix being a $1 \times N_{\text{var}}$ array (chromosome) of continuous values. Given an initial population of N_{pop} chromosomes, the full matrix of $N_{pop} \times N_{\text{var}}$ random values is generated by using a random number generator. Most random number generators generate values which are normalized to fall between 0 and 1. But the variable values of 'unnormalized', and thus the random numbers need to be altered according to the bounds or range. In general, the values could be unnormalized using the following relation,

$$p = \left(p_{hi} - p_{lo} \right) p_{norm} + p_{lo} \tag{3.5}$$

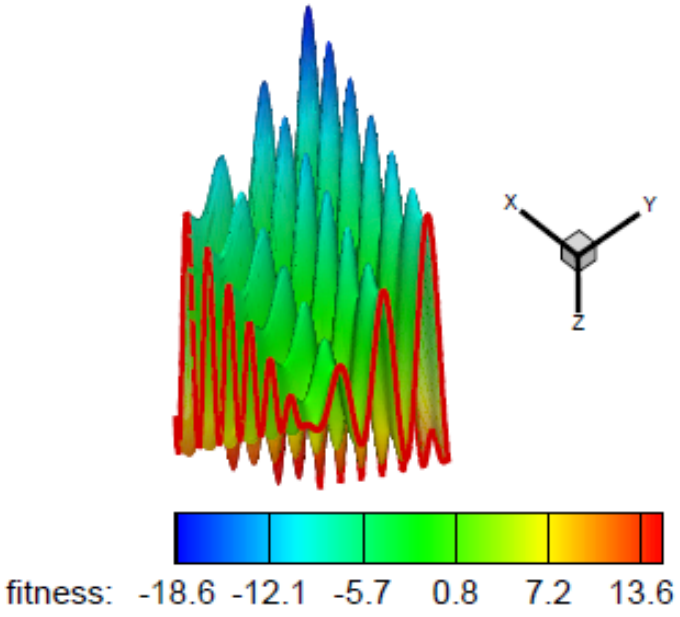
where p_{hi} = highest value of the variable range

p_{lo} = lowest values of the variable range

p_{norm} = normalized value of variable



(a)



(b)

Figure 3.3: Contour plot of the example problem

For the current example, the unnormalized values are just $10\,p_{norm}$. Chromosomes are now passed to the fitness function to evaluate the cost of each chromosome. Table 3.1 lists in the order of fitness, the initial population ($N_{pop} = 8$) and fitness associated with each of the chromosomes and in Figure 3.4 the fitness values are marked by '+'.

Table 3.1: Initial population arranged according to fitness

x	y	Fitness
7.636937	8.786544	-15.077730
9.125283	5.202538	-13.252140
8.735126	8.786544	-12.513817
7.732671	8.232512	-9.826081
5.789647	5.000474	-8.321793
0.840779	8.981157	-7.844455
0.671330	8.981157	-7.363932
8.371161	5.340574	1.768091

3.4.1.3 NATURAL SELECTION

After evaluating the fitness of all the chromosomes, we have to decide which chromosomes are fit enough to survive and possibly reproduce offspring in the next generation. To do this, the N_{pop} costs and associated chromosomes are ranked from lowest cost to highest cost (Table 3.1) and the bottom 50% of the chromosomes is discarded. From the remaining chromosomes ($N_{keep} = 4$), top 50% are selected for reproduction, which go through the crossover process to generate the offspring's, i.e. a crossover percentage (P_c) of 50%. Therefore, in the

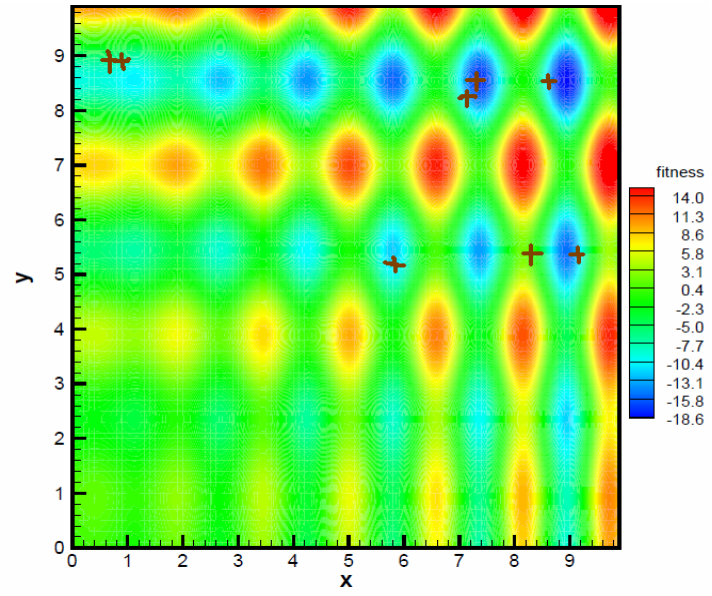


Figure 3.4: Contour plot showing the initial population

current example, the chromosomes which survive the selection process are listed in Table 3.2.

Table 3.2: Chromosomes which survived the selection process

x	*y*	Fitness
7.636937	8.786544	-15.077730
9.125283	5.202538	-13.252140
8.735126	8.786544	-12.513817
7.732671	8.232512	-9.826081

3.4.1.4 PAIRING

The chromosomes in Table 3.2, i.e. the most-fit chromosomes form the mating pool, make up for two mothers and two fathers pair in some random fashion. Each pair produces two offspring that contain traits from each parent. In addition the parents survive to be part of the next generation. There are several approaches for pairing the chromosomes, such as top-down pairing, random pairing, tournament selection, rank weighting, and cost weighting. The one used in the current research is ‘cost weighting’.

The cost weighting is a form of weighted random paring in which the probability assigned to the chromosomes in the mating pool depends on their cost. As the example problem

is a minimization problem, the probability is inversely proportional to the cost, i.e. the lowest cost has the greatest probability of mating, while the chromosome with the highest cost has the lowest probability of mating. A random number determines which chromosome is selected. A normalized cost is calculated for each chromosome by subtracting the lowest cost of the discarded chromosomes $\left(C_{N_{keep}+1}\right)$ from the cost of all the chromosomes in the mating pool:

$$C_n = C_n - C_{N_{keep}+1} \tag{3.6}$$

Doing this ensures that all the costs are negative and the probability P_n is calculated using

$$P_n = \left| \frac{C_n}{\sum_{m}^{N_{keep}} C_m} \right| \tag{3.7}$$

This approach tends to weight the top chromosome more when there is a large spread in the cost between the top and bottom chromosomes. On the other had, it tends to weight the chromosomes evenly when all the chromosomes have approximately the same cost.

3.4.1.5 MATING/CROSSOVER

The mating process begins by randomly selecting a variable in the first pair of the parents to be the crossover point

$$\alpha = roundup\{random\_number \times N_{\text{var}}\} \tag{3.8}$$

Let,

$$\begin{aligned} parent_1 &= \left[p_{m1}\ p_{m2} \cdots p_{m\alpha} \cdots p_{mN_{\text{var}}} \right] \\ parent_2 &= \left[p_{d1}\ p_{d2} \cdots p_{d\alpha} \cdots p_{dN_{\text{var}}} \right] \end{aligned} \tag{3.9}$$

where, the *m* and *d subscripts* discriminate between the mom and the dad parent. Then the selected variables are combined to form new variables that will appear in the children:

$$p_{new1} = p_{m\alpha} - \beta\left[p_{m\alpha} - p_{d\alpha}\right]$$
$$p_{new2} = p_{d\alpha} + \beta\left[p_{m\alpha} - p_{d\alpha}\right] \tag{3.10}$$

where, β is also a random value between 0 and 1. The final step is to complete the crossover with the rest of the chromosome:

$$offspring_1 = \left[p_{m1}\ p_{m2} \cdots p_{new1} \cdots p_{dN_{\text{var}}}\right]$$
$$offspring_2 = \left[p_{d1}\ p_{d2} \cdots p_{new2} \cdots p_{mN_{\text{var}}}\right] \tag{3.11}$$

If the first variable of the chromosomes is selected, then all the variables to the right of the selected variable are swapped. If the last variable of the chromosomes is selected, then all the variables to the left of the selected variable are swapped. This method does not allow the offspring variables outside the bounds set by the parent unless $\beta > 1$.

For the example problem, in this generation, chromosome 2 and 4 are selected for pairing, i.e.

$$chromosome_2 = \left[9.125283, 5.202538\right]$$
$$chromosome_4 = \left[7.732671, 8.232512\right]$$

A random number generator selects p_1 as the location of the crossover. The random number selected for β is $\beta = 0.027241$. The new offspring are given by

$$offspring_1 = \left[9.125283\text{-}0.027241\left(9.125283\text{-}7.732671\right),\ 8.232512\right]$$
$$offspring_2 = \left[7.732671\text{+}0.027241\left(9.125283\text{-}7.732671\right), 5.202538\right]$$

3.4.1.6 MUTATION

Mutation is an important GA operator which makes sure that the search does not get stuck in local minima. If care is not taken, the GA can converge too quickly into some specific region of fitness surface. If this area is a region of global minima, it is good; but this does not

happen often. For example, in some functions, such as the one we have considered in this section, have many local minima and the GA will most likely to get stuck in one of those. To avoid this problem of overly fast convergence, we force the routine to explore other areas of the cost surface by randomly introducing changes, or mutations, in some of the variables.

To introduce mutation, we first decide how many variables we want to mutate. This depends on the mutation rate. In the current example, we use a mutation rate (P_m) of 20%, i.e. 3 variables are mutated each generation. This could be calculated as below:

$$\begin{aligned} Number\ of\ mutations &= P_m \times N_{pop} \times N_{\text{var}} \\ &= roundoff\,(0.2 \times 8 \times 2) = 3\ mutations \end{aligned}$$

Next random numbers are generated to select the row and column of the variables to be mutated. The random numbers are then scaled accordingly, as we did while generating the initial population. Once mutated and scaled, the fitness function is recalculated using the new variable values.

For the example problem, at the end of all the genetic operations, the set of chromosomes that become the input to the next generation are shown in Table 3.3.

Table 3.3: Variables and fitness at the end of 1st generation

x	*y*	Fitness
0.283255	8.782450	-9.011352
5.789647	5.000474	-8.321793
9.125283	5.202538	-13.252140
8.735126	8.786544	-12.513817
1.493311	8.386663	-8.527298
9.183612	6.413751	-5.726168
0.840779	8.981157	-7.844455
8.877505	8.975071	-14.963850

3.4.1.7 THE NEXT GENERATION

The process described is iterated until an acceptable solution is found. For our example, the GA is run until the change in the mean and standard deviation of all the fitness values

between two consecutive generations is less than 6×10^{-6}, which is the convergence criteria. Table 3.4 lists the variables and the fitness values after convergence and Figure 3.5 shows the contour plot with the fitness values of the final population (circled).

Table 3.4: Variable and fitness values after convergence

x	*y*	Fitness
9.039025	8.673142	-18.554251
9.028314	8.614508	-18.491527
9.009554	8.677984	-18.490353
9.009554	8.614508	-18.437245
9.051708	8.786544	-18.273391
9.013859	8.951057	-16.984275
8.808161	8.786544	-14.754626
9.075129	3.124597	-9.057720

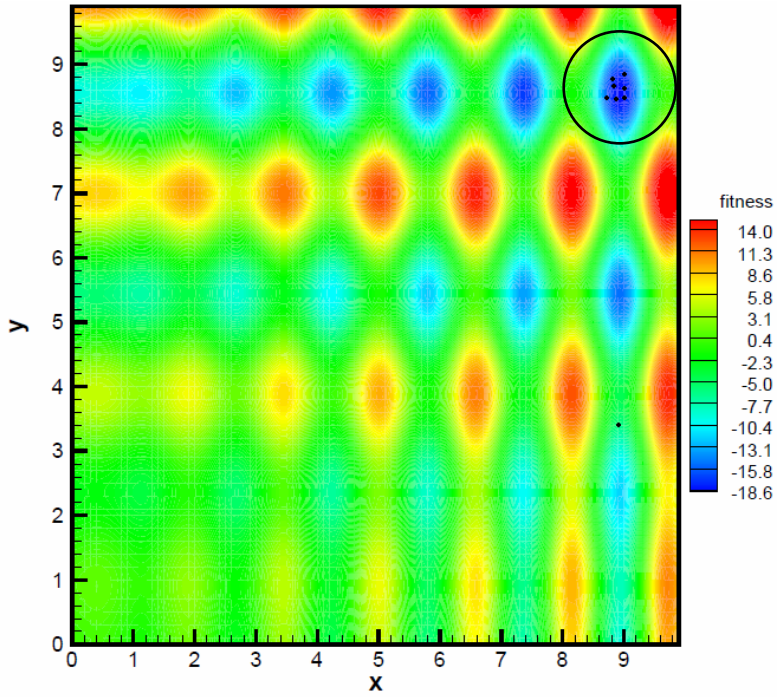


Figure 3.5: Contour plot showing region of high fitness (minimum cost) in blue

It is evident from the plots and the table of converged solutions that the GA was successful in locating the region of optimum solution and also the near optimum values of the variables.

3.5 SUMMARY

In this section we explained various GA techniques concentrating on the Continuous GA. Various components of the CGA were discussed and the ability of CGA to find the region of optimum solution of a complex optimization problem is successfully demonstrated. We will now apply this GA and CFD to the flow control problem; results of this application are discussed in chapters 5 and 6.

CHAPTER – 4

4. COMPUTATIONAL TOOLS

Computational tools are the basis of this research therefore, it is important to understand what is behind each of these codes. In this chapter we will discuss about the grid generation process using two in-house computer codes (Flexgrid.f90 and g.f90) followed by a discussion of the CFD code (GHOST) and the GA-CFD system. Lastly the computational platforms (commodity clusters) used to run the simulations are discussed. The process is primarily divided into four steps. First the experimental setup is modeled using a two dimensional grid using the grid generation code (Flexgrid.f90). Second the grid data is converted into the format required by the CFD code to solve the flow field equations by using a second code called g.f90. Here we also introduce the boundary conditions that govern the flow field. Third the flow field is solved used the CFD code GHOST and the lift and drag data is generated. Lastly, using the lift and drag data, the Genetic Algorithm computes the fitness and based on the fitness values generates the next set of configurations. In the current research a combined total of more than 9000 simulations of the steady and unsteady cases have been performed to achieve the optimum solution.

4.1 GRID GENERATION

The grid generation method is a two step process. First, we generate a full two dimensional grid (background and airfoil) using FlexGrid.f90 and then we use g.f90 to convert the grid data files into generalized coordinate system and also add the boundary conditions. The next two sections discuss the process in detail.

4.1.1 FLEXGRID.F90

Flexgrid.f90 is an in-house grid generation tool which was originally developed by Dr. Liang Huang and Dr. P.G. Huang at the University of Kentucky and is capable of generating two-dimensional structured grids.

The grid constructed for the four jet case consisted of 16 two dimensional, multi-zonal blocks (Figure 4.1 and Table 4.1) of which 11 are background blocks and 5 airfoil blocks. The NACA0012 airfoil block (block # 12 to 16) overlaps on three background blocks (block # 2 to 4) at the center of the grid. These blocks are surrounded by 8 peripheral blocks (block # 1, 5 and 6 to 11).

The dimensionless outer boundary is chosen as,

$$A_H \times A_W = 12c \times 8c = 12 \times 8 ,$$

large enough to prevent the outer boundary from affecting the near flow field in the vicinity of the airfoil. Previously [20], extensive grid dependence studies of the basic grid setup has been performed and validated.

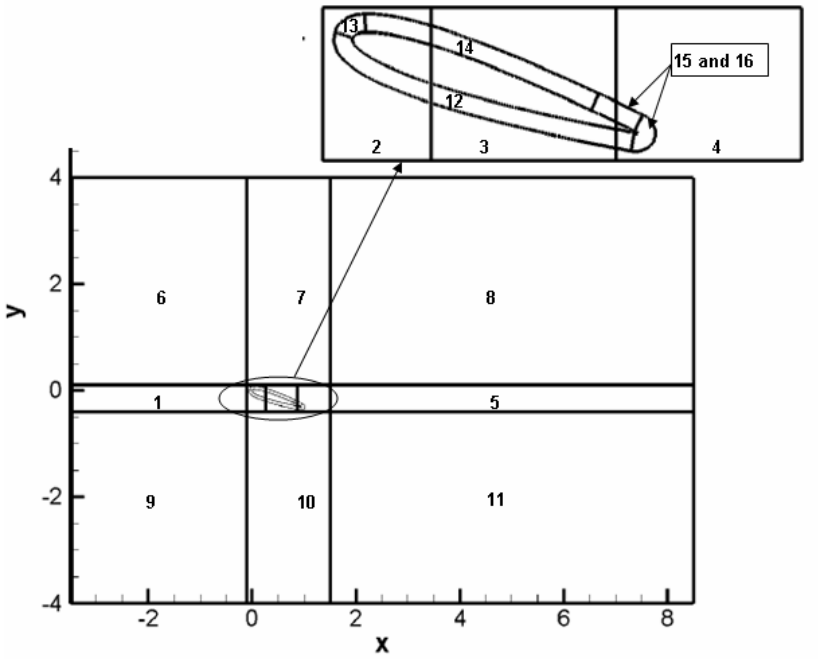


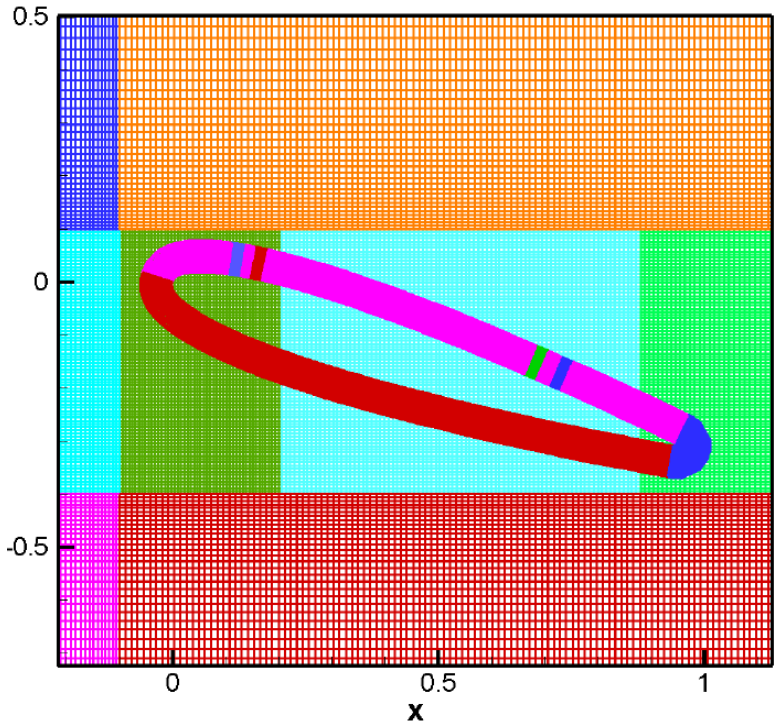
Figure 4.1: Multi-zone grid setup

The airfoil block in the current research is different form the one used in the case of two-jet optimization system. In the case of two-jet setup, the jet blocks were generated exclusively in each simulation. This approach is not only time consuming (considering the long GA runs), but

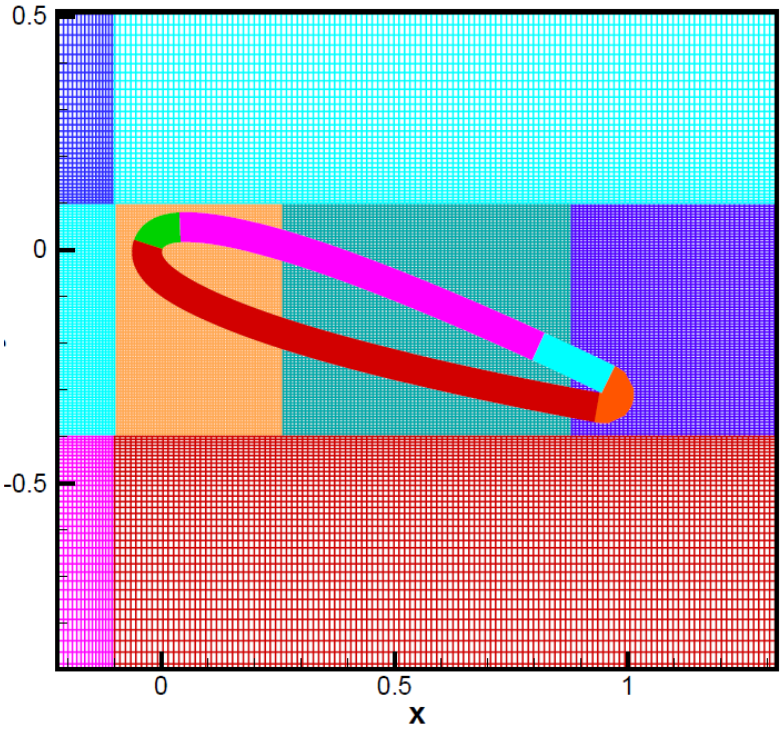
also is very cumbersome to use with an array of jets. For example, even with four jets the coding process of implementing the jets without the jet blocks being overlapped is an extremely complicated process, not to mention the additional effort that is required to generate the intermediate blocks. Considering this difficulty is particularly difficult to adapt this type of grid in a setup where we intend to extend to an array of jets. Hence, the airfoil block was modified such that the grid resolution on the top portion of the airfoil, where the jets are typically placed (5% to 80% of chord length) is made equal to the grid resolution required by the jets (Figure 4.2). With this kind of a grid setup we can easily place the suction and blowing jets along the available length by simply varying the boundary conditions at that location. Also, we can increase or decrease the number of jets or have an array of jets, as this no longer requires any change in grid generation.

The grid used in the case of unsteady synthetic jet setup is again modified. This is because the jet width of the oscillatory jets are much smaller than the steady jets and a fine grid spacing (as in steady four jet case) on the whole upper block would be computationally very expensive. More details of this grid setup are discussed in Chapter 6.

Flexgrid is also hardwired with the boundary information. A file called ‘input’ is generated using this information and is used by G.f90 to generate the boundary condition in each of the grid block data files. A brief description of this ‘input’ file is presented in a later part of this chapter.



(A)



(B)

Figure 4.2: A: old two-jet grid (multi jet blocks), B: new grid (single jet block)

Table 4.1: *i* and *j* points of the 16 grid blocks

Block Number	***i points***	***j points***
Background 1	61	84
Background 2	78	84
Background 3	63	84
Background 4	110	84
Background 5	110	84
Backupper 6	110	100
Backupper 7	140	100
Backupper 8	110	100
Backlower 9	110	100
Backlower 10	140	100
Backlower 11	110	100
Lower 12	474	50
LeaUpper 13	35	50
MidUpper 14	799	50
TraUpper 15	42	50
Tra 16	40	50
Total Grid Points = 176948		

4.1.2 G.F90

GHOST, the CFD code used in this research requires that the input data be in generalized coordinate system. This process of converting the grid data files generated by Flexgrid.f90 to generalized coordinate system and adding other physical (boundary) information is performed by the code, g.f90 which was originally developed by Dr. P. G. Huang. The contents of the grid file for a non moving grid are as follows [91]:

- Number of grid points in the *x* and *y* direction.
- Number of ghost points.
- Grid point weight in the *x* and *y* direction
- *x* & *y* co-ordinates of the grid points.
- Volume of the cell surrounding each grid point.
- Distance between the wall and the grid point.

- Values for the various transformation functions such as η_x, η_y, ξ_x *and* ξ_y .
- A variable called "*inx*" which specifies if a particular grid point is a ghost point. If the value of *inx* for a grid point is 1, then that grid point is treated as a ghost point, whereas when its zero, it is treated as a normal point.
- Boundary conditions

4.1.3 INPUT FILE – "input"

As mentioned in the previous section, g.f90 is used to generate the grid data required by GHOST. In order for g.f90 to generate a grid it requires certain data regarding the size of the computational grid, boundary conditions and number of grid points. This data is provided using the file called "input", which is generated by Flexgrid.f90. An input file used to generate the multi-zonal grid is presented in Appendix A2.

4.2 GHOST

The computations were carried out using the CFD code, GHOST. GHOST is an in-house CFD code originally developed at the University of Kentucky by Dr. P. G. Huang. This code has been tested extensively and is routinely used for turbulence model validation [92] [93] [94]. The code has also been used to generate published flow control results such as the suction/blowing on NACA0012 [3], morphing wing [12], and plasma actuator [95] flow control setups. It is a two-dimensional incompressible finite-volume structured formulation, computational fluid dynamics code with chimera overset grids for parallel computing. The QUICK and TVD schemes are applied to discretize the convective terms in the momentum and turbulence equations, respectively. A central difference scheme is used for the diffusive terms and the second order upwind time discretization is employed for the temporal terms. The code employs a variety of Reynolds-Averaged Navier Stokes Turbulence models. The turbulence model used in the present computation is Menter's SST two-equation model [94], which provides excellent predictive capability for flows with separation [97]. The multi-block and chimera features allow the use of

fine gird patches near the jet entrance and in regions of highly active flow. GHOST also employs MPI parallelization to allow different computational zones to be solved on multiple processors [98]. Simulations have been performed on a variety of computer architectures which will be discussed in the next section.

The governing equations for unsteady incompressible viscous flow under the assumption of no body force and heat transfer that are used to calculate the various flowfield parameters in GHOST are as below:

Conservation of Mass

$$\frac{\partial}{\partial t}\int_V \rho\, dV = -\oint_S \rho u_i n_i dS \tag{4.2}$$

Conservation of Momentum

$$\frac{\partial}{\partial t}\int_V \rho u_j dV = -\oint_S \rho u_i n_i\, u_j dS - \oint_S p n_j dS + \oint_S \tau_{ij}\, n_i dS \tag{4.3}$$

Conservation of Energy

$$\frac{\partial}{\partial t}\int_V \rho E dV = -\oint_S \rho u_i n_i\, E dS - \oint_S p u_j n_j dS + \oint_S u_j\, \tau_{ij} n_i dS \tag{4.4}$$

where ρ is density, p is pressure, u_i are the components of the velocity vector, n_i is unit normal vector of the interface, τ_{ij} is tensor of shear force, and specific internal energy is $E = e + \frac{1}{2}(u^2 + v^2 + w^2)$ [89].

Flow and geometry data in GHOST for a given grid or subgrid are stored in individual arrays, as in $\phi_1(i,j), \phi_2(i,j), ..., \phi_n(i,j)$. On a given grid, GHOST performs the majority of its calculations as a series of i,j bi-directional sweeps in nested double loops. In brief, the momentum equations are solved implicitly in a delta form, shown here for the time discretization in one dimension:

$$\frac{3(\Delta\phi)^m}{2\Delta t}+\frac{\partial f(\Delta\phi)^m}{\partial x}=\frac{(\phi^n-\phi^{n-1})}{2\Delta t}-\frac{3((\phi^{n+1})^m-\phi^n)}{2\Delta t}-\frac{\partial f((\phi^{n+1})^m)}{\partial x} \tag{4.1}$$

where ϕ represents any variable, m is the subiteration level, and n is the time iteration level. The right-hand side of Eq. (4-1) is explicit and can be implemented in a straightforward manner to discretize the spatial derivative term. The left-hand side terms are evaluated based on the first order upwind differencing scheme. The deferred iterative algorithm is strongly stable, and the solution ϕ^{n+1} is obtained by using inner iterations to reach the convergent solution of the right-hand side of Eq. (4-1), corresponding to $\Delta\phi$ approaching zero. At least one subiteration is performed at every time step so that this method is fully implicit.

The resulting matrices generated at each subiteration based on the QUICK and TVD schemes as well as evaluation of source/sink terms are solved with ADI-type decomposition into a pair of sweeps alternately in the i and j-directions which are solved sequentially in tri-diagonal matrices. This sequence may be repeated for improved accuracy. The Rhie and Chow technique [99] is then used to extract the pressure field from the continuity equation.

4.3 GA-CFD SYSTEM

The implementation of the genetic algorithm is designed for commodity clusters or similar architectures, although with modification it may be used on any system. The code is run from a server node on which a specified number of genomes from the current generation are selected, the grids generated, and then the required data is transferred into a series of directories, one corresponding to each genome. These directories are then distributed among the designated set of nodes for the CFD evaluation of a given genome. The CFD computation may be accomplished on a single or multiple processors, depending on the choice of the user. Once the computation is completed, the resultant fitness data is collected by the server node while simulation details not needed for the genetic search process are stored for future reference. Sets

of genomes are similarly simulated until the full generation is completed; then, the server node applies the genetic search algorithm to generate a new generation and the process is repeated until the desired full evolution is complete. The basic architecture of the system was developed as part of the previous two jet simulation [20]. In the current research, it was modified to accommodate the Continuous GA and after each full generation the Continuous GA is executed to generate the next set of individuals. The original system incorporated various genetic codes and files which was a reasonably complicated system. It was not robust enough to easily accommodate the changes in the design parameters. On the other hand with the implementation of the CGA, any changes in the design parameters (e.g. increasing number of jets) could be easily implemented by updating the CGA, thus making the system more robust than before.

4.4 COMPUTATIONAL PLATFORMS – KENTUCKY FLUID CLUSTERS

The computer platforms used for the simulations in this research are Kentucky Fluid Clusters (KFC). Various versions of KFC's have been constructed at the University of Kentucky, and those used in the current research are KFC5, KFC6A and KFC6I.,

KFC5 was built in the summer of 2005 by the UK-CFD group at the University of Kentucky. It consists of 47 nodes on a single Gigabit switch with AMD64 3200+, 2.01GHz processors. Each node is mounted with 512MB of physical memory and the processors have a L2 cache of 512KB each. The computation time of the current four-jet simulations was ~9.0 hrs per simulation with one simulation on one node.

KFC6A and KFC6I were built in the fall of 2006. KFC6I is built with Intel processors. It consists of 24 nodes networked using a single Gigabit switch. The processors on these nodes are Intel E6400 family, which is a 64bit dual core 2.13 GHz processor; the nodes are mounted with a physical memory of 1GB each. The main advantage of this cluster over the dual core AMD cluster is the L2 cache; the Intel's have a L2 cache of 1MB per core, which boosts the

performance relative to AMD when memory intensive CFD codes are run. Also, the dual core capability of this cluster is operational; this increased the power by two fold. The simulation time on the dual core Intel processors for the four-jet case is 9.0 hrs per two simulations, with one simulation on each core of a node. This is achieved by exploiting the dual core capability of the processor.

KFC6A has 23 nodes; again on a single Gigabit switch with AMD x64 4600+, 2.40GHz processors (dual core). It is equipped with a main memory of 1GB on each node and a L2 cache of 512KB on each core. Computation time of the four-jet case on this is little less than KFC5, ~8.5 hrs per simulation with one simulation on one node. It should be noted that the dual core capability of these processors is still not tested.

KFC6I KFC5

Figure 4.3: KFC6-I and KFC5

4.5 SUMMARY

In this chapter we discussed about the various codes that were used to perform the simulations of current research. A brief description of the Kentucky Fluid Clusters was also presented and a comparison of the simulation time for the four-jet case was shown. Chapter 5 presents the basic case setup procedure and results of the steady four-jet simulation obtained using both the CGA and the EARND GA. This also includes a comparison of the CGA and the EARND GA results. As the focus of current research is the Continuous GA, detailed analysis of this GA approach is presented. We will also discuss the possibility of improving the aggregate fitness by combining both the GA approaches.

In chapter 6, results of the unsteady two synthetic-jet results are presented. It begins with a discussion of the modifications made to the airfoil grid to accommodate the relatively narrow synthetic jets, followed by a detailed discussion of the case setup and the jet configurations obtained using the Continuous GA.

CHAPTER – 5

5. STEADY FOUR JET RESULTS

The flow under consideration is over a NACA0012 airfoil at an 18^o angle-of-attack and a Reynolds number (*Re*) of 500,000. Given the relatively large *Re* a fully turbulent flow with no transition is assumed for the computation. Because the focus of the current investigation is the control of the flow separation through blowing and suction jets, an incompressible Navier-Stokes solver (GHOST) is used to eliminate possible additional uncertainties caused by compressibility effects. The NACA0012 airfoil is placed at 18^o relative to the freestream, resulting in a strongly separated flow when there is no flow control.

5.1 GRID AND BOUNDARY CONDITIONS

As previously stated, the basic two-dimensional grid consists of 11 background blocks in a three-by-three pattern, the central region consisting of three subgrids over which the airfoil grid is placed (Figure 5.1-a). The airfoil grid consists of 5 blocks; these blocks are the most refined blocks, where fine grid spacing is employed relative to the background blocks. An even finer grid spacing (equal to that required for a jet) is employed on the upper block of the airfoil, (5% chord to 80% chord), where the jets are typically placed.

On the outer boundary, the left (inlet) boundary is fixed with a uniform dimensionless inlet velocity of unity, the upper and lower boundary condition are “freestream” boundaries which satisfy the Neumann condition, and the right (outflow) boundary condition is set to a zero velocity gradient condition. For the airfoil blocks, the inner boundary condition is a no-slip wall boundary condition, and the outside boundary is set to “overlap” which allows the background grid points being overlapped by the airfoil block grid points to interpolate values from the foreground airfoil grid points. The jet inlet velocity for the jets in general is given by eqn 5.1.

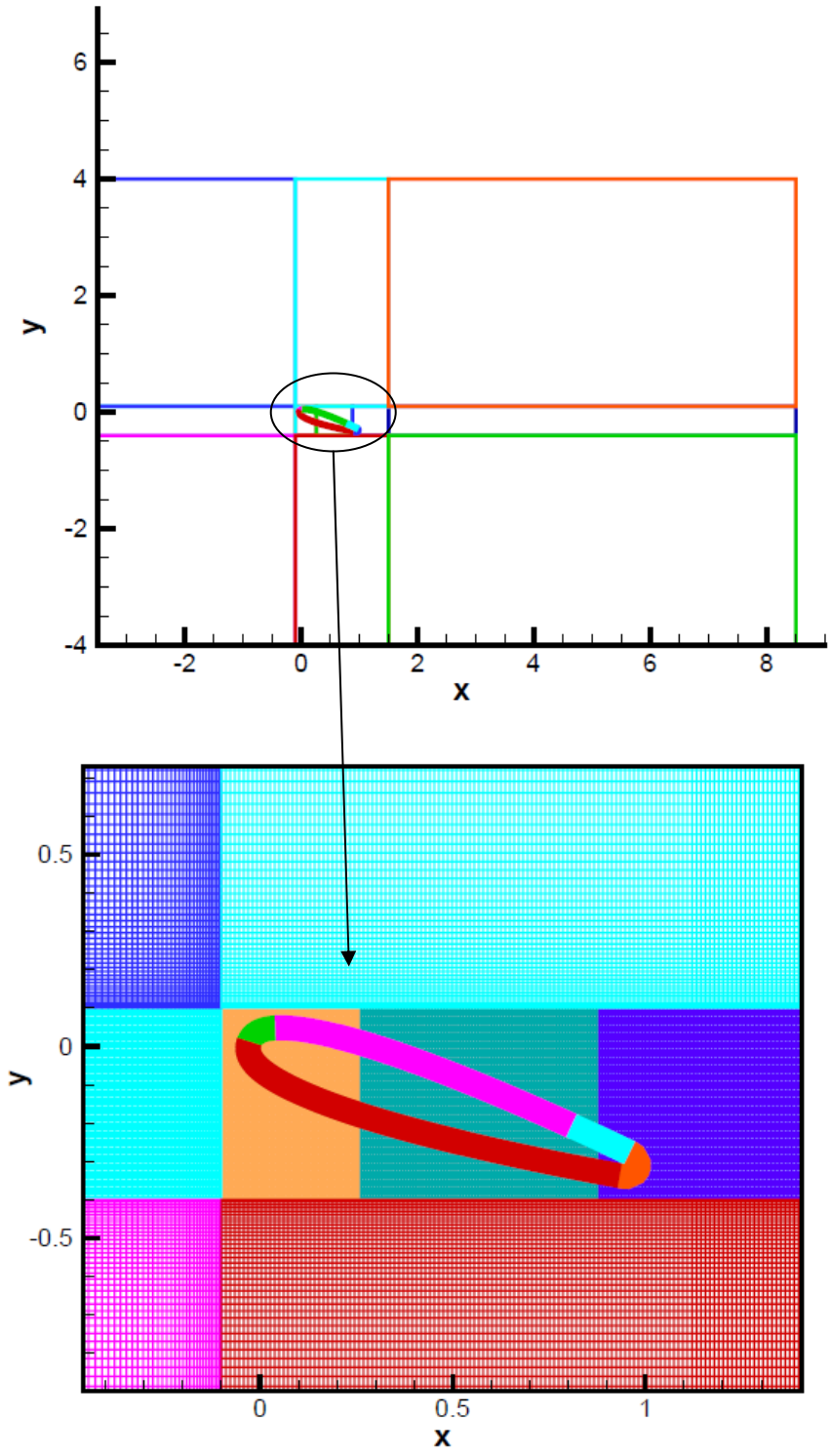


Figure 5.1 (a): Grid and boundary conditions for four jet case

Jet Velocity:

$$u(i,j) = A \times U_\infty \times \cos(\beta)$$
$$v(i,j) = A \times U_\infty \times \sin(\beta) \qquad (5.1)$$
$$\text{where } \beta = (Baseangle + \theta)$$

Computation information between adjacent blocks is exchanged by two ghost points. All the parameters chosen in the computation are dimensionless. An important criterion that has to be satisfied to make sure that we clearly capture the velocity profile near the wall (in the boundary layer) is the non-dimensional wall distance (y^+). This is defined as

$$y^+ = \frac{u_* y}{\upsilon}, \qquad u_* = \sqrt{\frac{\tau_0}{\rho}}, \qquad (5.2)$$

where

$u_* =$ friction velocity at the wall
$\tau_0 =$ shear stress at the wall
$y =$ distance to the wall
$\upsilon =$ kinematic viscosity

Therefore, care was taken to maintain near wall y^+ values of the airfoil blocks within 0.5, well within the region of laminar viscosity (viscous sublayer region). Details on general grid independence and numerical accuracy for this case without flow control jet have been presented in the previous work [20] and are generally satisfactory for the given grid.

5.2 PARAMETER SELECTION

The jet parameters for this case are selected based on our previous [3] [20] single-jet and two-jet cases. Three parameters (Figure. 5.1-b) for each jet are selected for the search investigation, viz. jet location L_j (measured in percent chord), suction/blowing amplitude A (measured in term of orifice velocity relative to the inflow freestream velocity), and suction/blowing angle θ . The jet width of all the four jets was fixed at 2.5% chord length, based

on the study by Dannenberg [100], who showed that increasing the orifice area beyond 2.5% chord will not increase the lift significantly. With an *x* spacing of 0.001 (used for the current setup) on the MidUpper block of the airfoil, there are 25 grid points along the span of the jets.

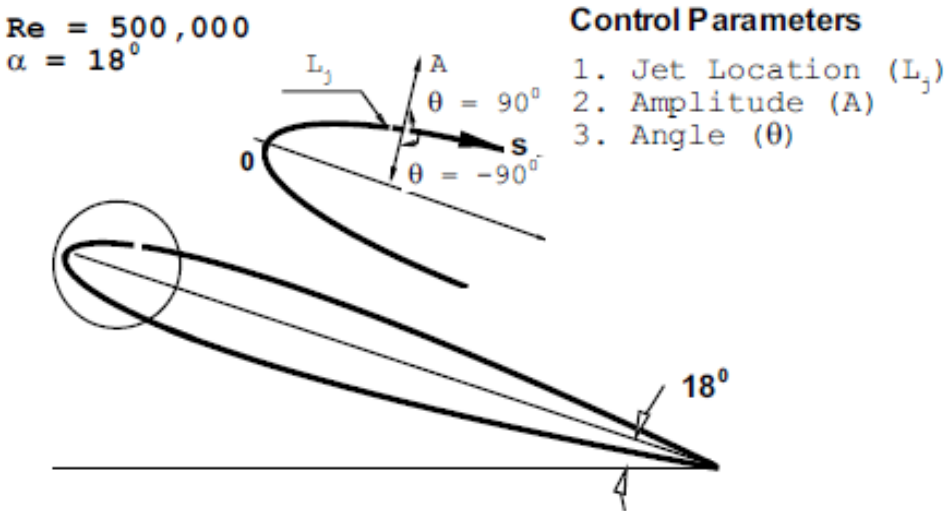


Figure 5.1 (b): Jet parameters for steady case [20]

For the numerical investigation, the jet entrance velocity is set as,

$$u = A.\cos(\theta + \beta)$$
$$v = A.\sin(\theta + \beta)$$

where θ is the angle between jet surface and the jet entrance velocity direction, and β is the angle between the free-stream velocity direction and the local jet surface. Note that a negative θ corresponds to suction while positive indicates blowing. Thus, perpendicular suction is -90^o and perpendicular blowing is 90^o.

It was noticed in the previous two-jet case that the potential flow control due to suction tends to be much more significant than that of blowing in a 'suction and blowing' configuration. In order to prevent suction control from dominating the genetic search (and thereby significantly reducing the complexity of the parameter space), the total suction amplitude of each jet is limited

to be no more than 0.02121, or 2.12% of the inlet velocity, while the net blowing velocity of the jets can reach 0.1414, or 14.14% of inlet velocity. These amplitude values are chosen based on the previous two jet simulations, i.e. in this simulation the potential maximum combined momentum flux supplied by the two suction or blowing jets is the same as the single suction or blowing jet in the two jet case. This corresponds to a momentum coefficient, C_μ (which is defined as shown below), of 0.0 to 0.004231, the maximum value corresponds to sum of all the four jets at full amplitude.

$$C_\mu = \frac{\rho . h . v_j^2}{\rho . c . u_\infty^2} = \frac{h}{c} . A^2, \text{and} \frac{h}{c} = 0.025$$

where,

v_j – Jet velocity

h – Jet width

c – Chord length

u_∞ – Freestream velocity

ρ – Density

This satisfies the criterion of a momentum coefficient to be around 0.002 or more to have some effect on the flow field [54]. Further, the amplitude of one of the suction jets is fixed at the full allowed amplitude, again based on the previous simulations. The location of the jets L_j is varied from 5% chord to 80% of chord. So, considering all the four jets, we have a total of 11 parameters to be optimized which are listed in Table 5.1.

5.3 GENETIC PARAMETERS

As compared to the two jet setup, the four-jet evolution more than doubles the number of parameters considered. Angle and location for each jet is allowed to vary in the same ranges as the two jet case and the amplitude of one suction jet is fixed as before. The genetic coefficients for the Continuous GA are set as,

$$NGeneration = 50$$
$$NPopSize = 56$$
$$NVariable = 11$$
$$Mutation\,Const. = 0.15$$
$$Crossover\,Const. = 0.50$$

and the aggregate fitness is defined as,

$$\left(Fit_A\right)_{\max} = a.C_l / C_{lB} + b.C_{dB} / C_d \tag{5.2}$$

a and b were set to 1, providing equal weight to both lift and drag and making the baseline fit equal to 2.0. This definition of fitness also provides flexibility of adjusting *a* and *b* such that different importance of lift and drag could be explored for a given search.

The fitness function used for the EARND GA is same as the one used for the Continuous GA (Equation 5.2). The genetic coefficients used for the EARND GA are listed below.

$$NGeneration = 50$$
$$NPopSize = 56$$
$$NVariable = 11$$
$$Mutation\,Const. = 0.10$$
$$Crossover\,Const. = 0.20$$

The mutation and crossover percentages for this GA were selected based on its performance in the two-jet system. Apart from the above coefficients the EARND GA also has the following parameter.

$$NUpdate = 8$$
$$Diversity\,Control = 20\%$$

Boundary updates occur each *NUpdate* generation over the second 50% of the evolution. The boundaries for each variable design space are explicitly updated according to the statistics of the best individuals up to that generation. The diversity control is the percentage of the total generations in which additional care is taken to maintain diversity.

Table 5.1: Parameter range for the four jet case

Variable Name	***Range***
Suction location –1 (L_{jS1}):	$0.05 \le L_{jS1} \le 0.80$
Suction angle – 1 (θ_{S1}):	$-90^o \le \theta_{S1} \le 0^o$
Suction Amplitude – 1 (A_{S1}):	$A_{S1} = 0.02121$
Suction location – 2 (L_{jS2}):	$0.05 \le L_{jS2} \le 0.80$
Suction angle – 2 (θ_{S2}):	$-90^o \le \theta_{S2} \le 0^o$
Suction Amplitude – 2 (A_{S2}):	$0.0 \le A_{S2} \le 0.02121$
Blowing location – 1 (L_{jB1}):	$0.05 \le L_{jB1} \le 0.80$
Blowing angle – 1 (θ_{B1}):	$0^o \le \theta_{B1} \le 90^o$
Blowing amplitude – 1 (A_{B1}):	$0.0 \le A_{B1} \le 0.1414$
Blowing location – 2 (L_{jB2}):	$0.05 \le L_{jB2} \le 0.80$
Blowing angle – 2 (θ_{B2}):	$0^o \le \theta_{B2} \le 90^o$
Blowing amplitude – 2 (A_{B2}):	$0.0 \le A_{B2} \le 0.1414$

5.4 OPTIMIZED CONFIGURATION - CGA

The simulation consisted of 50 generations with 56 individuals per generation, leading to a total of 2800 simulations. The values of the baseline lift and drag (C_{lB} and C_{dB}) were determined from simulations of the base airfoil (without jets) and were fixed at 0.8918 and 0.1610 respectively. The initial population was selected such that, the individuals cover the full allowed parameter range. The maximum fitness obtained from the evolution was 2.1910. This fitness value corresponds to the jet parameters listed in Table 5.2.

Table 5.3 lists the best 10 individuals of the evolution. The fitness values of these top ten individuals are nearly equal, but the configurations are somewhat different. It is interesting to note that the suction jets configuration does not vary much (particularly the leading suction jet), but the blowing jet configuration does not behave in a similar manner. The location of blowing jets varies over a much bigger range as compared to the suction jets while still yielding a similar fitness value. The same analysis applies to other parameters of the blowing jets, suggesting that

the blowing parameters are less important as compared to the suction jets. More discussion about the sensitivity of these parameters is presented in the latter part of this section.

Table 5.2: Best configuration obtained from CGA simulation

Leading Suction	0.0500	Trailing Suction	0.0942	Location
	-84.3770		-82.2890	Angle
	0.0212		0.0207	Amplitude
Leading Blowing	0.5219	Trailing Blowing	0.7150	Location
	0.5970		26.1910	Angle
	0.0423		0.0928	Amplitude
Cl	0.9962	**Cd**	0.1499	
Fitness		**2.1910**		

Figure 5.2 presents the configurations from all the 50 generations of the CGA simulation, sorted by fitness, the most-fit configuration being the one on the *y*-axis, and Figure 5.3 presents the best 500 configurations. The configurations that yield the best fitness are those where both suction jets take a leading position on the airfoil, with an orientation approaching normal suction, and both jets operating at near-maximum amplitude. In effect, the two jets act much like a single suction jet, similar to the earlier two jet simulation [20], but in the two jet case the preferred location of the suction jet was a bit further back (at about 13% chord). The trailing suction jet generally seems to favor a slightly less normal orientation compared to the leading suction jet.

Table 5.3: Best 10 individuals from CGA simulation

Gen →	Parameter	36	29	31	31	31	30	26	37	26	38
Leading Suction	Location	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	0.0501	0.0500	0.0500
	Angle	-84.3770	-85.6760	-84.4750	-84.5630	-84.4750	-85.6760	-84.3770	-88.0070	-84.3770	-84.3770
	Amplitude	0.0212	0.0212	0.0212	0.0212	0.0212	0.0212	0.0212	0.0212	0.0212	0.0212
Trailing Suction	Location	0.0941	0.0818	0.0902	0.0897	0.0902	0.0818	0.0941	0.0933	0.0941	0.0941
	Angle	-82.2890	-76.0530	-83.1200	-81.6100	-83.1200	-76.0530	-82.2890	-79.0220	-82.2890	-82.2890
	Amplitude	0.0207	0.0205	0.0204	0.0205	0.0204	0.0205	0.0202	0.0206	0.0202	0.0202
Leading Blowing	Location	0.5219	0.4584	0.5095	0.4924	0.5095	0.4584	0.5219	0.4996	0.5219	0.5219
	Angle	0.5970	1.7690	2.0050	2.0110	2.0050	1.7690	0.5970	3.4650	0.5970	0.5970
	Amplitude	0.0423	0.0240	0.0275	0.0730	0.0275	0.0240	0.0423	0.0141	0.0423	0.0709
Trailing Blowing	Location	0.7150	0.7836	0.7561	0.7328	0.7561	0.6702	0.7150	0.6719	0.7150	0.7150
	Angle	26.1910	21.9110	38.7450	29.9180	27.0090	21.9110	26.1910	22.1910	26.1910	26.1910
	Amplitude	0.0928	0.1020	0.1028	0.1025	0.1028	0.1020	0.0928	0.0950	0.0631	0.0890
C_l		0.9962	0.9965	0.9925	0.9936	0.9953	0.9949	0.9955	0.9948	0.9962	0.9953
C_d		0.1499	0.1500	0.1494	0.1496	0.1499	0.1498	0.1500	0.1499	0.1501	0.1500
C_l / C_d		6.6440	6.6440	6.6450	6.6420	6.6400	6.6400	6.6370	6.6360	6.6360	6.6360
Fitness		**2.1910**	**2.1910**	**2.1910**	**2.1906**	**2.1902**	**2.1902**	**2.1899**	**2.1897**	**2.1897**	**2.1896**

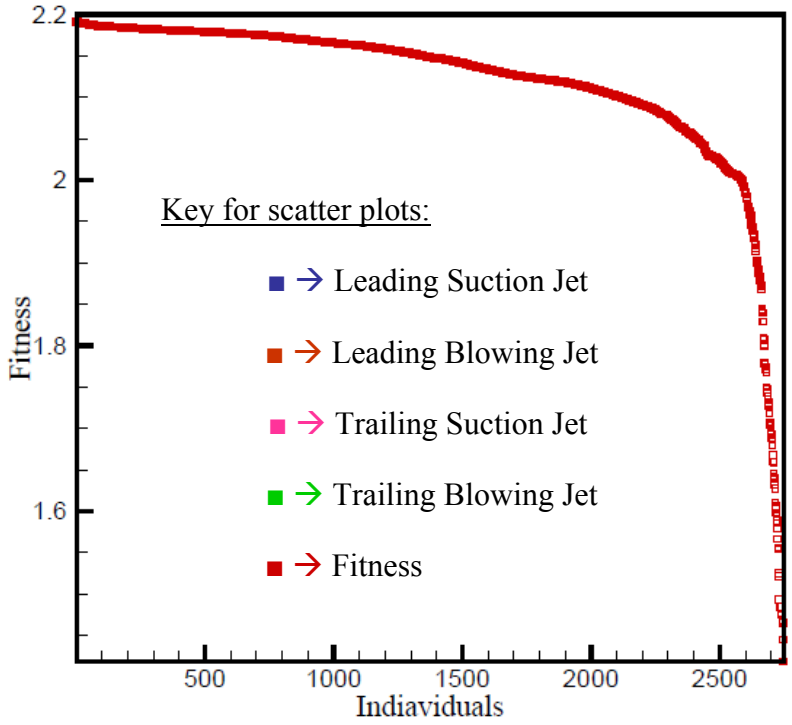


Figure 5.2 (a): Fitness of all configuration from CGA

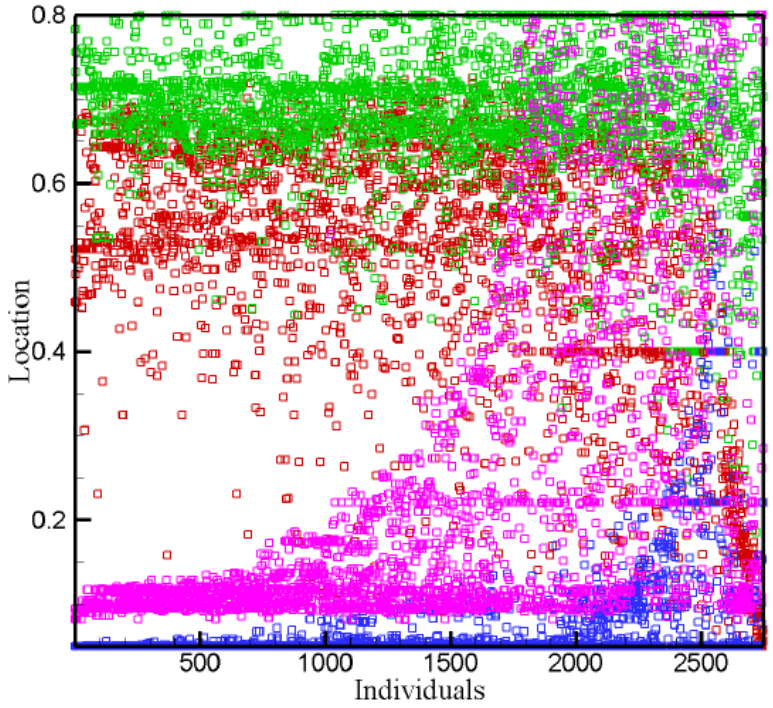


Figure 5.2 (b): Location sorted by fitness - CGA

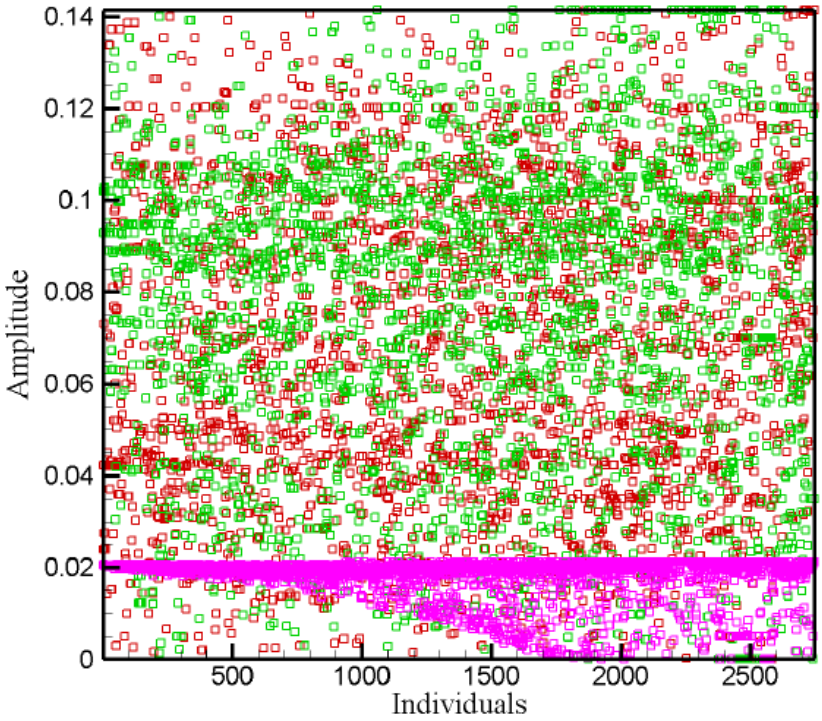


Figure 5.2 (c): Amplitude sorted by fitness – CGA

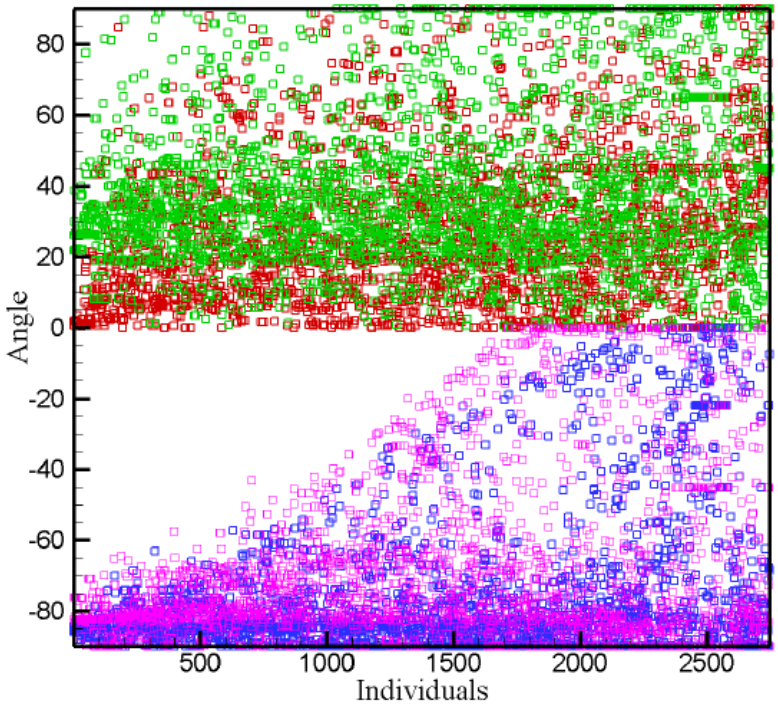


Figure 5.2 (d): Angle sorted by fitness – CGA

The characteristics of the suction jets are relatively consistent across the top 500 configurations (Figure 5.3). The blowing jets, on the other hand, exhibit a far greater spread of values in the high fitness range. The first 460 configurations have a fitness of 2.18 or higher, which suggests that the overall influence of the blowing jets on the computed lift and drag is smaller even though these jets can have much higher amplitude. The best solutions have the leading blowing jet generally located near the middle of the surface (50% chord), with a near-tangential angle and at lower amplitude (1%-7% freestream) relative to the more rearward (trailing) jet.

The trailing blowing jet is located more towards the trailing edge (greater than 65% chord) at a moderate angle (20^0 to 40^0) and at amplitude higher than the leading blowing jet, but less than the maximum possible amplitude (about 10% of freestream velocity). However, there are a considerable number of solutions that violate these general conditions, again suggesting that the blowing jets play a less important role in this flow control scheme, with high fitness results possible even when one of the blowing jets approach zero amplitude. So, like in the previous two jet studies, the suction jets play the dominant role.

A closer look at the top 500 parameters individually (Figure 5.4), suggests that the dominant parameters converge more aggressively than the other (not so dominant) parameters. Leading suction jet location (Figure 5.4 (a)) is almost a straight line, with the jet located at the most leading position possible, 5% chord, with a near normal angle.

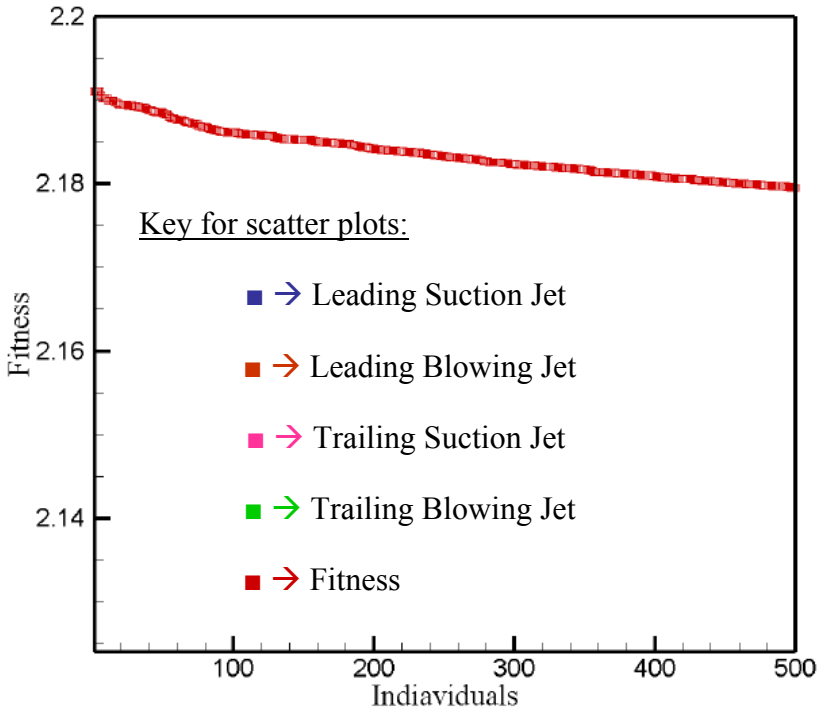


Figure 5.3 (a): Fitness of best 500 individuals – CGA

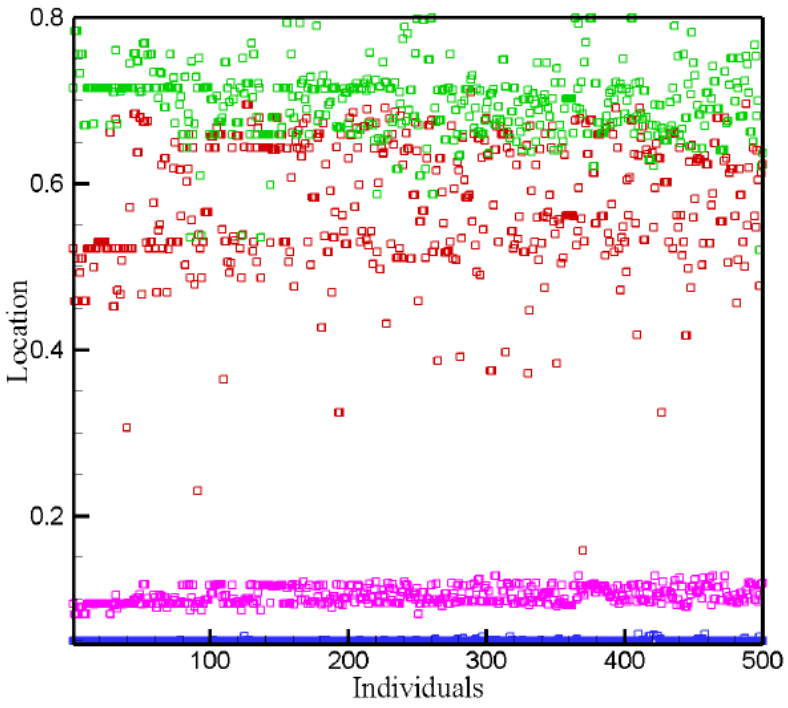


Figure 5.3 (b): Location of best 500 individuals - CGA

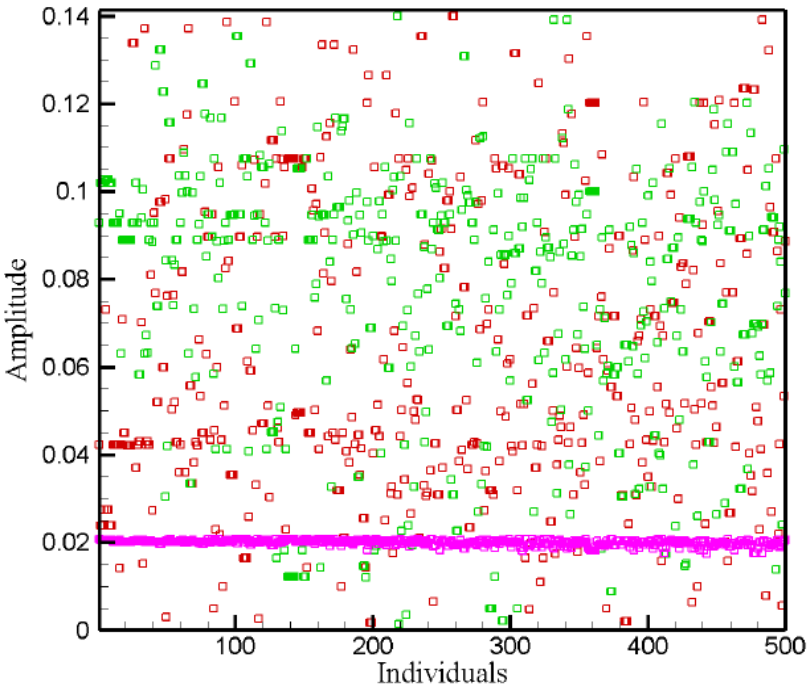


Figure 5.3 (c): Amplitude of best 500 individuals - CGA

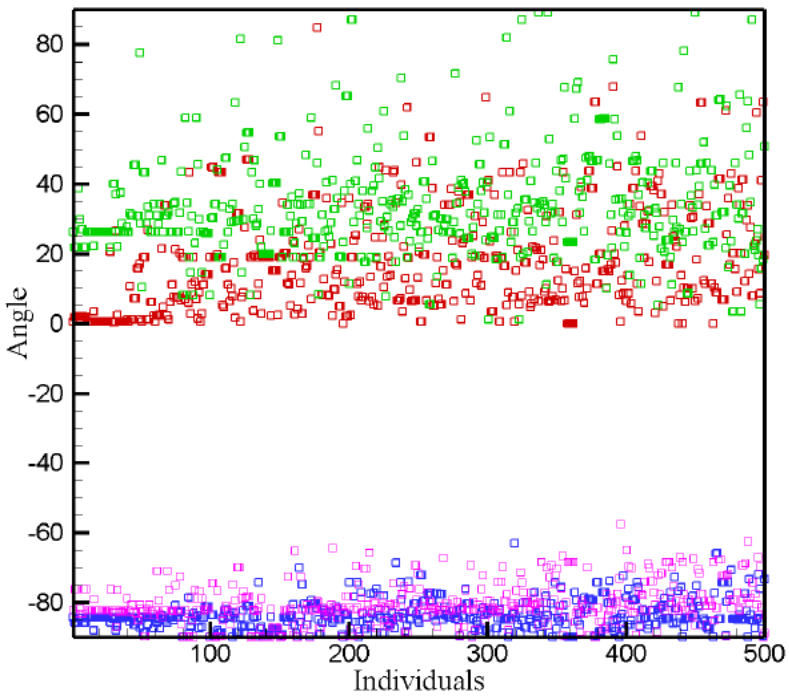


Figure 5.3 (d): Angle of best 500 individuals – CGA

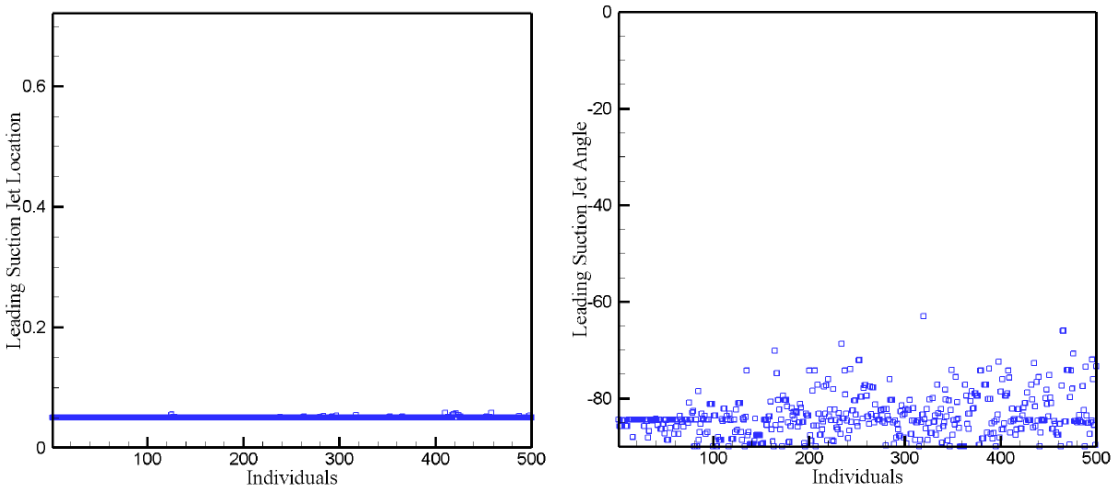


Figure 5.4 (a): Best 500 leading suction jet configurations (fixed amplitude)

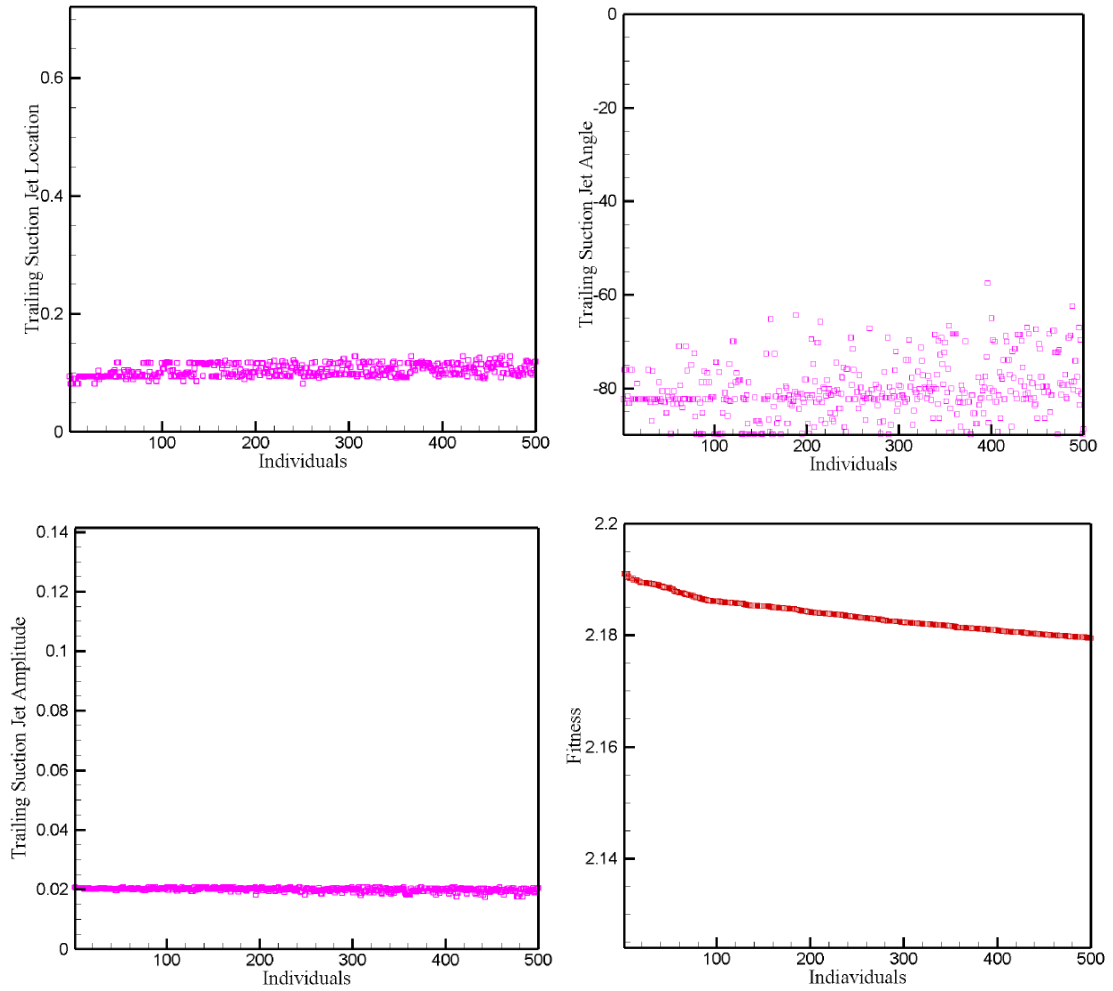


Figure 5.4 (b) Best 500 trailing suction jet configurations along with fitness

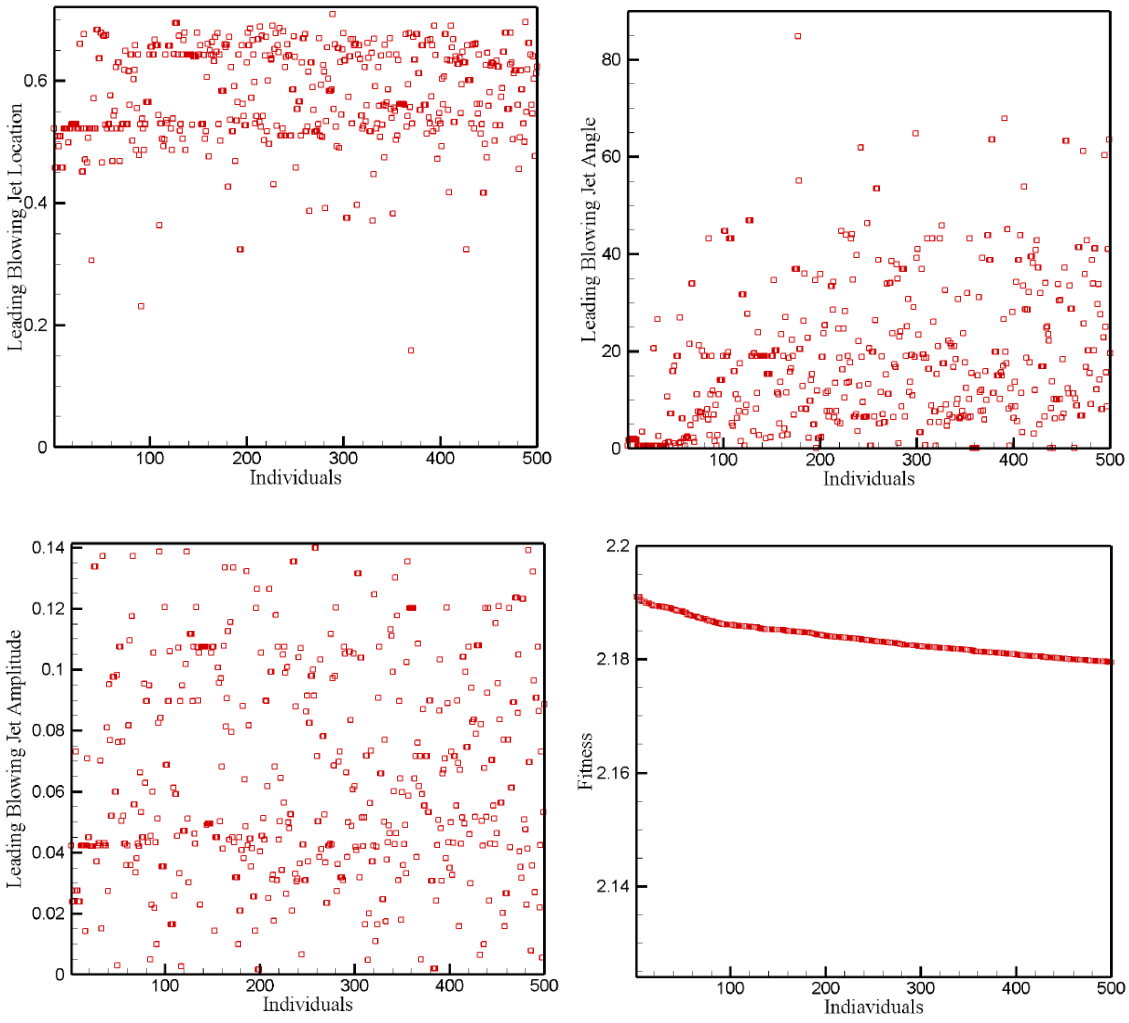


Figure 5.4 (c): Best 500 leading blowing jet configurations along with fitness

As shown in Figure 5.4(b), the trailing suction jet is not very far from the leading suction jet; located at about 10% chord, it almost coincides with the leading suction jet. As mentioned earlier, this is consistent with the earlier two jet study, where it was suggested that if two suction jets were used, then the system will favor towards placing the jets in such a way that they behave as a single suction jet. The amplitude of this jet is also pushed towards the full allowed amplitude and the angle is generally a bit less normal relative to the leading suction jet (-75^0 to -90^0).

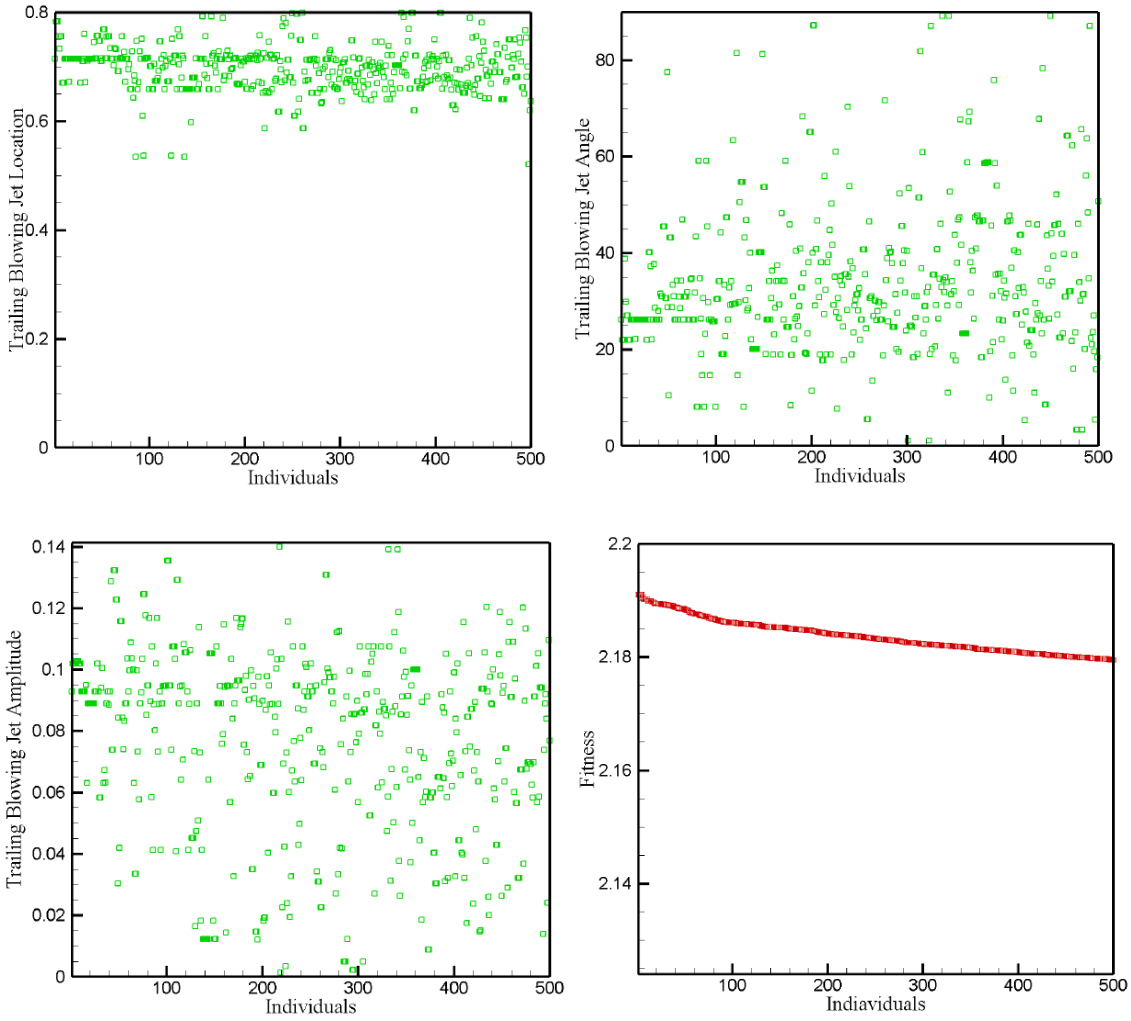


Figure 5.4 (d) Best 500 trailing blowing jet configurations along with fitness

For the leading blowing jet (Figure 5.4 (c)), the location is fairly consistent for the top 500 individuals, i.e. between 45% chord to 70% chord, with a more tangential angle (0^0 to 40^0). The blowing amplitude on the other hand is not very consistent and spreads almost along the whole parameter range, even for just the best 500 individuals. The location of the trailing blowing jet (Figure 5.4 (d)) is relatively consistent among the best 500 individuals. The amplitude on the other hand is again spread out in the allowed parameter range. The angle seems to be more converged than the amplitude hovering between 15^0 and 45^0 for the best fit

individuals. Interestingly the trailing blowing jet angle is generally a bit higher than the leading blowing jet.

5.5 FLOW CONTROL PHYSICS

To better understand the effect of suction and blowing on the flow field of the system we compare the vorticity and streamline plots of the baseline (no jets) case with the optimum configuration. We also consider suction and blowing separately for this analysis.

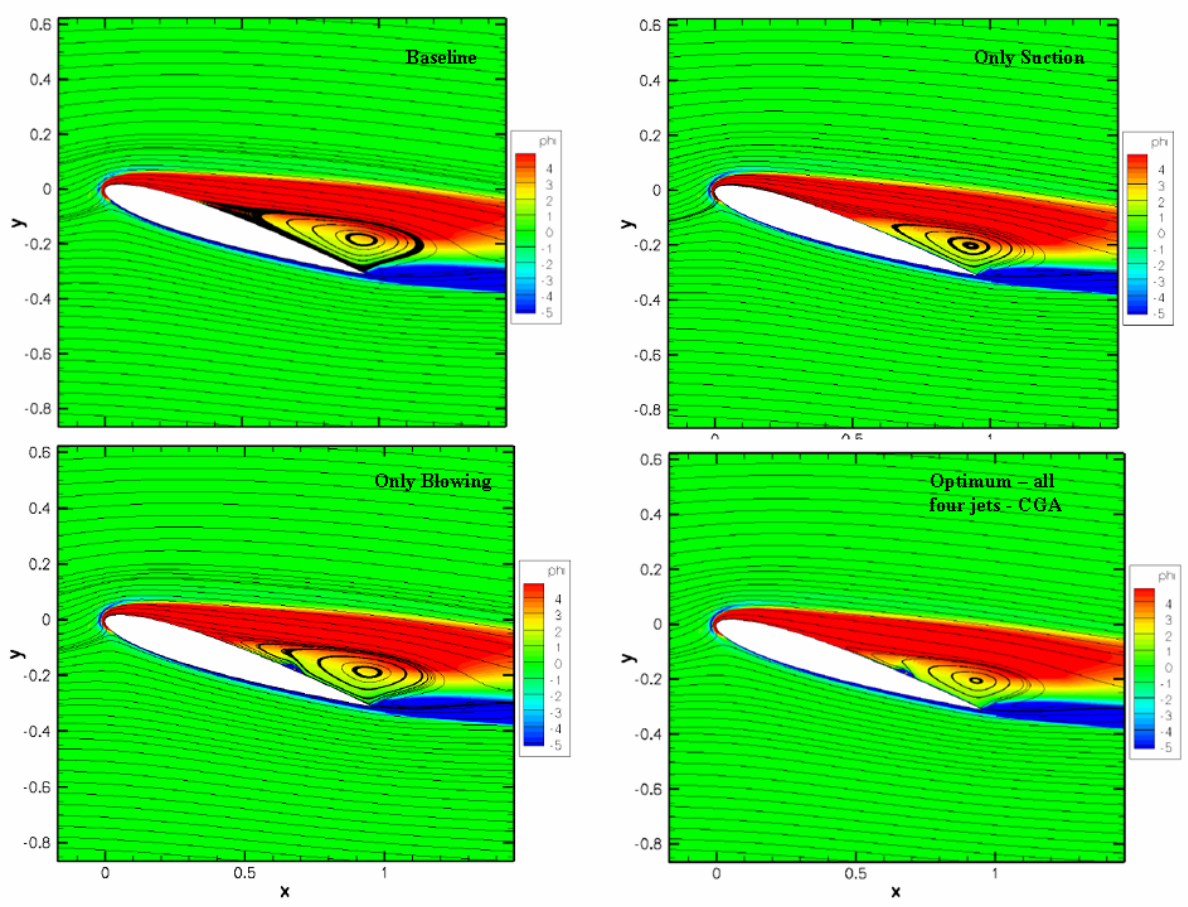


Figure 5.5: Streamline and vorticity plot using the CGA configuration

The lift and drag improvements are possibly a result of the weakening of the separation bubble (Figure 5.5). Comparing the baseline case with the case where the jets are in optimized configuration, we can see a clear reduction in the size of the separation bubble, thus augmenting

lift and suppressing drag. To see the effects of blowing and suction, plots of suction only and blowing only are also shown. Adding the suction jets reduces the size of the separation and shifts

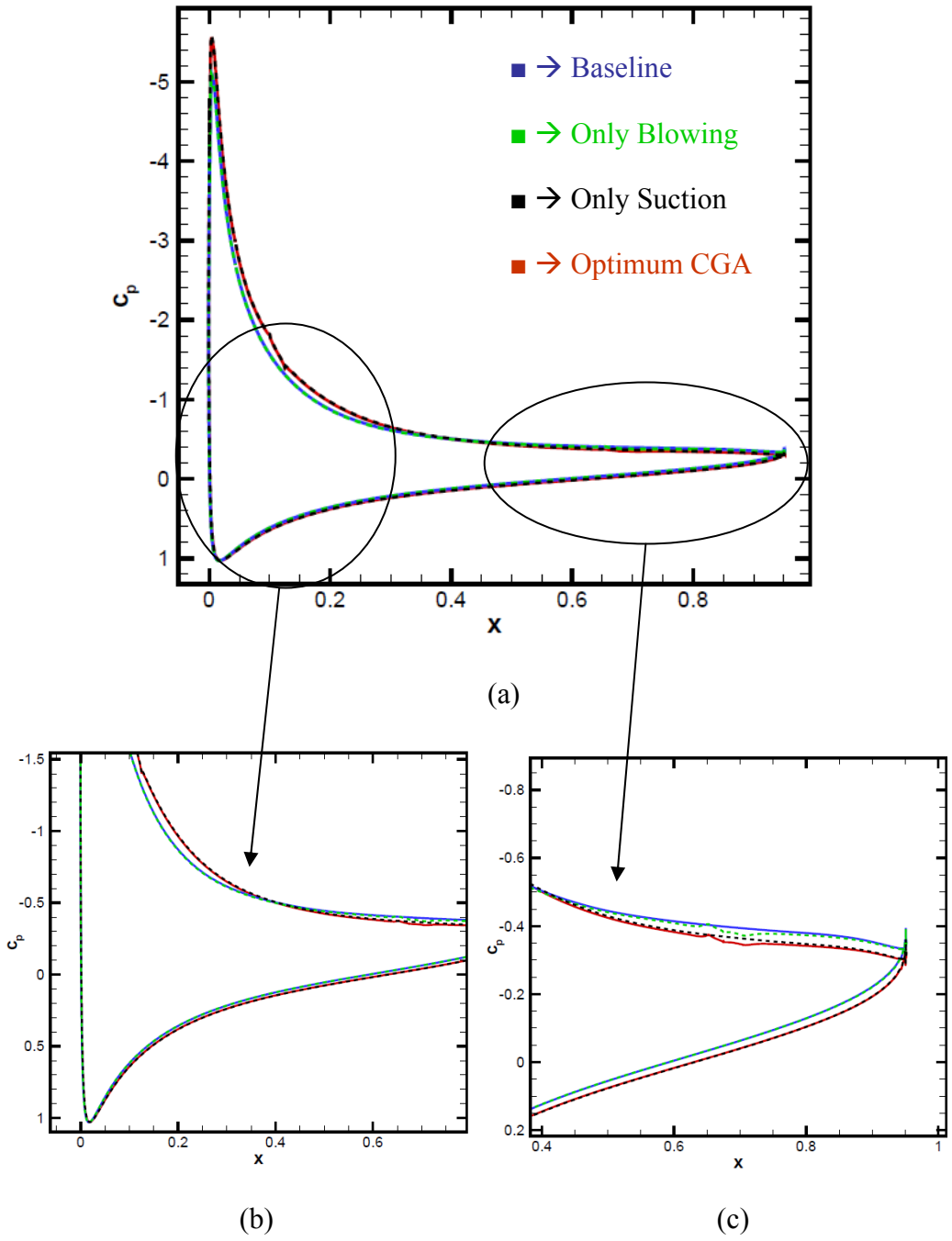


Figure 5.6: C_p (coefficient of pressure) using the CGA configuration

the separation point downstream. The suction-only jet configuration is visually indistinguishable from the full four jets configuration. The blowing jets are not particularly effective by themselves and in this case the size of the separation bubble is not very different from the baseline case. But when combined with the two suction jets, the location of the leading blowing jet is close to the point of separation with the near-tangential blowing, accelerating the flow downstream, potentially keeping the flow attached a bit longer.

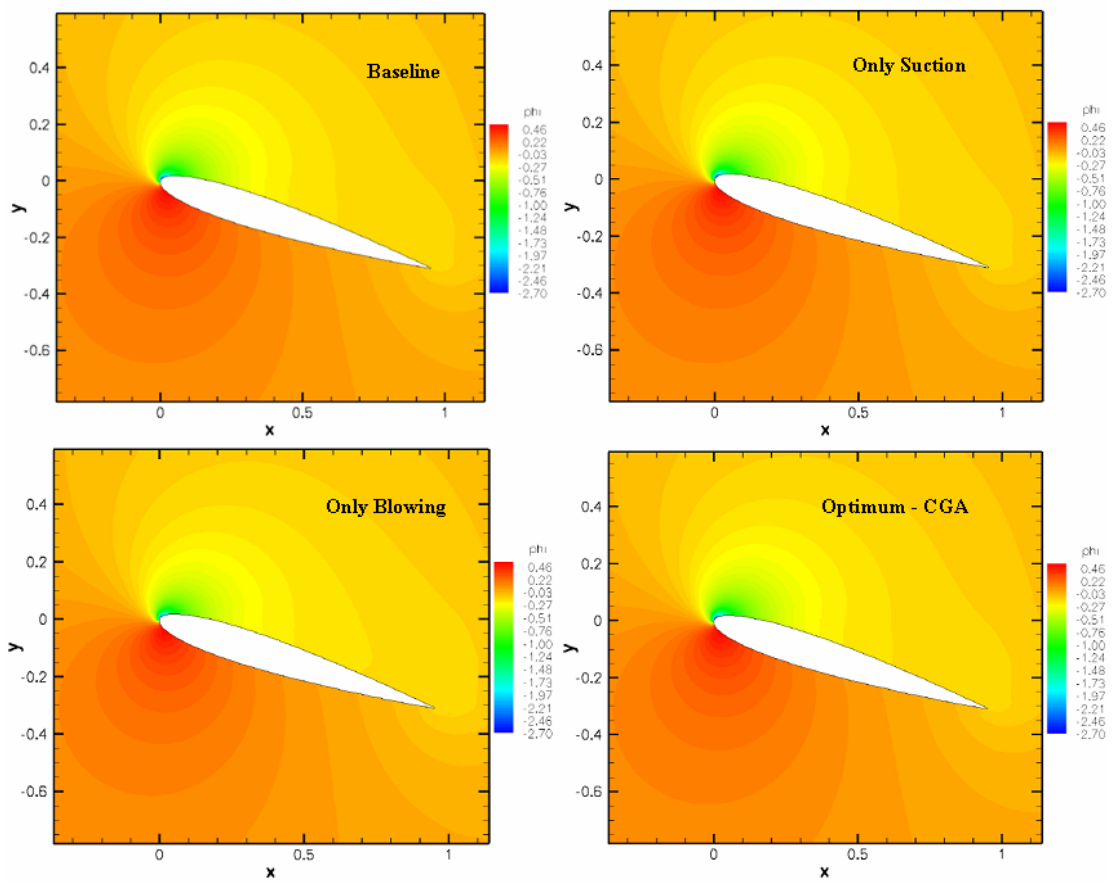


Figure 5.7: Pressure plots using the CGA configuration

The more trailing blowing jet is in the separation bubble and is also relatively tangential, presumably to counter the adverse flow of the re-circulation. This effect of the blowing jet could be seen in the C_p and the pressure plots (Figure 5.6 and 5.7). The plot curves of blowing only and the base line case overlap at most places and so are the suction only and optimum case, again

illustrating the minimum change in these two set of cases. Also it is interesting to note that the mitigating effects of the two blowing jets appear to be fairly limited and seem to be weaker than the effect generated by the single blowing jet in the two jet simulation [20]. The skin friction (C_f) plot (Figure 5.8) and the lift-drag plots (Figure 5.9) are also consistent with the above analysis.

Table 5.4: Separation point for various configurations

Configuration	% of Chord
Baseline	22.686
Only blowing	22.899
Only suction	30.391
Optimum (CGA)	30.412

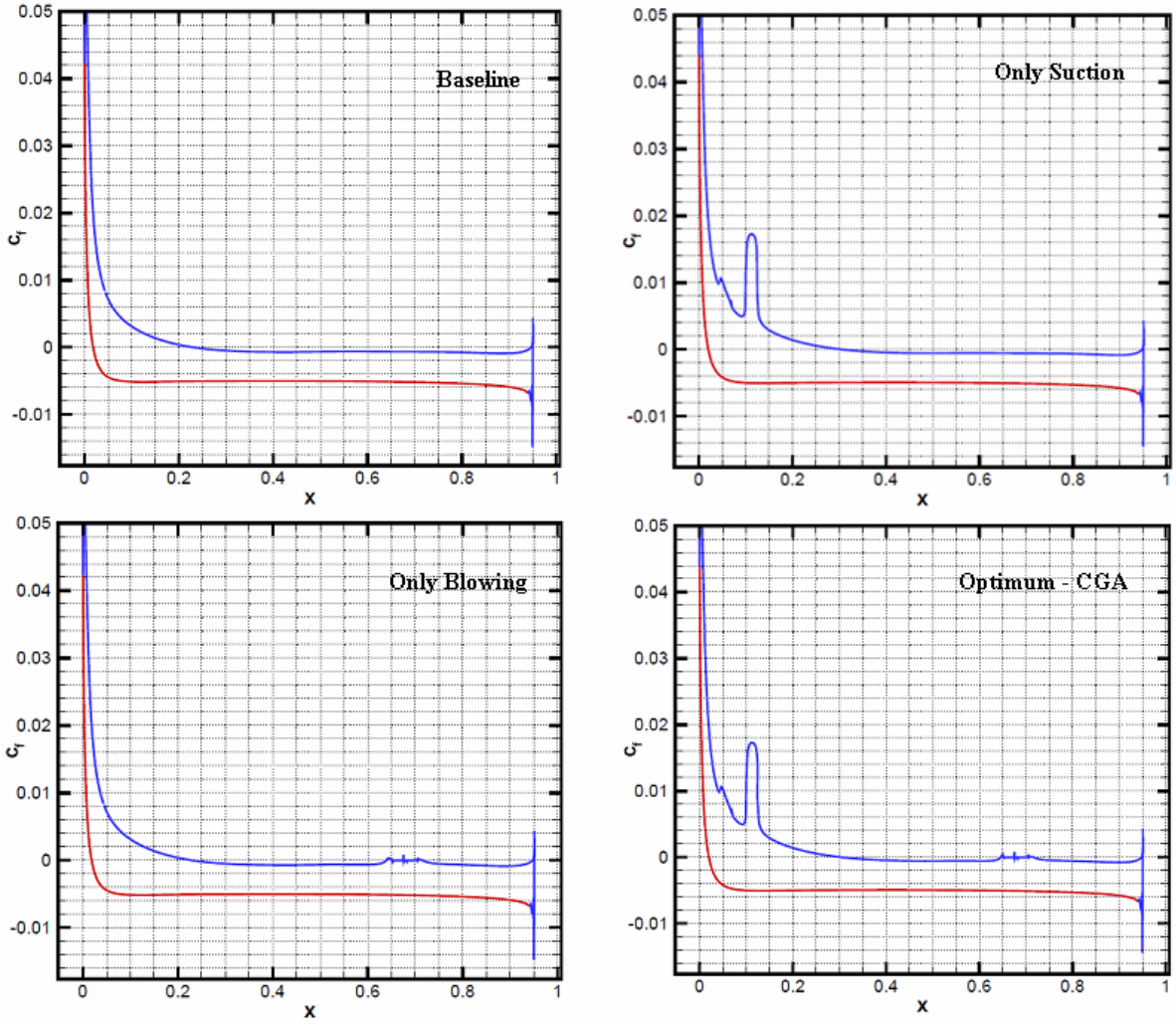


Figure 5.8: C_f (skin friction) using the CGA configuration

Using the C_f plots the location of the separation point was obtained for each of the configurations above (Table. 5.4). It is clear from the table that the point of separation moves downstream in the optimum configuration relative to the baseline case, but again the effect of only blowing or only suction is not very different from the baseline and the optimum cases respectively. Also, a closer look at the blowing only and optimum cases shows the effect of the blowing jet. Relative to the suction-only and baseline case, the curve is pushed upwards, i.e. C_f is moved towards zero or higher. This presumably is an affect of the second blowing jet which is located around the same region (65% to 70% chord) that may be augmenting the circulation around the region and thus keeping the flow attached a bit longer than the former cases. Figure 5.9 show the evolution of the lift and drag of various configurations. Each CFD evaluation is iterated 35000 times for the steady case and the plot clearly shows the converged steady state solution at the end of 35000 iterations.

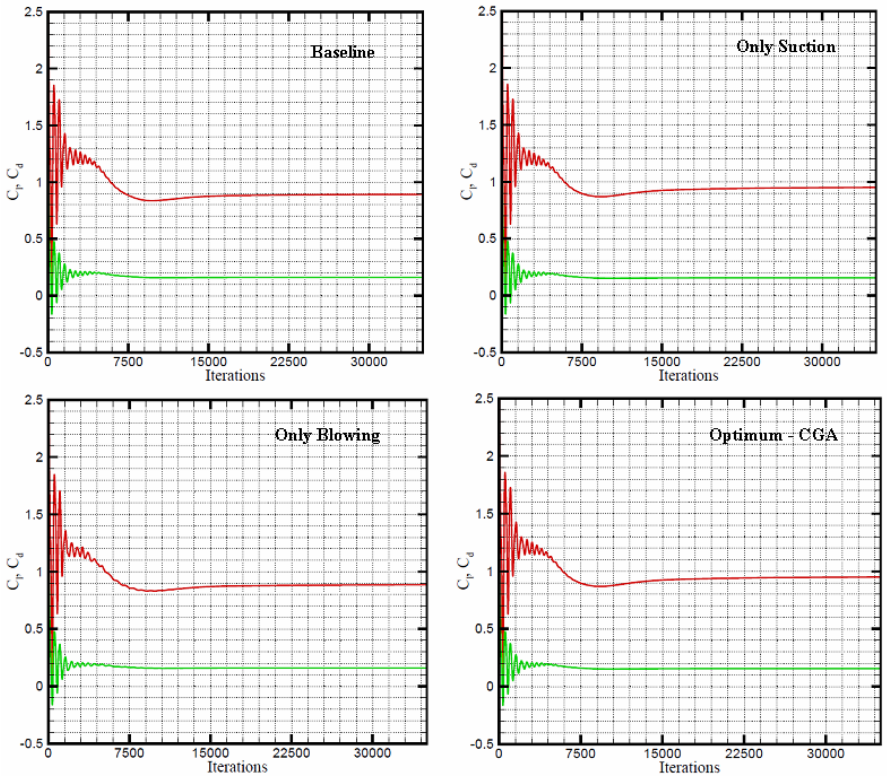
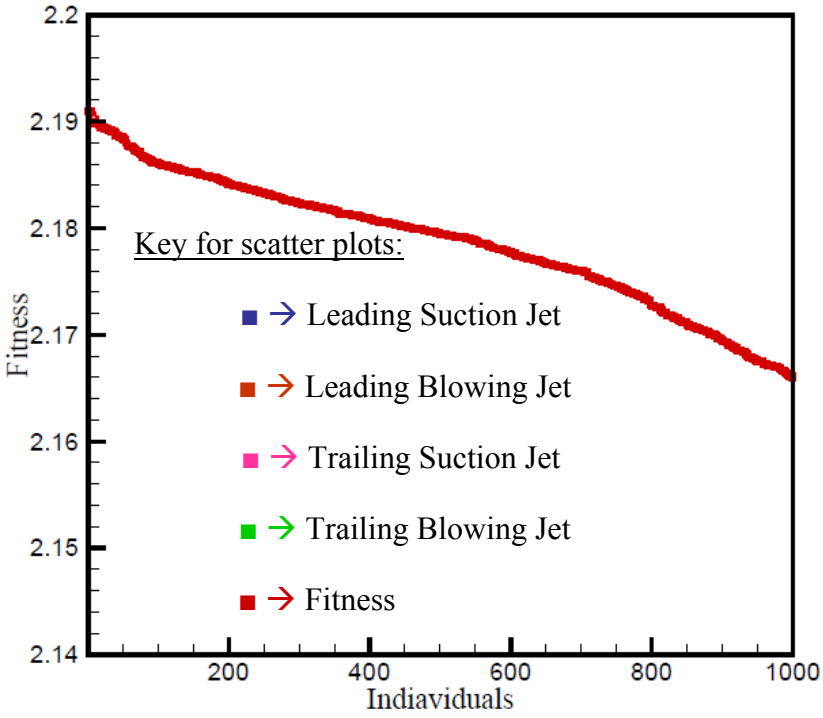
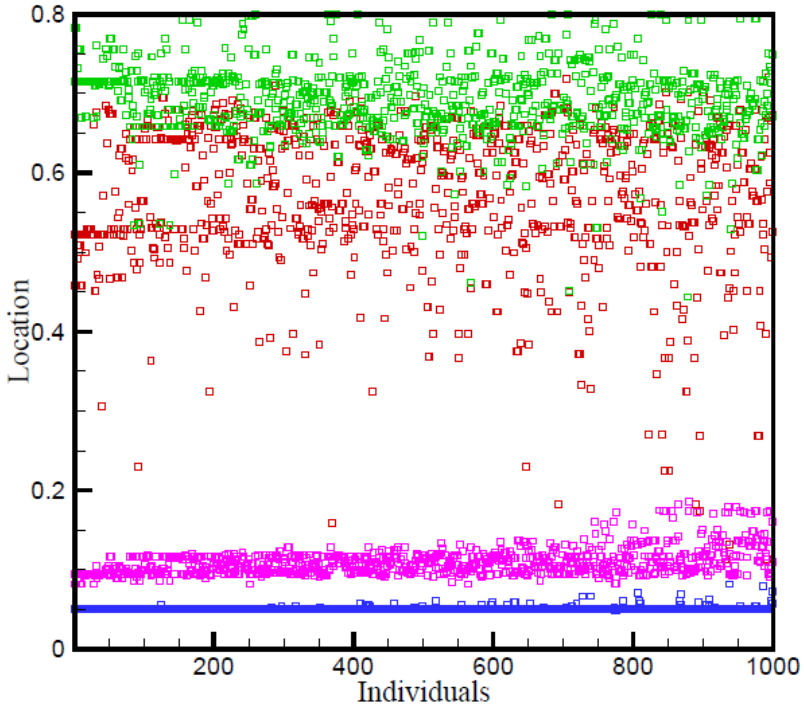


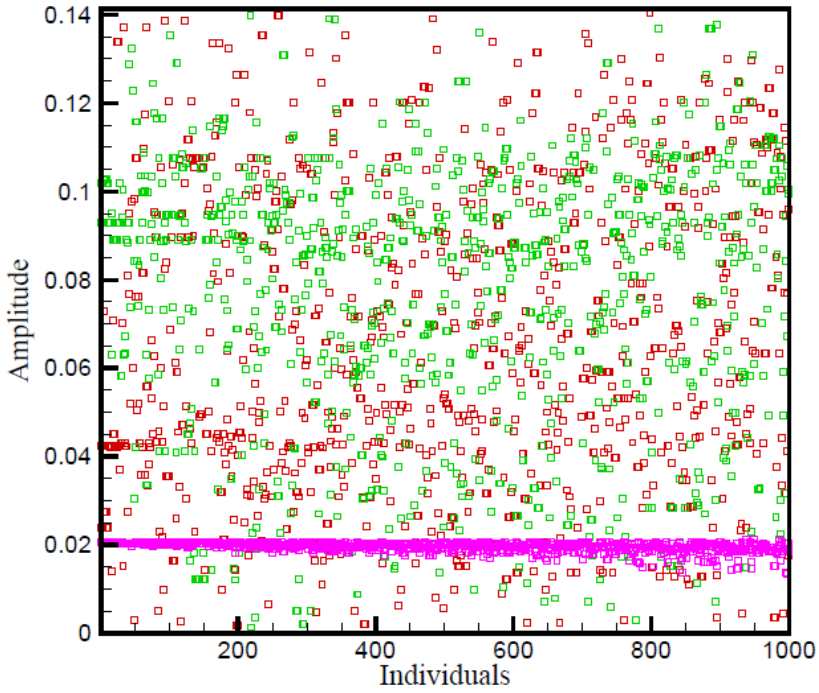
Figure 5.9: C_l and C_d using the CGA configuration


2.2
2.19
2.18
2.17
2.16
2.15
2.14
Fitness
200
400
600
800
1000
Indiaviduals
Key for scatter plots:
Leading Suction Jet
Leading Blowing Jet
Trailing Suction Jet
Trailing Blowing Jet
Fitness

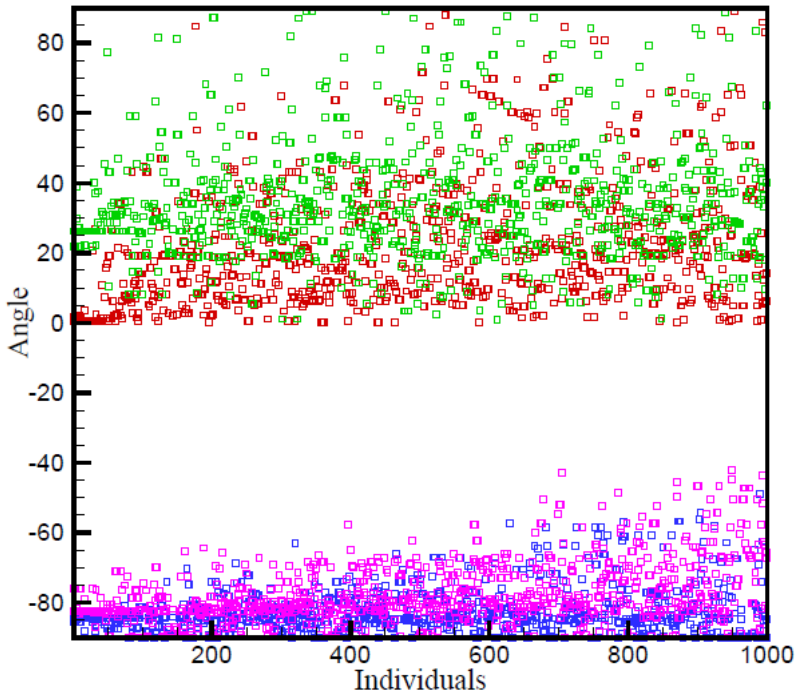
(a)


0.8
0.6
0.4
0.2
0
Location
200
400
600
800
1000
Individuals

(b)



(c)



(d)

Figure 5.10 (a) to (d): Jet parameters for configurations having fitness within 1.5% of the maximum fitness

An alternate way to look at the data is to consider all the configurations that fall within a certain error limit of the maximum fitness value and see how wide a range these parameters cover. Figure 5.10 show solutions from the CGA evolution with fitness values within 1.5% of the maximum fitness value. The error allows for the fact that the computation may not precisely mirror reality, or that reality can not always be accounted for with this degree of numerical rigor in reasonably complex flows.

From these results it is apparent that the suction location is fairly tightly bound for even moderate improvement in the fitness value and that the suction angle and blowing location are reasonably constrained. Conversely, the blowing amplitude is more weakly constrained, with about half the parameter space being able to achieve the 1.5% error in fitness. Similarly the blowing angle is hardly restricted, with reasonably close solutions being generated for nearly all angles with in the allowed range.

5.6 EARND GA

A second simulation consisting of the same parameters was done using the EARND GA. The EARND GA was originally developed to optimize the two-jet control system and was successful in doing so [3] [20]. It is clear from the fitness plots (Figure 5.11) that the EARND GA did not perform as well relative to the Continuous GA. The maximum fitness value obtained is only 2.1228 (Table 5.5) which is about 6% higher than the baseline fitness. Although this is a reasonable improvement considering the complexity of the system, the EARND GA seems to have got stuck in a not so optimum solution space. This is based on the results obtained using the Continuous GA which obtained a fitness of 2.1910, an increase of 9.5% from the baseline value.

To better understand what may have gone wrong with the EARND GA, we look at the convergence of the various parameters and compare the convergence pattern with the Continuous GA convergence pattern (Figure 5.12). Table 5.6 presents the 10 best configurations generated

by the EARND GA, with clear differences in the placement of the jets, the angles of both suction jets and one blowing jet, and the magnitude of blowing in both cases relative to the Continuous GA configurations. The most prominent parameters that the EARND GA has failed to optimize are the trailing suction jet location and angle. It seems to have converged these parameters into local optima, thus yielding a lesser fitness than the CGA fitness. The reduced effect of this configuration on the separation bubble can be seen in Figure 5.13(a), thus having a weaker effect on lift and drag, when compared with the Continuous GA configuration Figure (5.13-b). This result was something of a surprise given the performance of this search approach on the previous two jet configuration.

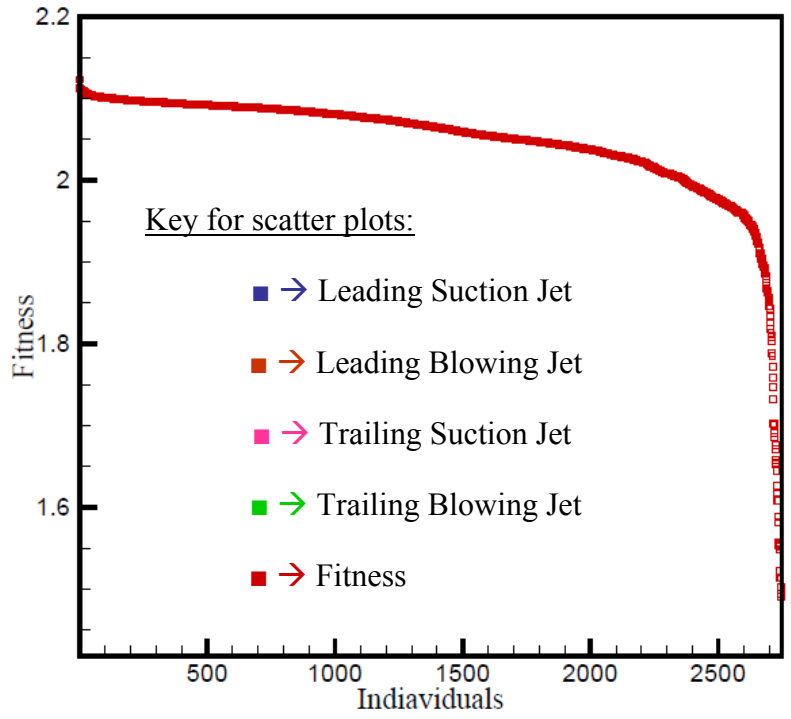
A possible cause of the failure is that with only 56 individuals per generation and 11 parameters, the degree of diversity in each generation may not be sufficient to overcome unfortunate random mutations and crossovers. The selection scheme of the Continuous GA, in which the best half of the previous generation is explicitly retained, can mitigate against dramatic fitness declines. On the other hand, the techniques designed to drive convergence in the EARND GA can keep the evolution heading towards a less optimal configuration region once it gets started down the wrong path. As with any stochastic approach, it is quite possible that a repeat of the EARND GA evolution would randomly stumble onto a better path and yield a better result, but the outcomes of the two completed evolutions and the algorithm details suggest that the Continuous GA may be the more robust approach for this type of flow control problem.

Table 5.5 Best configuration from the EARND GA

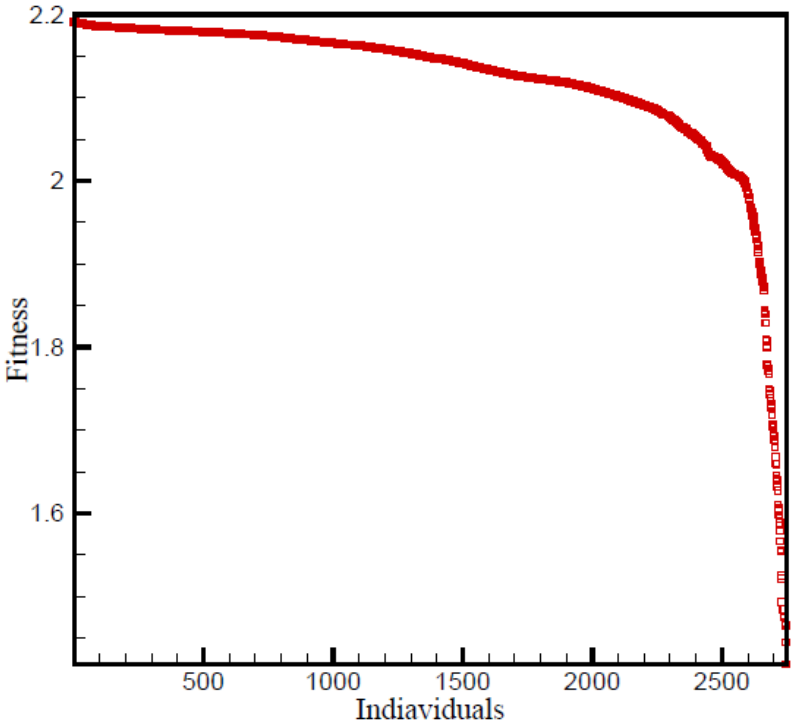
Leading Suction	0.0500	Trailing Suction	0.2200	Location
	-90.0000		0.0000	Angle
	0.0212		0.0212	Amplitude
Leading Blowing	0.6188	Trailing Blowing	0.800000	Location
	45.0000		90.000000	Angle
	0.1000		0.141400	Amplitude
Cl	0.9253	**Cd**	0.1484	
Fitness		**2.1228**		

Table 5.6 Best 10 configurations obtained using EARND GA

Gen →	Parameter	2	27	41	9	43	21	33	49	29	47
Leading Suction	Location	0.0500	0.0500	0.0605	0.0500	0.0545	0.0500	0.0776	0.0500	0.0500	0.0500
	Angle	-90.0000	-56.0900	-64.6613	-55.3155	-59.4624	-55.0600	-67.2103	-59.6696	-60.3479	-56.4229
	Amplitude	0.0212	0.0212	0.0212	0.0212	0.0212	0.0212	0.0212	0.0212	0.0212	0.0212
Trailing Suction	Location	0.2200	0.3801	0.4011	0.4877	0.4339	0.5454	0.3663	0.4380	0.4266	0.4500
	Angle	0.0000	-31.4074	-20.7963	-23.5523	-32.3734	-47.8005	-25.9168	-22.7464	-29.7045	-28.5912
	Amplitude	0.0212	0.0131	0.0113	0.0212	0.0137	0.0145	0.0135	0.0113	0.0149	0.0125
Leading Blowing	Location	0.6188	0.5757	0.5886	0.0810	0.5981	0.7168	0.6529	0.6197	0.5538	0.6152
	Angle	45.0000	39.0195	26.3684	22.5982	37.4227	27.0611	32.9645	44.2594	43.9457	40.4108
	Amplitude	0.1000	0.1120	0.1150	0.0000	0.1032	0.0761	0.1087	0.1037	0.1056	0.0961
Trailing Blowing	Location	0.8000	0.6935	0.6017	0.8000	0.6234	0.7726	0.7123	0.6210	0.6433	0.6267
	Angle	90.0000	25.4947	43.0920	41.0303	31.5631	40.0945	34.4371	41.4773	29.1941	33.2553
	Amplitude	0.1414	0.0928	0.0747	0.1414	0.0904	0.1115	0.1131	0.0899	0.0861	0.0886
C_l		0.9253	0.9391	0.9422	0.9459	0.9378	0.9424	0.9391	0.9423	0.9353	0.9423
C_d		0.1484	0.1520	0.1527	0.1533	0.1520	0.1528	0.1523	0.1528	0.1518	0.1529
C_l / C_d		6.2366	6.1784	6.1719	6.1705	6.1697	6.1661	6.1659	6.1653	6.1621	6.1612
Fitness		**2.1228**	**2.1124**	**2.1113**	**2.1110**	**2.1109**	**2.1103**	**2.1102**	**2.1101**	**2.1096**	**2.1094**



(a) EARND GA



(b) CGA

Figure 5.11: Fitness curves from EARND GA (a) and CGA (b)

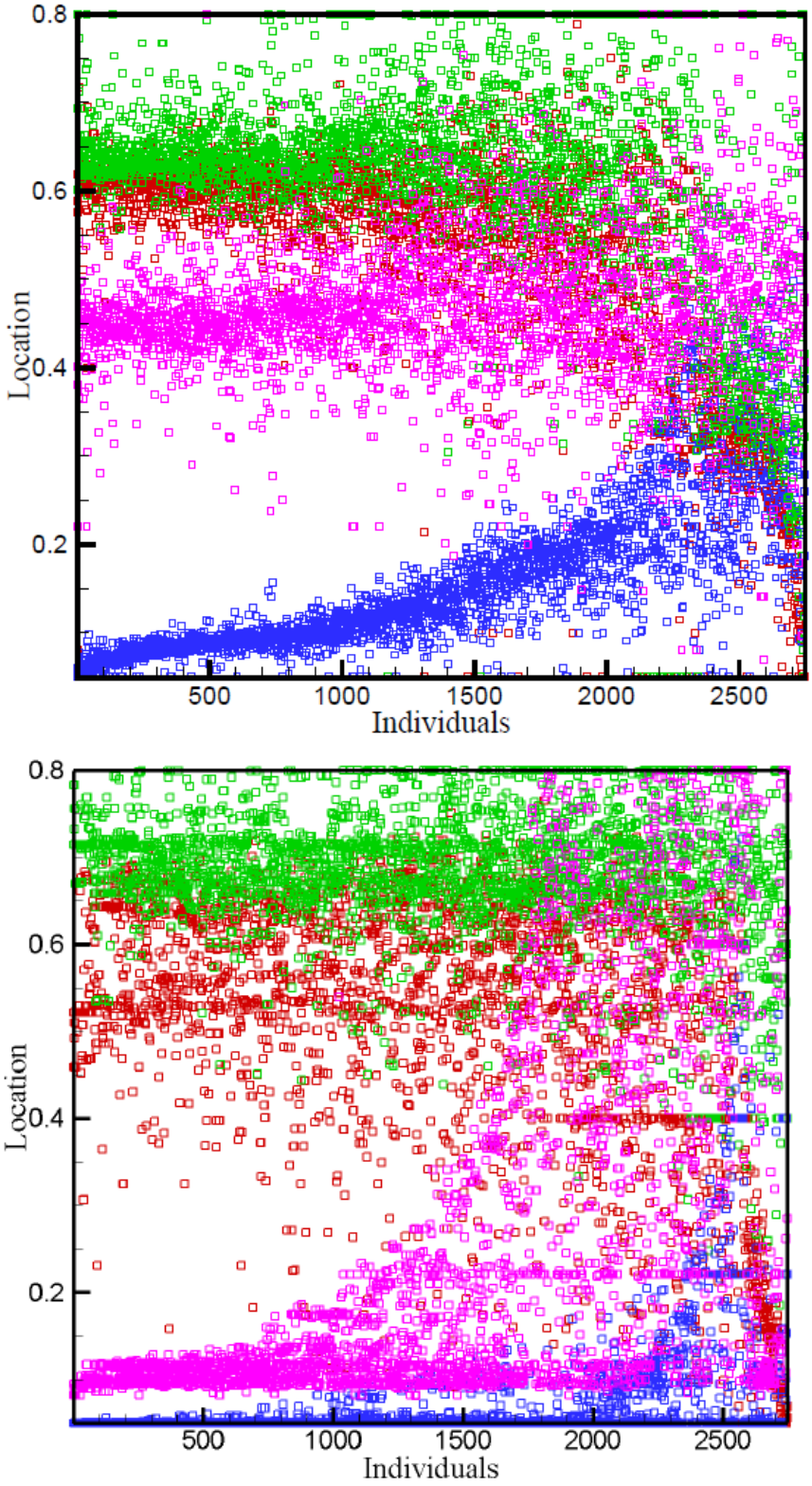


Figure 5.12 (a): Location of jets from EARND GA (top) and CGA (bottom)

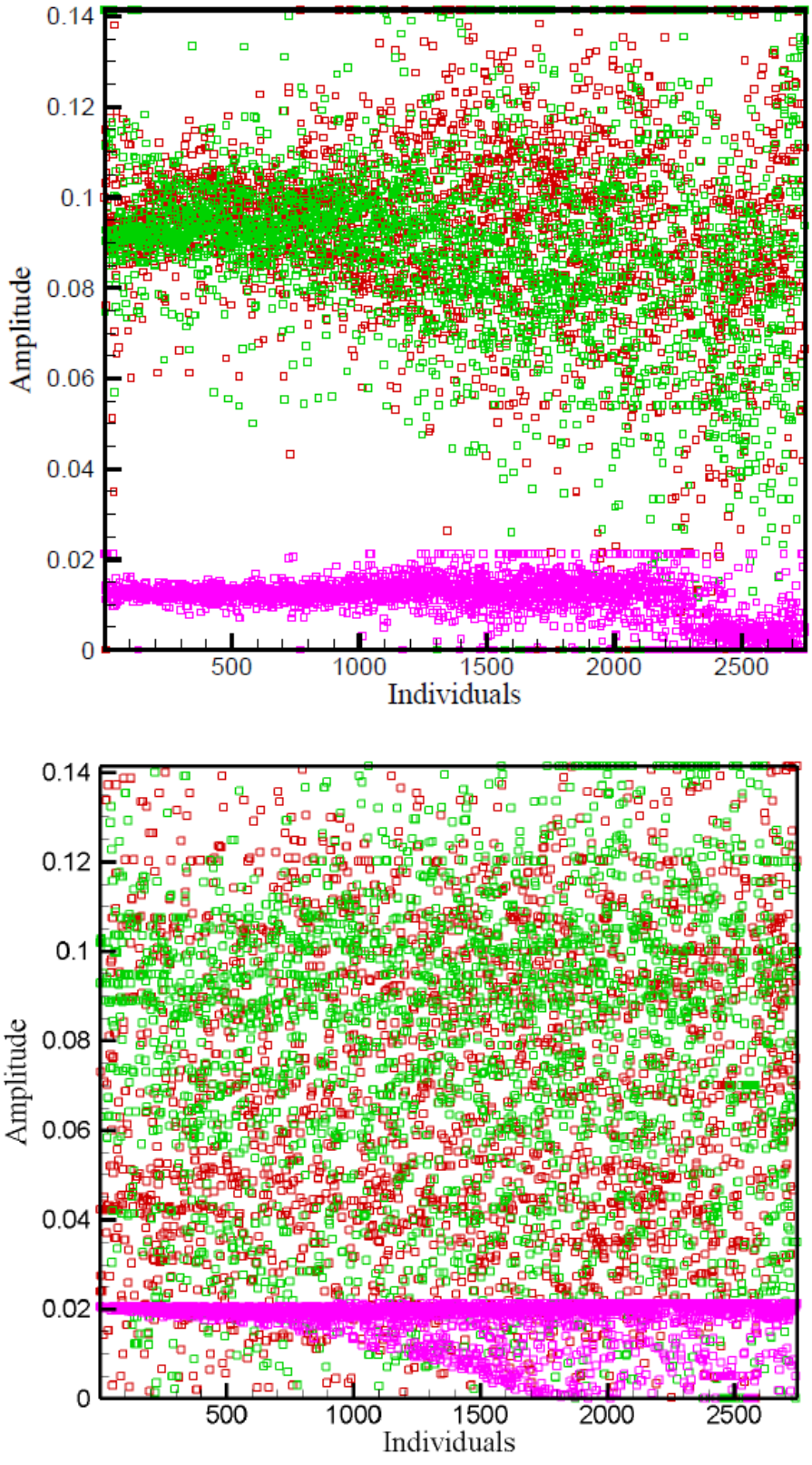


Figure 5.12 (b): Amplitude of jets from EARND GA (top) and CGA (bottom)

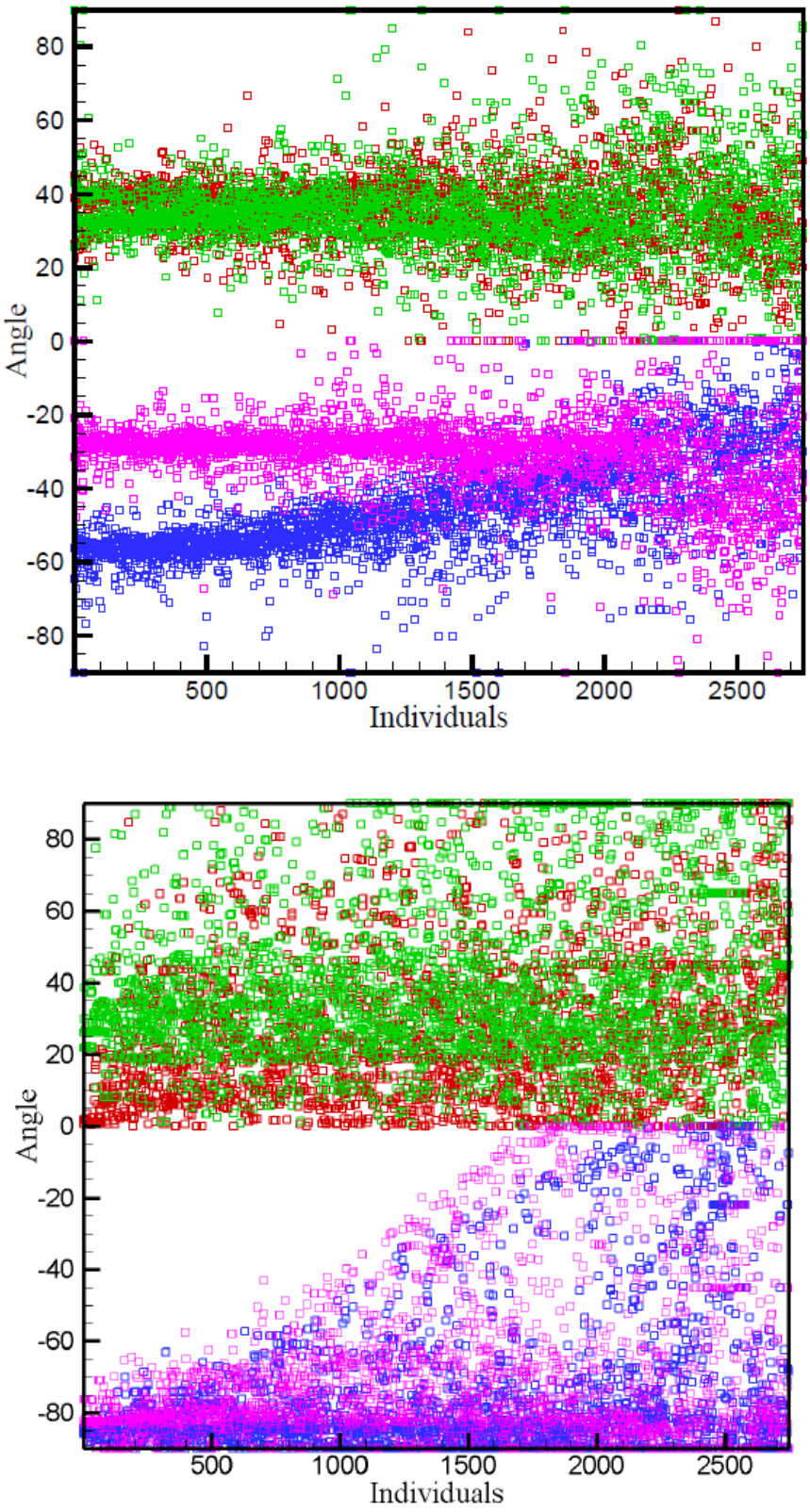


Figure 5.12 (c): Angle of jets from EARND GA (top) and CGA (bottom)

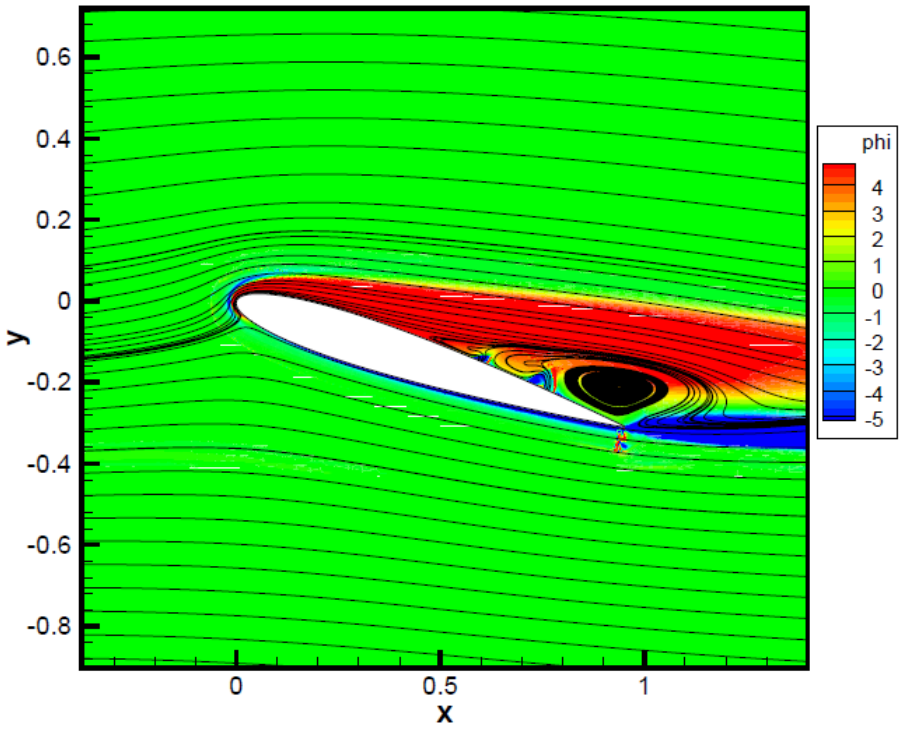


Figure 5.13 (a): Vorticity and streamline plot with EARND GA optimum jet configuration

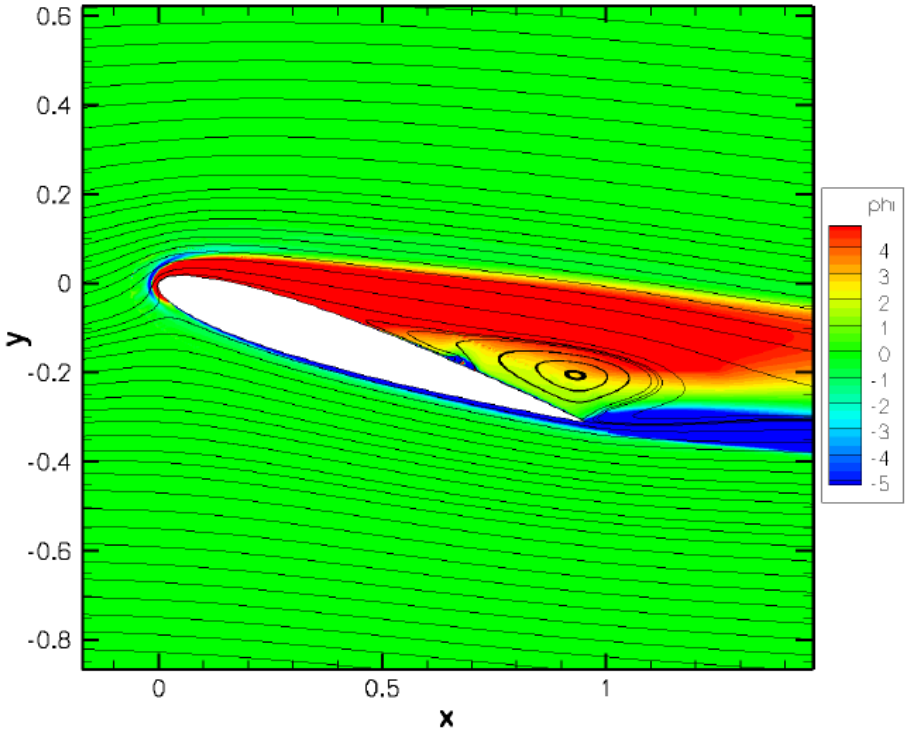


Figure 5.13 (b): Vorticity and streamline plot with CGA optimum jet configurations

5.7 COMBINATION OF CGA AND EARND GA CONFIGURATION

From the previous EARND GA and CGA comparison, it is clear that the EARND GA optimum configuration does not quite push towards having the trailing suction jets at full amplitude, or the angles to near normal, and this reduces the effect of the suction jets on the setup, i.e. the suction jets are not so dominant as they would be if they were at full amplitude and normal angle. However, the failure of the EARND GA to best situate the suction jets appears indirectly to have caused the blowing jets to achieve a greater degree of convergence than in the CGA evolution. As an experiment the suction jet parameters from the CGA best fitness configuration have been combined with the EARND GA best configuration for blowing jets. The resultant fitness, although not much different from the CGA best fitness, is the highest of all. This suggests that the optimum blowing configuration may be better attained from a simulation with further reduced amplitude on the suction jets. The various jet parameters and the fitness obtained by this setup are listed in Table 5.7.

Table 5.7 Jet configuration and fitness using both GA configurations

Leading Suction	0.0500	Trailing Suction	0.0941	Location
	0.0212		0.0206	Amplitude
	-84.3700		-82.2890	Angle
Leading Blowing	0.6187	Trailing Blowing	0.8000	Location
	0.1000		0.1414	Amplitude
	45.0000		90.0000	Angle
Cl	0.9675	**Cd**	0.1448	
Fitness		**2.1970**		

Comparing the vorticity and stream plot of the combined GA configuration with the Continuous GA best configurations (Figure 5.14), it is seen that the blowing jets do a slightly better job in controlling the separation bubble downstream, potentially keeping the flow attached a bit longer relative to the Continuous GA configuration.

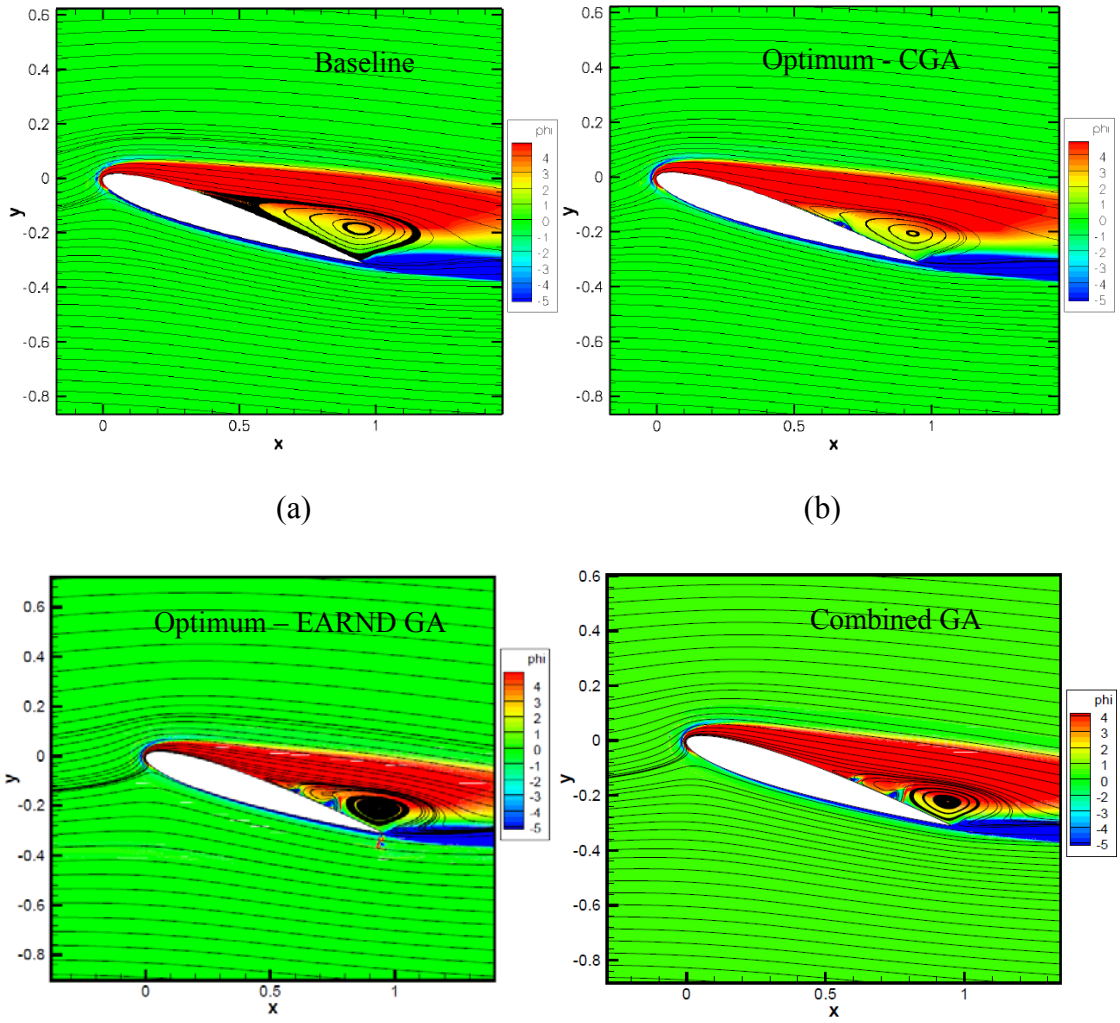


Figure 5.14: Vorticity and streamline comparisons

5.8 SUMMARY

In this chapter we presented the optimized configuration for the steady four jet simulation. The jet parameters were optimized to yield high lift and low drag. Initially the Continuous GA results were discussed, followed by a comparison of the CGA results with the EARND GA results and finally a setup using the results from both the Genetic Algorithms was discussed. The optimum configuration showed a great improvement in the fitness by increasing lift and decreasing drag, which was achieved by the weakening of the separation bubble. It was

also shown that, for the steady four jet case optimization, the Continuous Genetic Algorithm performed better than the previously developed EARND Genetic Algorithm. The configuration setup by combining both Continuous and EARND Genetic Algorithms results achieved the best fitness value as a result of the better optimization of the blowing jet parameters.

CHAPTER – 6

6. UNSTEADY SYNTHETIC JET RESULTS

In the previous chapter we presented the optimized results of the steady four jet control system using both GA approaches. However most active flow control techniques are not steady, but rather rely on unsteady means to control separation. Therefore an unsteady synthetic jet flow control system is developed and the Continuous Genetic Algorithm is applied to optimize the various jet parameters. Initially a pure oscillating two-jet setup was tested and was found not to strongly affect the flow field for the current flow conditions. Thus a hybrid unsteady setup was developed and the various jet parameters were optimized.

6.1 SYNTHETIC JETS

Synthetic jets have emerged as a versatile micro-actuators with potential applications ranging from separation [101] [102] and turbulence [103] control to thrust vectoring [104] and augmentation of heat transfer and mixing [104]. Among all these applications, the use of these devices for active control of separation has been studied quite extensively, and in a number of experimental studies [101] [102], it has been demonstrated that synthetic jets can reduce the extent of separation over bluff as well as streamlined bodies.

Separation over an airfoil is typically an unsteady process that is accompanied by the formation of large-scale vortex structures in the separated shear layer. The characteristic frequency of formation of these vortex structures is $O(U_\infty / L_s)$ where L_s is the length of the separation zone and U_∞ the freestream velocity. There is broad consensus [101] [106] that synthetic jets operating in this frequency range tend to promote and amplify the formation of the vortex structures in the separation region. These vortex structures entrain high momentum, freestream fluid into the separated flow region and this promotes the early reattachment of the separated boundary layer [107]. In the case where the boundary layer is laminar at separation,

synthetic jets operating at much higher frequencies could also lead to earlier transition in the boundary layer. Since a turbulent boundary layer is more resistant to separation, earlier transition to turbulence can delay the separation.

Figure 6.1 shows a 2D synthetic jet simulation. The slot along with the oscillating frequency forms a resonator. As the oscillations are applied, fluid is periodically entrained into and expelled from the orifice, essentially creating a zero-net mass flux jet. During the expulsion portion of the cycle, a vortex ring can form near the orifice and, under certain operating conditions, convect away from the orifice to form a time-averaged jet.

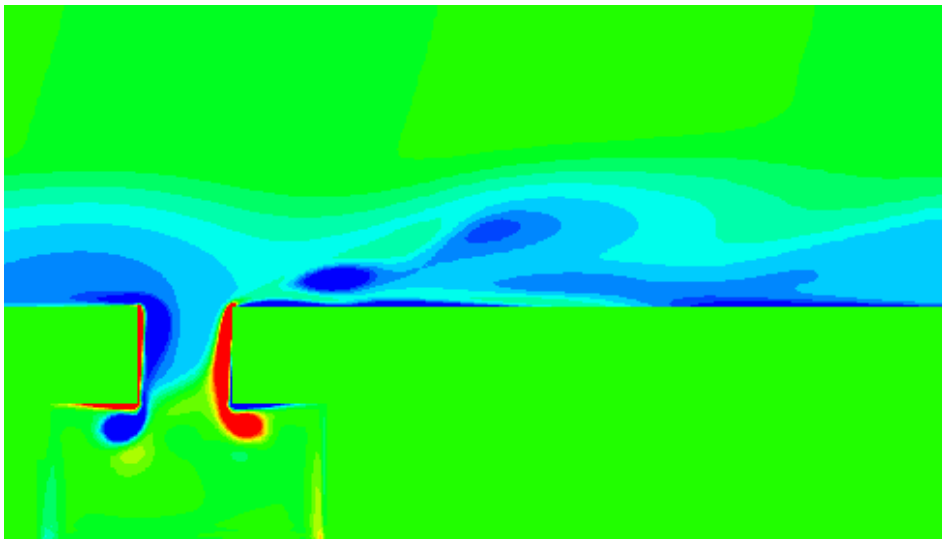


Figure 6.1: A 2D synthetic jet interacting with a laminar boundary layer [108]

One of the important criterion that in considered while studying synthetic jets is the dimensionless Strouhal Number (*St*), which for a fixed oscillation frequency jet varies as a function of, $St \; \alpha \; \frac{f\,L}{U} \; \alpha \; \frac{V}{d^3}$, where *V* is the orifice volume and *d* is the slot width.

6.2 CASE SETUP

The flow under consideration is same as in the case of the four jet case. The unsteadiness is introduced by using an oscillatory frequency with which the synthetic jets oscillate. As

previously stated, the basic two-dimensional grid (Figure 5.1-a) consists of 11 background blocks in a three-by-three pattern, the central region consisting of three subgrids over which the airfoil grid is placed (Figure 5.1).

The airfoil grid used for this case also consisted of 5 blocks and fine grid spacing was employed on them relative to the background blocks, but it was different from the one used in the steady case. This is because the jet width required for a typical synthetic jet is much smaller than the steady jets as the synthetic jets that are typically used have much more design constraints and auxiliary power requirements than the steady jets. Therefore, the upper airfoil block where the jets are placed uses a stretching grid type mechanism, i.e. the grid narrows into finer grid spacing around the jet locations as shown in Figure 6.3. The part of the code in FlexGrdi.f90 that generates this stretch grid is listed in Appendix A.3. This ensures that we have an adequate number of grid points to capture the relatively small, high activity region with reasonable accuracy. It also considerably reduces the total number of grid points as we no longer use the resolution equal to the jet resolution on the whole upper block of the airfoil gird, thereby making the setup computationally less intensive. The jet width used for these jets is 0.4% of the chord length, sufficiently small to represent the selected frequency range. The *x* spacing used for the jet region is 0.0004, thus we have 10 grid points to represent the jet width. The boundary conditions used are same as the steady four jet case and are detailed in section 5.1. With the introduction of the frequency as a function of time, the *u* and *v* velocity equations at each of the jet boundary are given by,

$$\begin{aligned} u(i,j) &= A_1 \times U_\infty \times \cos(\beta) \times \sin(2 \times \pi \times f_1 \times t + \phi) \\ v(i,j) &= A_1 \times U_\infty \times \sin(\beta) \times \sin(2 \times \pi \times f_1 \times t + \phi) \\ \text{where, } \beta &= (Baseangle + \theta) \end{aligned} \tag{6.1}$$

The CFD parameters used to run this unsteady simulation are listed below.

CFD Parameters:

timestep: 0.005

sub-iterations: 10

total non-dimensional timesteps: 20

total iterations: 40000

The timestep is selected such that the chosen frequency range could be reasonably represented by the simulation. Figure 6.2 presents the lift and drag plots of the CFD simulation. The oscillating effect of the synthetic jets could be clearly seen in these plots and also we have a reasonably good convergence at the end of 20 non-dimensional timesteps. For the fitness calculations, the lift and the drag values are averaged from the last 1500 outputs of the simulation.

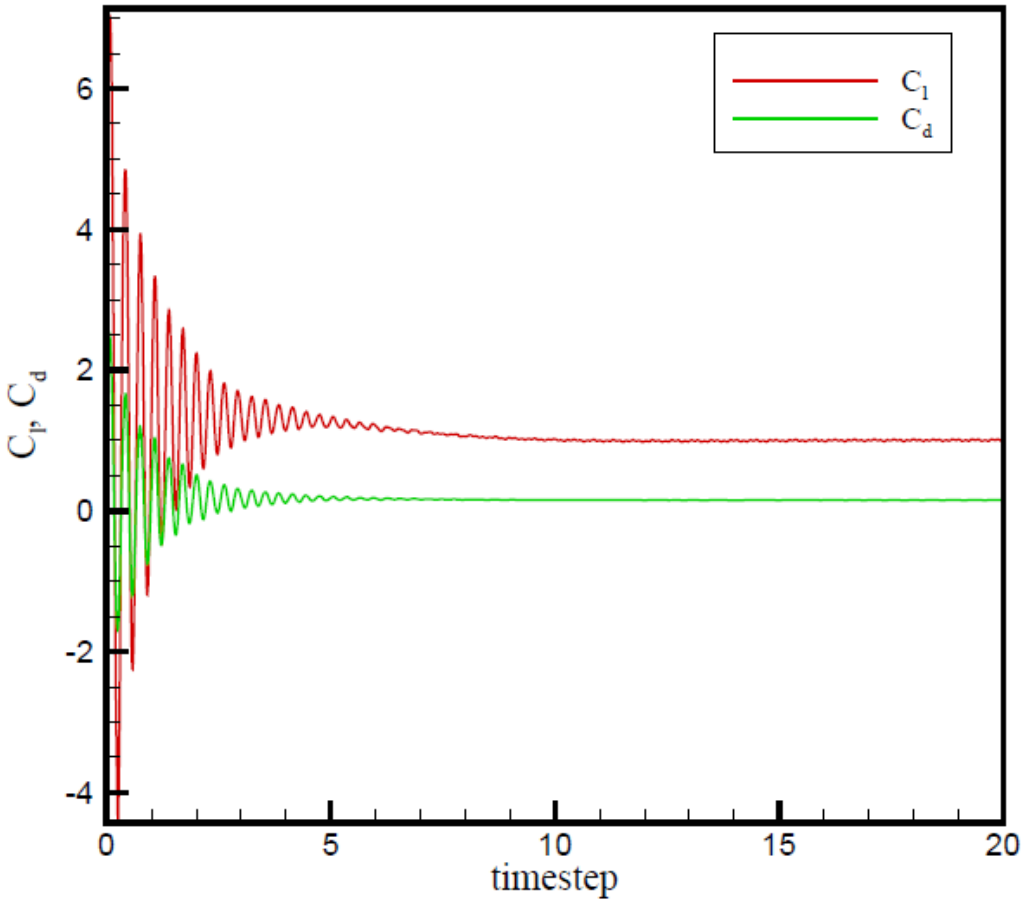
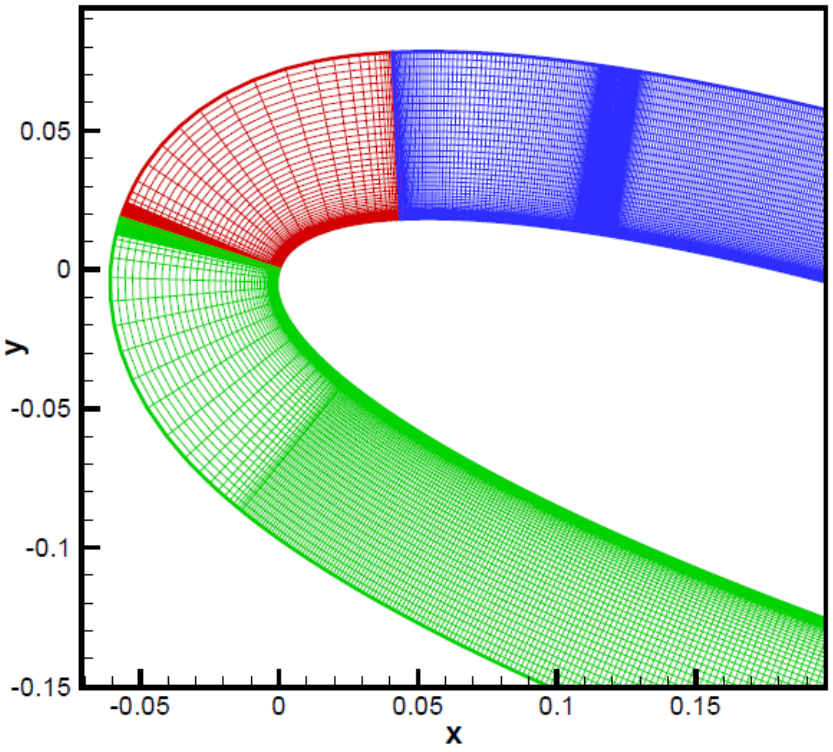
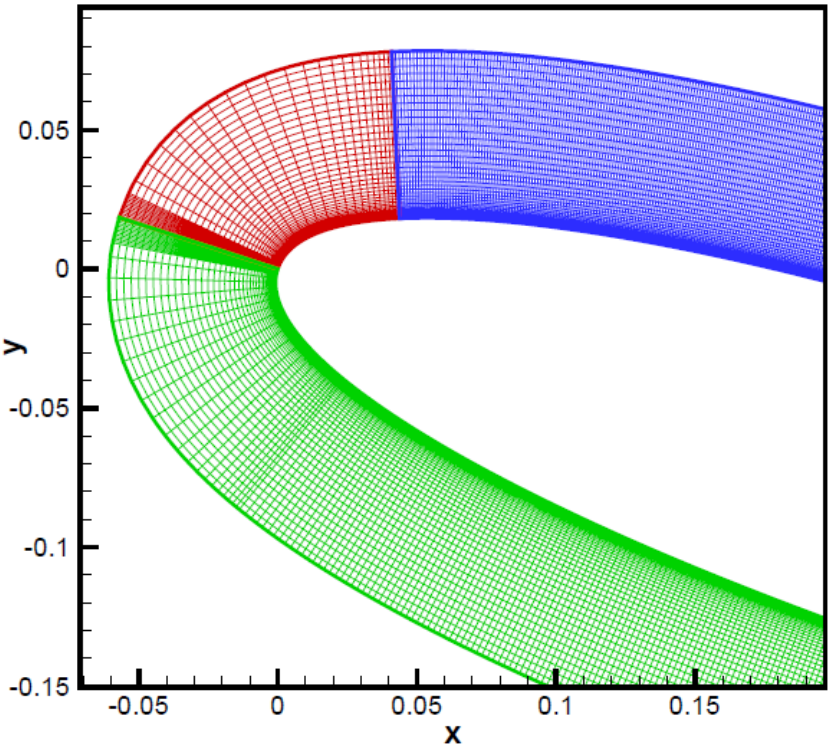


Figure 6.2: Lift and drag convergence for the unsteady synthetic jet case



(a)



(b)

Figure 6.3: Comparison of unsteady (a) and steady (b) grid setup

6.3 PARAMETER SELECTION

The parameters considered for optimization for the pure oscillating case are the jet location L_j, the jet amplitude A, the jet angle θ, the jet frequency f^* and the phase shift ϕ for one the jets . The range of these parameters is listed in Table 6.1. The location essentially takes the same range, but as we are using a stretched grid, the limits change as we account for the tapering before we actually reach the jet location. The angles on the other hand have a bigger range. Previously [20], pure suction and pure blowing were represented by negative and positive angles respectively, and varied from 0^0 to 90^0. As the synthetic jets perform both suction and blowing, we are now exploring a wider range of angles for both the jets, i.e. from 0^0 to 180^0 (Table 6.1), which covers all the angles for the current 2D flow case. A 0^0 angle implies actuation in the same direction as the flow field and tangential to the surface of the airfoil. 90^0 implies perpendicular actuation into the flow field and 180^0 means actuation tangential to the surface of the airfoil but in the direction opposite to the flow direction. The non-dimensional frequency listed in Table 6.1 is evaluated based on the flow. At *Re* = 500,000 (current flow condition) and assuming an altitude of about 20 km, the freestream velocity is around 100 m/s. The non-dimensional frequency is defined as,

$$f^* = \frac{f\,L}{U} \tag{6.2}$$

f^* = Non-dimensional frequency
f = Actual frequency in *Hz*
L = Chord length equal to 1.0
U = Freestream velocity

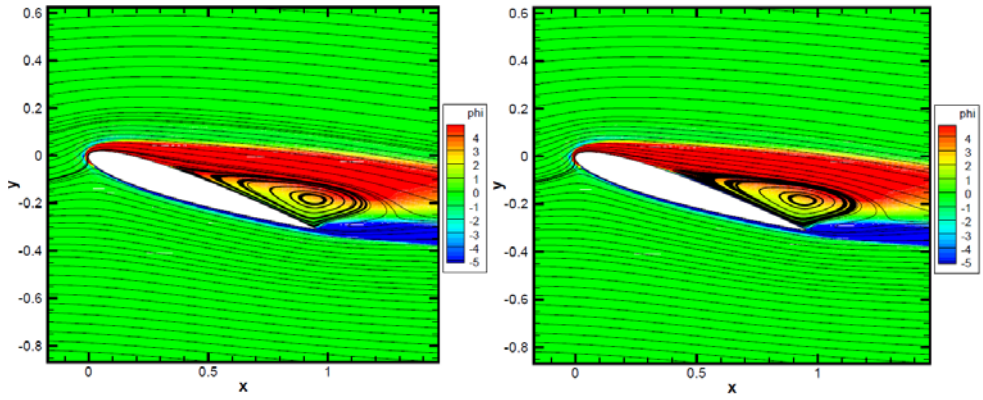
So an f^* of 1 to 10 would yield an actual frequency range of 100 to 1000 *Hz*. This frequency range has been experimentally tested for many synthetic jet setups and is known to affect the flow field. The frequency range used for this setup is 0.1 to 10.0 (non-dimensional). The amplitude of the jets is selected in such a way as to maintain the minimum momentum flux to

affect the flow field [54]. As we are using a relatively small jet width this range is considerably larger than the one used for the steady case (Table 6.2). The phase shift ϕ is varied from 0 to 3.14 radians, allowing one of the jet oscillations to vary from in-phase to completely out of phase relative to the other jet.

Table 6.1: Parameter range for the pure oscillating case

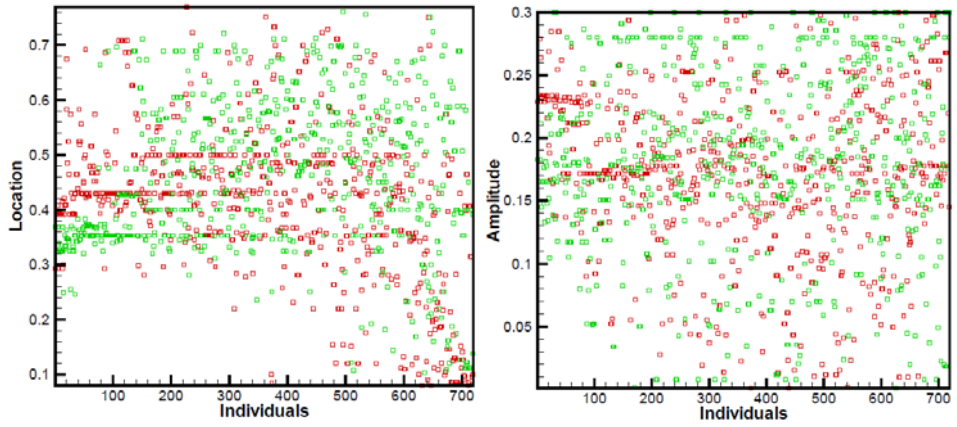
Variable Name	***Range***
Jet location –1 (L_{j1}):	$0.08 \leq L_{j1} \leq 0.77$
Jet angle – 1 (θ_1):	$0^o \leq \theta_1 \leq 180^o$
Jet amplitude – 1 (A_1):	$0.0 \leq A_1 \leq 0.30$
Jet frequency – 1 (f^*_1):	$0.1 \leq f^*_1 \leq 10$
Jet Location – 2 (L_{j2}):	$0.08 \leq L_{j2} \leq 0.77$
Jet angle – 2 (θ_2)	$0^o \leq \theta_2 \leq 180^o$
Jet amplitude – 2 (A_2):	$0.0 \leq A_2 \leq 0.30$
Jet frequency – 2 (f^*_2):	$0.1 \leq f^*_2 \leq 10$
Phase Shift – (ϕ)	$0.0 \leq \phi \leq 3.14$

The effect of these pure oscillating jets on the flow field are shown in Figure 6.4 and the CGA results for this case are shown in Figure 6.4. The flowfield plots (Figure 6.4) of baseline and the final configurations (after 24 generations with 30 individuals per generation) are visually indistinguishable, suggesting that the jets are not affecting the separation bubble. The scatter plots (Figure 6.5) presents the CGA results from 24 generations of this case. It is clear that the GA has sufficiently covered the parameter range and still the configurations do not seem to affect the flow field. The best configuration obtained using this setup has a fitness of 2.0099, barely higher than the baseline fitness of 2.0. This suggests that the pure oscillating jets may not be an ideal means of flow control for this particular case. Therefore a hybrid unsteady case was setup. To achieve this we introduce two more parameters into the setup, viz. the amplitude shift (a) for both the jets.

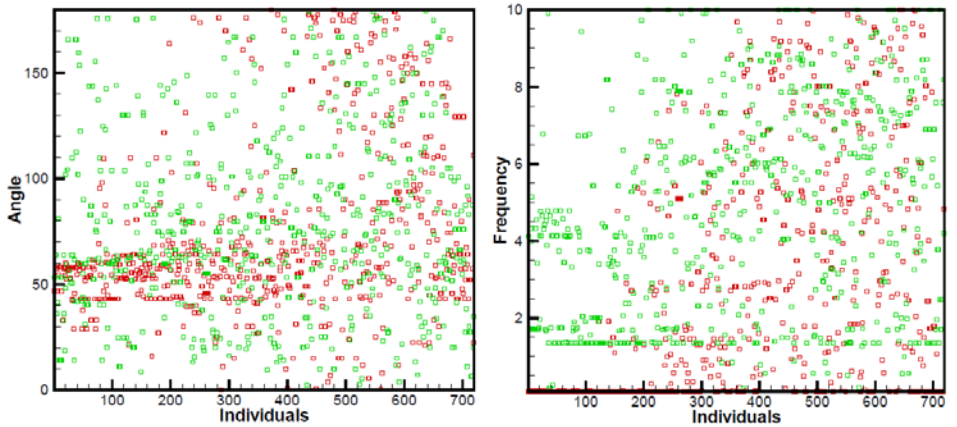


Baseline Final

Figure 6.4: Vorticity and streamline plot for pure unsteady two-jet case



(a) (b)



(c) (d)

Figure 6.5: Results of the pure oscillating unsteady two-jet case (24 generations)

The amplitude shift varies in the range listed in Table 6.2. This shift in the amplitude intermittently turns the jets into pure suction or pure blowing (depending on the sign of the shift) along the unsteady oscillations. With the introduction of the amplitude and phase shift the velocity equations at the jet inlet are given by

Jet-1, only amplitude shift:

$$u(i,j) = A_1 \times U_\infty \times \cos(\beta) \times \sin(2 \times \pi \times f_1 \times t) + a_1 \times U_\infty \times \cos(\beta)$$
$$v(i,j) = A_1 \times U_\infty \times \sin(\beta) \times \sin(2 \times \pi \times f_1 \times t) + a_1 \times U_\infty \times \sin(\beta)$$

Jet-2, both amplitude and phase shift: (6.3)

$$u(i,j) = A_2 \times U_\infty \times \cos(\beta) \times \sin(2 \times \pi \times f_2 \times t + \phi) + a_2 \times U_\infty \times \cos(\beta)$$
$$v(i,j) = A_2 \times U_\infty \times \sin(\beta) \times \sin(2 \times \pi \times f_2 \times t + \phi) + a_2 \times U_\infty \times \sin(\beta)$$

Table 6.2: Modification of variables for hybrid unsteady case

Variable Name	***Range***
Jet amplitude shift – 1 (a_1):	$-0.20 \le a_1 \le 0.30$
Jet amplitude shift – 2 (a_2):	$-0.20 \le a_2 \le 0.30$
Jet frequency – 1 (f^*_1):	$2 \le f^*_1 \le 10$
Jet frequency – 2 (f^*_2):	$2 \le f^*_2 \le 10$

The range of the amplitude shifts is again selected in such a way that, when we there is pure suction or blowing the momentum flux is at least the minimum flux required to effect the flowfield [54]. The frequency for this case is varied from 2.0 to 10.0 (non-dimensional). With the introduction of the amplitude shift for both the jets, we now have 11 parameters (Table 6.1 and 6.2) which are to be optimized for the hybrid unsteady system. The Continuous GA is used to optimize these parameters and the results obtained are discussed in the next section.

6.4 GENETIC PARAMETERS

As previously mentioned we have 11 parameters that need to be optimized. The Continuous GA is applied to optimize these parameters and the genetic coefficients are set as

$$NGeneration = 50$$
$$NPopsize = 30$$
$$NVariable = 11$$
$$Mutation\,Const. = 0.15$$
$$Crossover\,Const. = 0.50$$

The equation of the aggregate fitness is given by,

$$\left(Fit_A\right)_{\max} = a.C_l / C_{lB} + b.C_{dB} / C_d \tag{6.4}$$

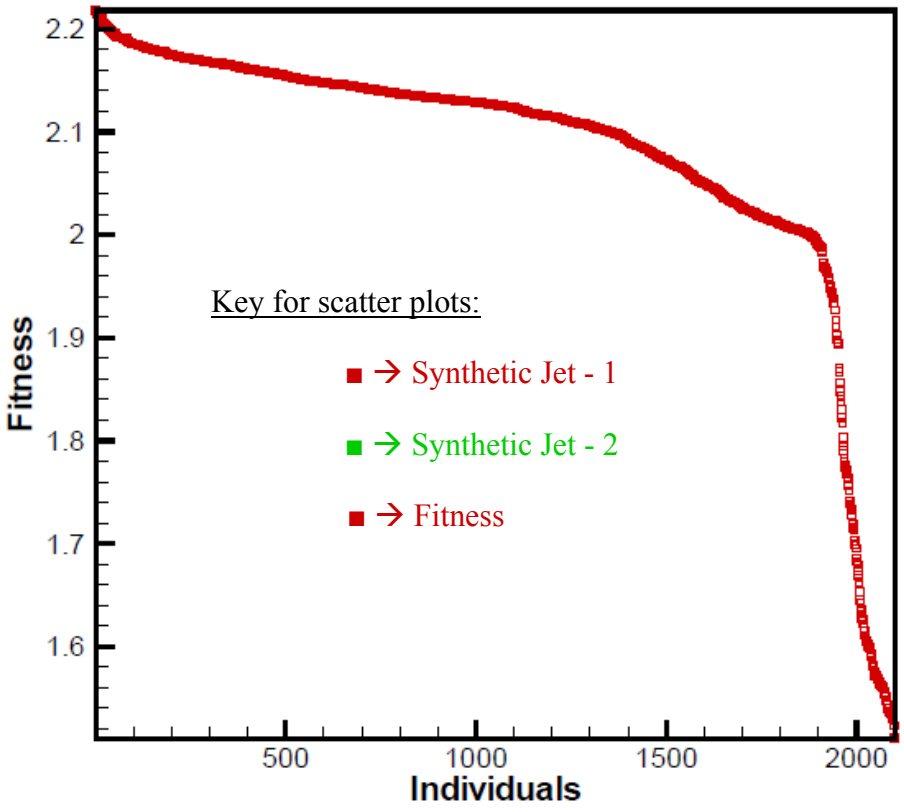
a and b are set to 1, providing equal weight to both lift and drag, and making the baseline fit equal to 2.0.

6.5 HYBRID UNSTEADY TWO JET RESULTS

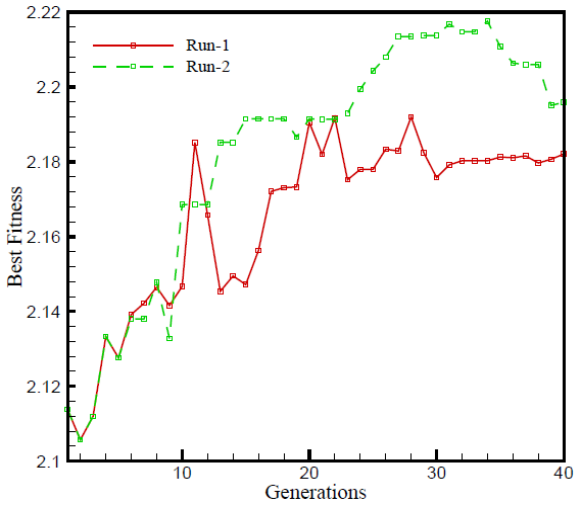
Two simulations of the hybrid unsteady two-jet case have been performed using the Continuous GA. Run-1 consisted of 40 generations with 30 individuals per generation and run-2 for which the initial population was the output of generation-5 of run-1, evolved for 35 generations with 20 individuals per generation. The values of the baseline lift and drag (C_{lB} and C_{dB}) were determined from simulations of the base airfoil (without jets) and were fixed at 0.8881 and 0.1601 respectively. The initial population for run-1 was selected such that, the individuals cover the full allowed parameter range. Figure 6.5 (i) presents the fitness plots of the hybrid unsteady two-jet simulations. The overall maximum fitness obtained from both the evolutions was 2.2176, much higher than the fitness of 2.009 that was obtained by the pure unsteady case.

Table 6.3: Best 5 individuals of the unsteady 2-jet case – CGA

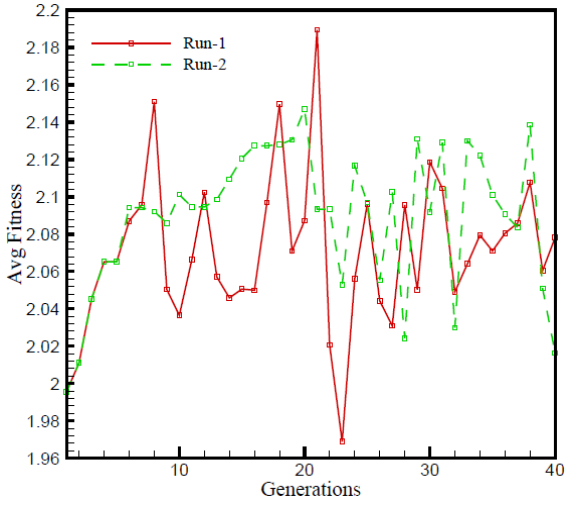
Jet - 1	Location	0.094300	0.094300	0.094300	0.094300	0.094300	0.094300	0.094300	0.094300	0.094300	0.094300	0.094300
	Amplitude	0.104060	0.083170	0.104060	0.083170	0.104060	0.083170	0.083170	0.148140	0.083170	0.083170	0.083170
	Angle	82.650945	82.650945	82.650945	82.650945	82.650945	82.650945	82.650945	82.650945	82.650945	82.650945	82.650945
	Frequency	2.401410	2.401410	2.401410	2.401410	2.401410	9.355040	7.154420	2.401410	8.130060	2.401410	2.401410
	Amp. Shift	-0.189851	-0.189851	-0.189851	-0.189851	-0.189851	-0.189851	-0.189851	-0.189851	-0.189851	-0.189851	-0.189851
Jet - 2	Location	0.134080	0.134080	0.134080	0.134080	0.134080	0.134080	0.134080	0.134080	0.134080	0.134080	0.134080
	Amplitude	0.035010	0.035010	0.035010	0.035010	0.035010	0.177140	0.155110	0.035010	0.126760	0.035010	0.035010
	Angle	96.086303	96.086303	96.086303	96.086303	96.086303	96.086303	96.086303	96.086303	96.086303	96.086303	96.086303
	Frequency	8.850050	6.685430	6.685430	6.685430	4.282250	6.685430	6.685430	6.685430	3.008830	5.726040	
	Amp. Shift	-0.198125	-0.198125	-0.198125	-0.198125	-0.198125	-0.198125	-0.198125	-0.198125	-0.198125	-0.198125	
	Phase Shift	2.462290	1.397710	1.397710	2.734960	2.668160	1.397710	1.397710	1.397710	1.397710	1.397710	
C_l		1.023556	1.022472	1.022541	1.022696	1.022477	1.021466	1.020660	1.021288	1.020799	1.021596	
C_d		0.150265	0.150204	0.150500	0.150602	0.150586	0.150469	0.150384	0.150499	0.150441	0.150724	
C_l / C_d		6.811680	6.807240	6.794300	6.790710	6.789980	6.788560	6.787040	6.786000	6.785400	6.777920	
Fitness		**2.217631**	**2.216844**	**2.214824**	**2.214276**	**2.214143**	**2.213833**	**2.213528**	**2.213417**	**2.213282**	**2.212177**	



(i)

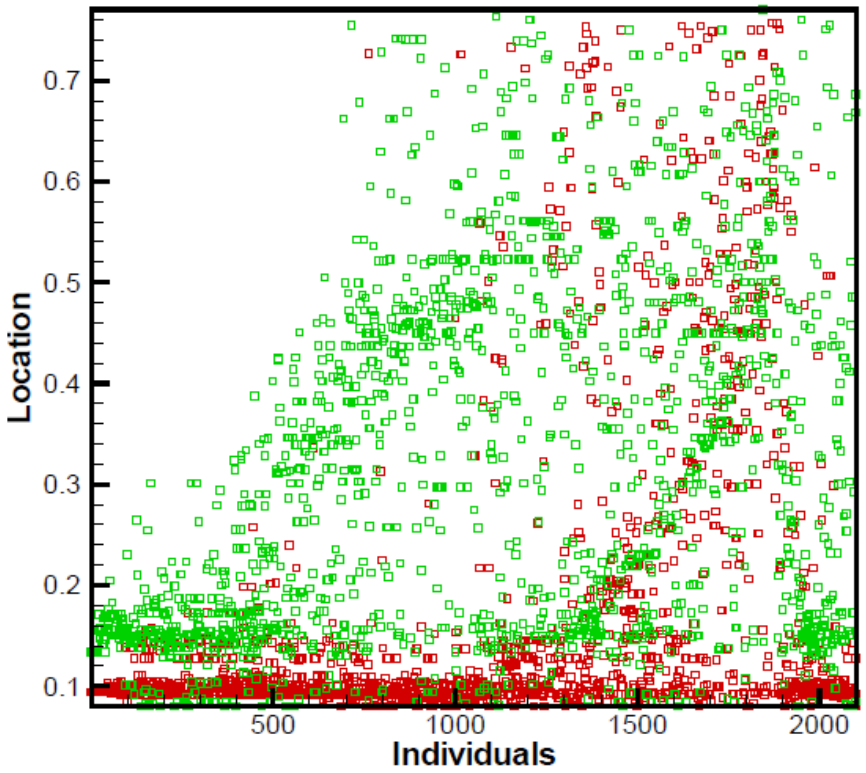


(ii)

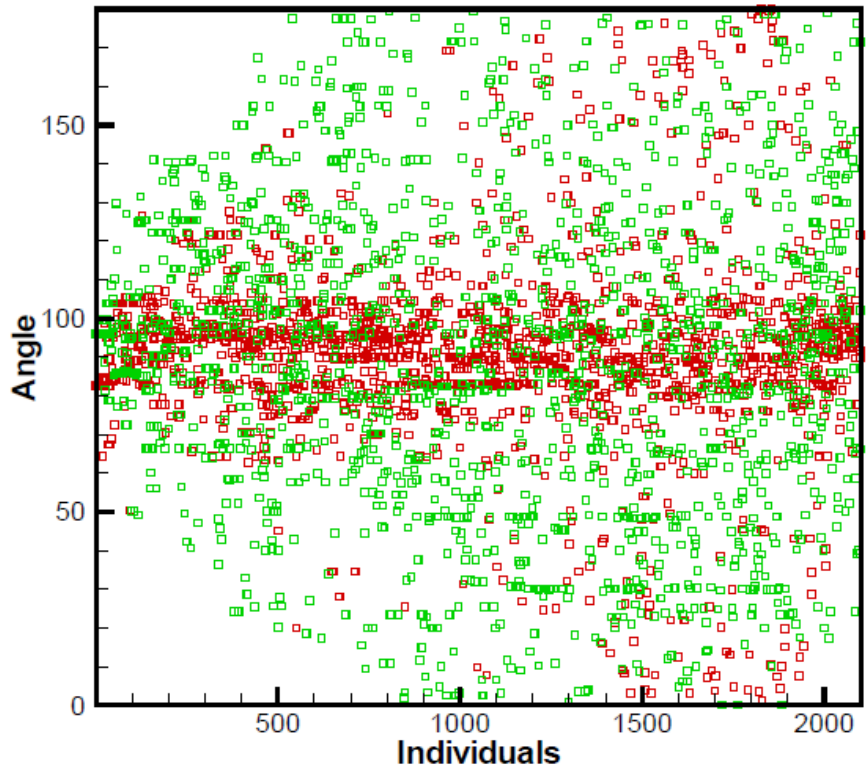


(iii)

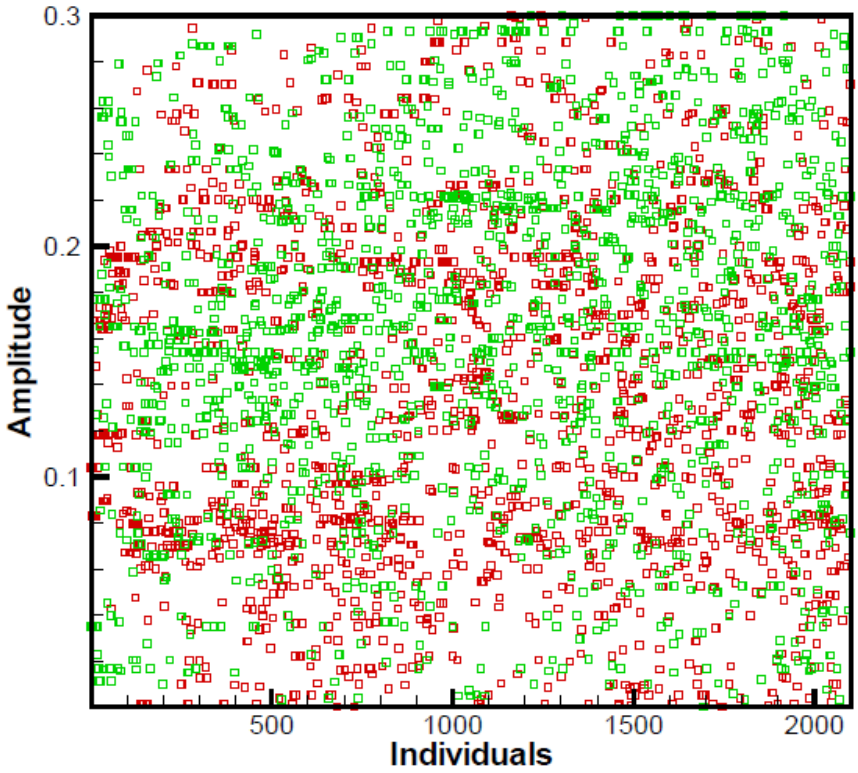
Figure 6.6: Fitness plots of unsteady simulations


Location
0.7
0.6
0.5
0.4
0.3
0.2
0.1
500
1000
1500
2000
Individuals

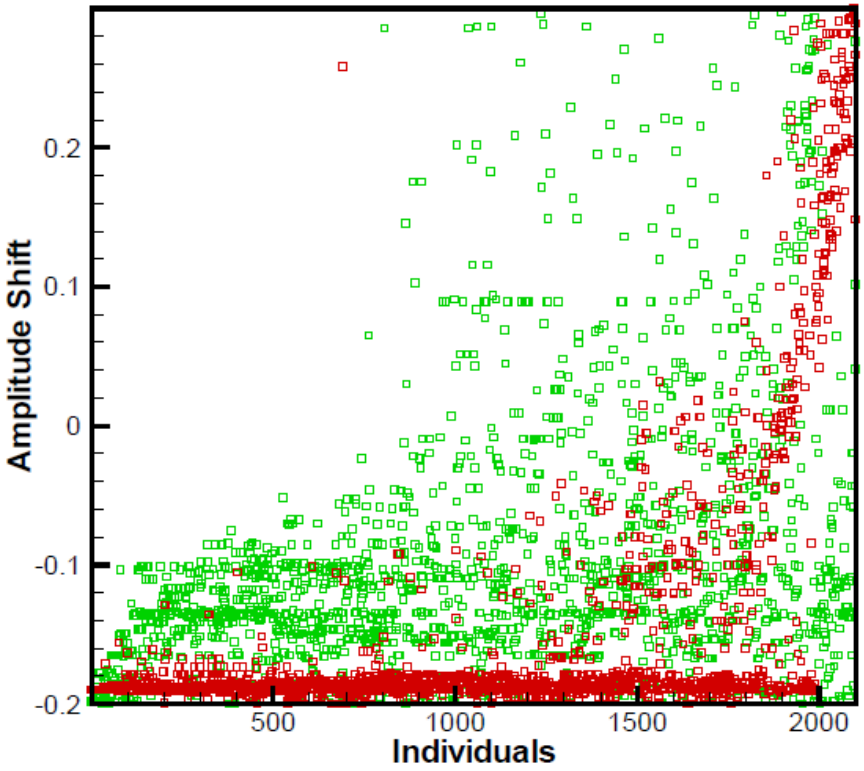
(a)


Angle
150
100
50
0
500
1000
1500
2000
Individuals

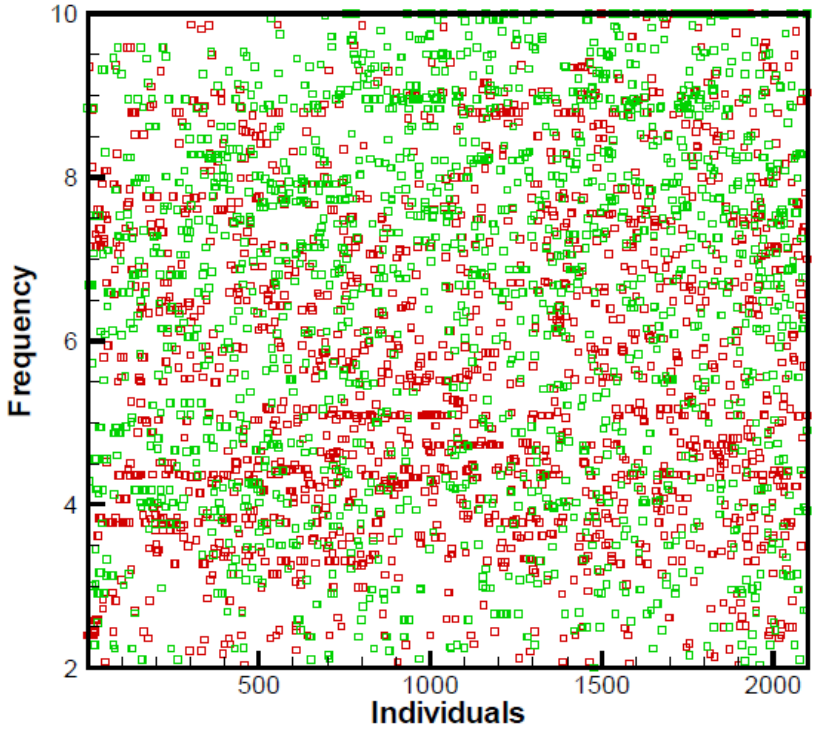
(b)


0.3
0.2
0.1
Amplitude
500
1000
1500
2000
Individuals

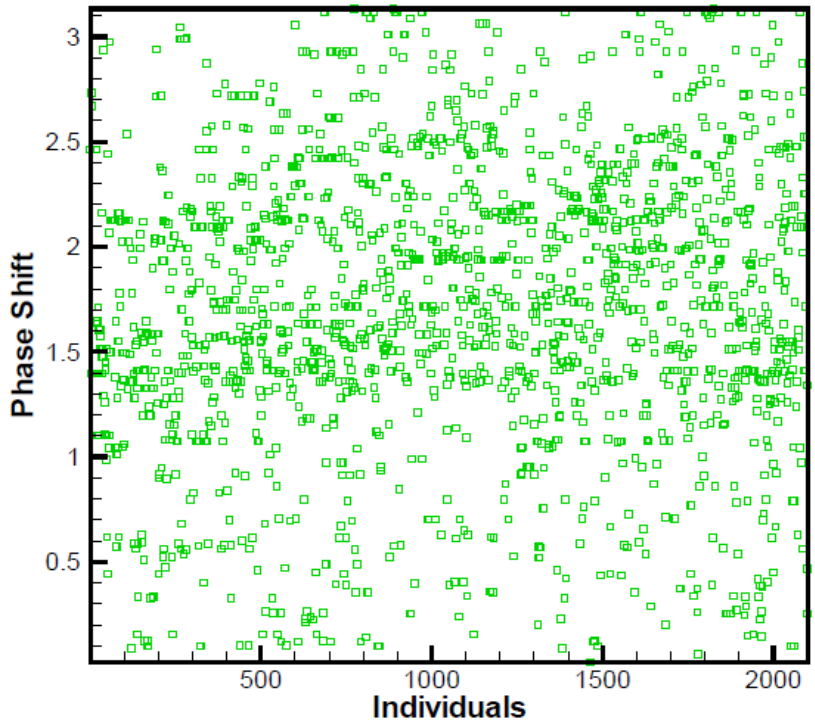
(c)


0.2
0.1
0
-0.1
-0.2
Amplitude Shift
500
1000
1500
2000
Individuals

(d)



(e)



(f)

Figure 6.7: Scatter plots for parameters of the unsteady case – CGA (both simulations)

It is important to make sure that the jets are not turning into a pure suction or pure blowing case. Table 6.3 lists the best 10 configurations obtained using the simulations. It is clear from this table that we have sufficient magnitude of unsteady jet amplitude and also a reasonably high frequency in the best fit individuals, which shows that we should have significant unsteadiness in the flow. Figure 6.6 (i) is the overall fitness plot of both the simulations and clearly we have a better fitness than the steady four-jet case. To compare the trends and the search path of the GA in both the runs we compare the best fitness and average fitness (Figure 6.6 (ii)-(iii)) of both the runs. In run-1 after the 20^{th} generation the GA seems to have moved into a not so fit region relative to the run-2, thus yielding a lesser fitness than run-2.

Figure 6.7 presents the scatter plots of all the parameters for both GA simulations. Clearly the first jet location is close to the leading edge at about 10% chord (Figure 6.7(a)), potentially acting as a suction jet of the two jet case but still having sufficient unsteady jet amplitude and frequency. The angle (Figure 6.7(b)) on the other hand seems to be moving towards near normal, although not as converged as the location, it appears to be between 90^0 and 100^0. The frequency and amplitude (Figure 6.7(e) and (c)) of this jet vary over the parameter range with small concentrations in few places, but these do not seem to have a large effect on the overall fitness, thus suggesting that these parameters are less critical compared to the location and angle of the jet. The amplitude shift (Figure 6.7(d)) is clearly biased towards negative maximum (-0.20), suggesting a pure suction type shift.

The second jet location is not as converged as the first jet location but is clearly being pushed towards the leading edge and is behind the first jet at about 15% chord. The angle of this jet is again evidently moving towards near normal, i.e. around 90^0 to 110^0 a bit higher than the first synthetic jet. The amplitude and frequency are behaving in a similar manner as with synthetic jet-1. Amplitude shift of this jet also seem to favor negative maximum and is ranging

between -0.1 and -0.2. The phase shift (Figure 6.7(f)) which is applied only on this jet is also hovering all over the parameter range with small concentrations in some region. Interestingly most of the high fit cases seem to float between 1.0 and 2.5 suggesting significant phase shift for the second jet.

6.6 SUMMARY

The best configuration for the hybrid synthetic-jet case with two jets is clearly inclined towards placing both the jets close to the leading edge of the airfoil. The angles of both the jets are moving towards near normal and the shifts in amplitude seem to be favoring towards negative maximum suggesting a pure suction type mechanism. The frequency, amplitude, and phase shift are not so critical as compared to the location, angle, and amplitude shift. Also, from the scatter plots is clear that the Continuous GA has successfully found the high fitness region for the hybrid synthetic jet and that it performed a reasonably good job in optimizing an unsteady flow control system.

CHAPTER – 7

7. GA-NEURAL NETWORKS-CFD

The results from the current steady four-jet and the unsteady two-jet system along with the previous steady two-jet results show that a GA-CFD system is a means to reasonably narrow down the parameter search space and predict a near optimum configuration of a complex flow control system. However as we move into the regime of more challenging flow control systems the CFD evaluations of the numerous configurations generated by a Genetic Algorithm would be excessively computationally expensive and time consuming.

A possible means for accelerating the evaluation process within a search algorithm is to replace some of the CFD computations with a neural network (NN). The neural network is non-linear means of interpolation that can take the same configuration input parameters as the CFD model and yield the same aggregate outputs, such as lift and drag. The neural network initially requires training and testing for the given problem, the data for which will be provided by current GA-CFD systems. Since neural networks do not actually solve the Navier-Stokes equations, the proposed optimum performance regions determined by the GA-NN system need to be confirmed through CFD simulation and, ideally, ultimately through experiment. However, as proposed, the NN approach will replace most of the CFD computations for a majority of the generations of the GA, which would dramatically reduce the computational cost without sacrificing final accuracy. The NN can also be used to test GA design with a realistic, but far less costly fitness evaluation, leading to improved GA design for both NN and CFD evaluations.

7.1 NEURAL NETWORKS

Fitness evaluation remains the primary cost of both the GA optimization approaches used in current research, as each evaluation of an individual requires a costly CFD simulation. While the computations considered in this research are on the order of CPU hours, not days, this is still

quite expensive when thousands of simulations are required. An alternative to the CFD computation of every fitness are to use interpolation schemes; however, these must be applied with care as a GA approach fundamentally assumes a multi-dimensional solution space with multiple local maximum and minimum which need to be maintained, not blurred out by poor interpolation. A potentially better option is to use a non-linear approach such as a neural network.

The field of Neural Networks (NN) has arisen from diverse sources. Applications range from machine learning to modeling and the prediction of complex relations. Generally, a NN consists of layers of interconnected nodes (Figure 7.1), each node producing a non-linear function of its input.

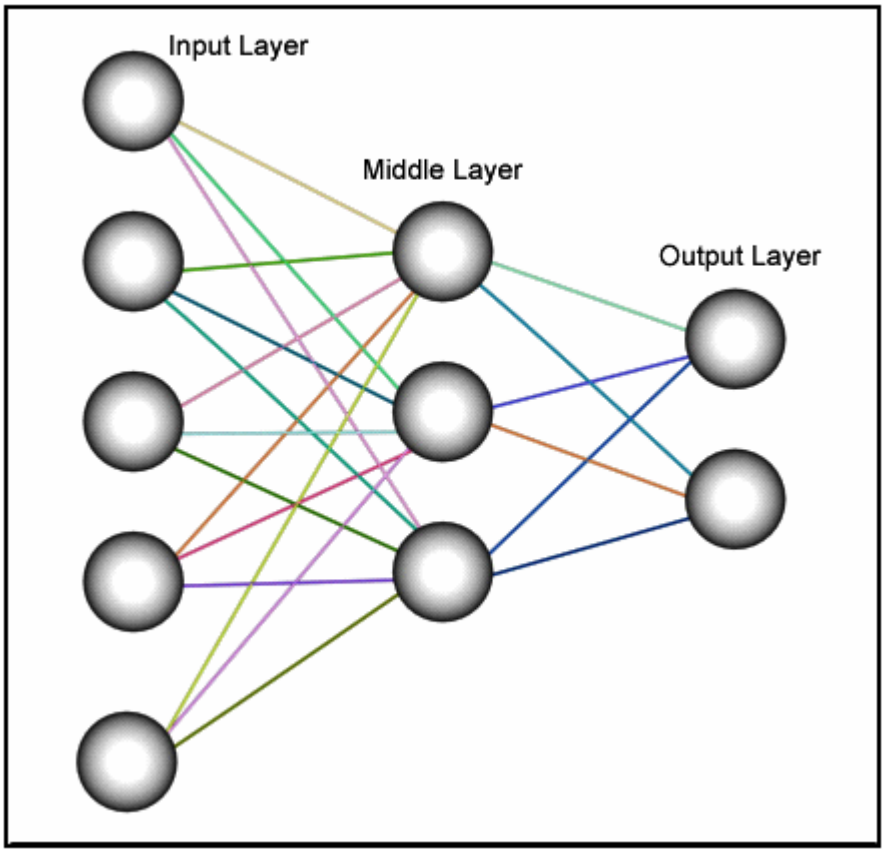


Figure 7.1: A typical neural network

The input to a node may come from other nodes or directly from the input data. Also, some nodes are identified with the output of the network. The complete network therefore represents a very complex set of interdependencies which may incorporate any degree of nonlinearity, allowing very general functions to be modeled.

7.2 GA-NN-CFD SYSTEM AND INITIAL RESULTS

As part of the current research in collaboration with researchers from Utah State University, a GA-NN-CFD is being developed [10]. This system has been tested for the steady two jet system and initial results of this test would be presented in this section.

The neural network approach setup for this research is based on the Pyro [109] library implemented in Python. Currently, we are using one input layer with the optimization variables like suction strength and suction angle. Then a hidden layer connects this input layer to the output layer consisting of two nodes representing lift and drag. There exist no general rules on the structure of the network and we train several different networks with an error of 3 percent. Then we select the best network with the fewest nodes in the hidden layer for the NN-GA approach. A reasonable neural network will allow both for accelerated testing of GA techniques and for rapid fitness evaluation. However, in the final analysis, any interpolation approach cannot be used to set the final values, so final determination requires full CFD simulation.

An initial test of this GA-NN-CFD system has been completed and with a reasonably trained neural network, the following results are obtained for the two-jet steady flow control setup [10]. This system is still in the testing stage and the configurations obtained by the neural network have not been validated by actual CFD calculations.

Figure 7.2 presents a comparison of the fitness computed by the CFD calculation with the fitness obtained through the application of the neural network. This figure shows that currently the NN overpredict the maximum fitness.

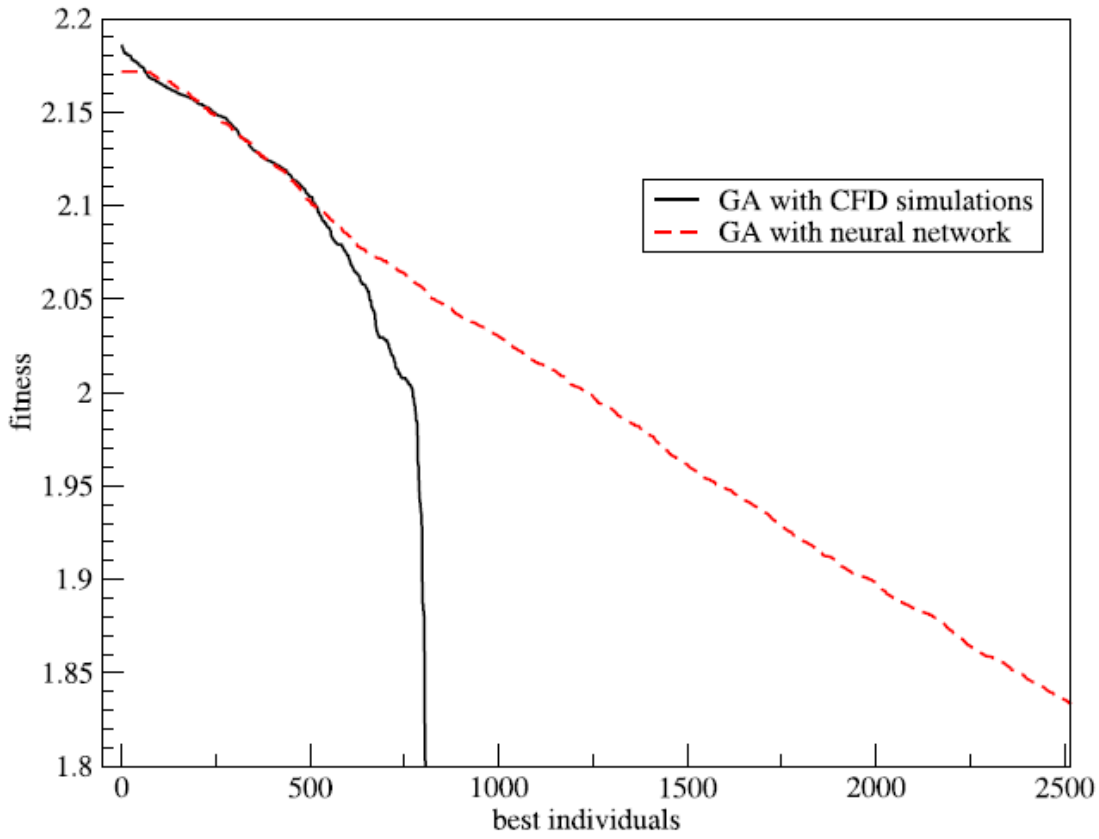
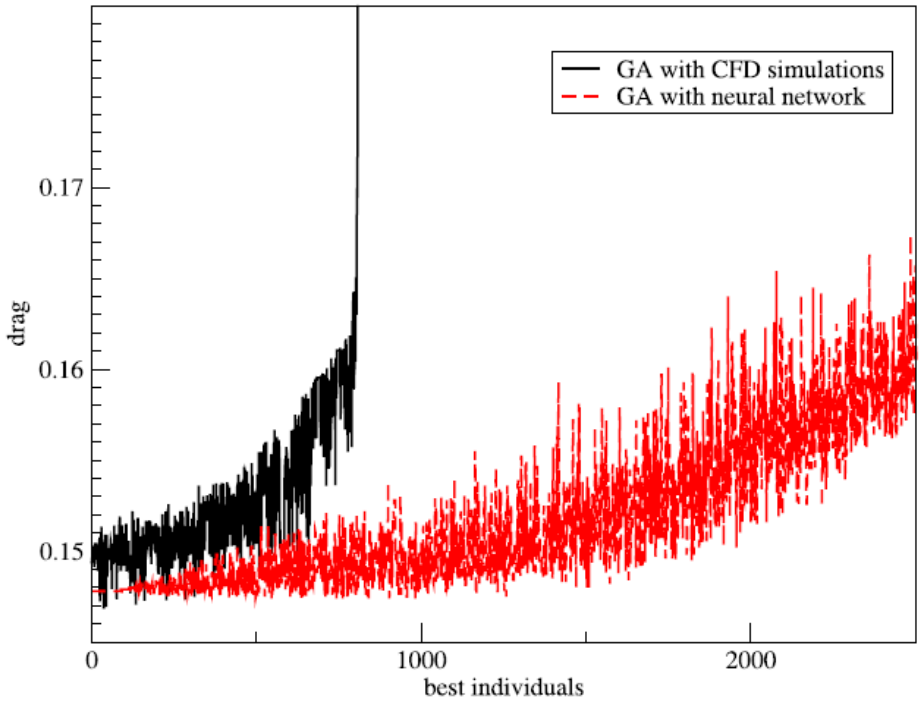
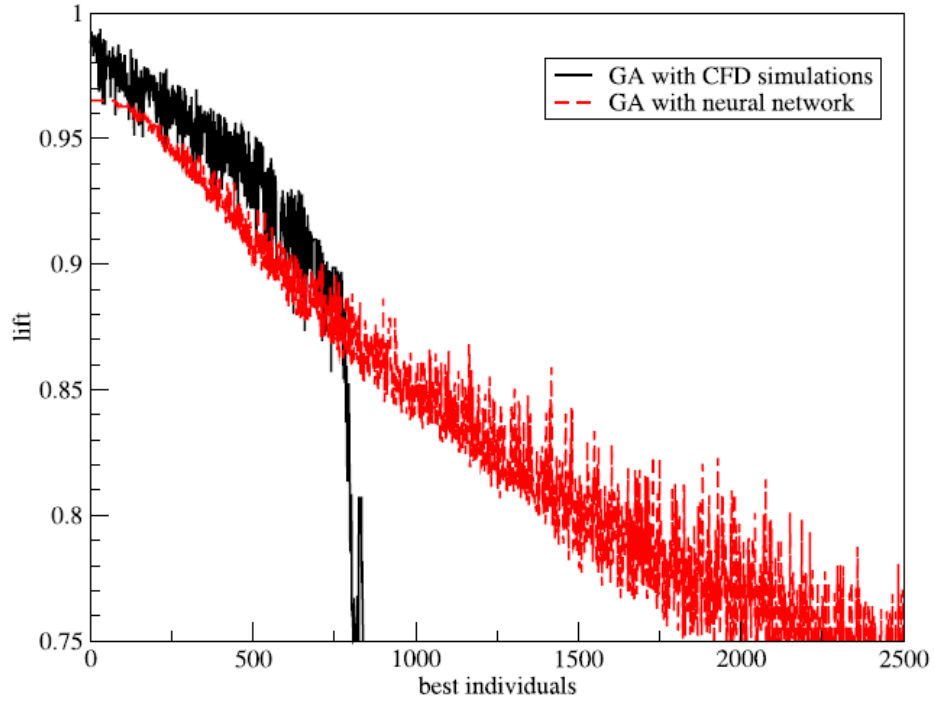


Figure 7.2: Fitness comparison GA-CFD and GA-NN system

In Figure 7.3 the drag and lift computed by the CFD simulations are compared by the predictions of the neural network for the best individuals of the GA optimizations. Here, as seen in Figure 7.3 (a) most of the error in the fitness comes from the lift prediction of the neural network. The drag prediction in Figure 7.3 (b) is also different, as expected, from the CFD prediction, but the best individuals in the CFD and NN approach have better agreement.



(a)



(b)

Figure 7.3: Lift and drag comparisons of GA-CFD and GA-NN system

The neural network does a reasonably good job in matching the output, i.e. the lift, amd drag, and the fitness (Figure 7.2 and 7.3). However, the configurations that generated these outputs are not similar to the configurations generated by the GA-CFD evaluation. The NN configurations were fundamentally different from the GA-CFD configuration, but were still yielding a similar fitness and lift/drag values. This suggests that the NN is only quantitatively matching the output data and thus requires more training and may be a better architecture which can better match the trend of the input parameters with the output data.

7.3 SUMMARY

The GA-NN system accelerates the optimization process from weeks to minutes but additional computational effort goes into generating the training data sets through CFD and training the NN. Since neural networks do not actually solve the Navier-Stokes equations and generate less accurate results, the proposed optimum performance regions determined by the GA-NN system need to be confirmed. The NN can also be used to test GA design with a realistic, but far less costly fitness evaluation, leading to improved GA design for both NN and CFD evaluations. Further work is necessary to develop guidelines for robust NN architectures, generation of training data sets and developing an integrated, automated GA-NN-CFD system.

CHAPTER – 8

8. CONCLUSIONS AND FUTURE WORK

8.1 CONCLUSIONS

A steady four-jet and a hybrid unsteady two-jet flow control system was successfully optimized using the Continuous Genetic Algorithm. The optimum configuration showed a great improvement in the fitness by increasing lift and decreasing drag, which was achieved by the weakening of the separation bubble. The improvements obtained are listed in Table 8.1 and 8.2. It was also shown that, for the steady four jet case optimization, the Continuous Genetic Algorithm performed better than the previously developed EARND Genetic Algorithm. The best configurations obtained for four-jet case using both the Genetic Algorithms and the best configuration for the unsteady case using the Continuous GA are presented in Table 8.3 and Table 8.4 respectively.

Table 8.1: Improvements in parameters for steady case

Steady Four-Jet Case			
GA Type	C_l	C_d	Fitness
Continuous GA	11.70% ↑	7.05% ↓	9.55% ↑
EARND GA	3.75% ↑	7.86% ↓	6.14% ↑
Combined Configuration	8.49% ↑	11.18% ↓	9.85% ↑

Table 8.2: Improvements in parameters for hybrid unsteady case

Hybrid Unsteady Two-Jet Case			
GA Type	C_l	C_d	Fitness
Continuous GA	15.24% ↑	6.52% ↓	10.88% ↑

Improved grids were developed for both the steady four-jet and the unsteady two-jet case. With the previous grid type, as the number of jets increased the complexity of arranging the grid

without overlapping also increased. The improved grid for this setup was more robust and was easy to handle. Also it is simple to increase or decrease the number of jets as it only requires changing the boundary conditions on the upper airfoil block. The stretch-grid developed for the unsteady case reduced the total grid points considerably as compared to the four-jet grid, hence making the grid setup computationally less expensive.

Table 8.3: Best configuration for the steady four-jet case

	Jet Parameters	Continuous GA	EARND GA
Leading Suction	Location	0.0500	0.0500
	Angle	-84.3700	-90.0000
	Amplitude	0.0212	0.0212
Trailing Suction	Location	0.0941	0.2200
	Angle	-82.2890	0.0000
	Amplitude	0.0206	0.0212
Leading Blowing	Location	0.5218	0.6187
	Angle	0.5970	45.0000
	Amplitude	0.0423	0.1000
Trailing Blowing	Location	0.7149	0.8000
	Angle	26.1900	90.0000
	Amplitude	0.0928	0.1414

Table 8.4: Best configuration for the hybrid unsteady two-jet case

	Jet Parameters	Continuous GA
Synthetic Jet-1	Location	0.0943
	Amplitude	0.1041
	Angle	82.6509
	Frequency	2.4014
	Amplitude Shift	-0.1899
Synthetic Jet-2	Location	0.1341
	Amplitude	0.0350
	Angle	96.0863
	Frequency	8.8501
	Amplitude Shift	-0.1981
	Phase Shift	2.4623

A new Genetic Algorithm viz. Continuous Genetic Algorithm was developed and integrated with the current GA-CFD setup. Necessary improvements were made to the previous

architecture to accommodate the Continuous GA. The Continuous GA was successfully applied to optimize the steady four-jet and the unsteady two-jet system.

An attempt was made towards setting up a GA-NN-CFD. The Neural Network performed a good job matching the output quantitatively, but the configurations generated by the NN were not very similar to the CFD configurations, suggesting the need for a better NN architecture and more training.

8.2 FUTURE WORK

As stated previously, the next step is to develop a more robust GA-NN-CFD system which would reduce the computations time from several days few hours. Using the data obtained by the current four-jet and unsteady cases along with the previous two-jet data, a well trained Neural Network could be developed. This setup can be then used to test various GA approaches as the evaluation of the fitness function would be much faster and computationally not expensive.

Another potential method that can replace most of the CFD computations is called kriging. The word "kriging" is synonymous with "optimal prediction"[112]. It is a method of interpolation which predicts unknown values from data observed at known locations. Kriging is also the method that is associated with the acronym B.L.U.E. (Best Linear Unbiased Estimator.)

- It is "linear" since the estimated values are weighted linear combinations of the available data.
- It is "unbiased" because the mean of error is 0.
- It is "best" since it aims at minimizing the variance of the errors.

The difference of kriging and other linear estimation method is its aim of minimizing the error variance. The kriging method estimates the output based on the linear combination of the input data while a neural network does a non-linear interpolation of the input data to generate the

output, therefore for the current flow control problem, kriging may better relate the trend of the input parameters with the output values rather than only matching it quantitatively.

A robust GA-CFD system in combination with an effective GA-NN-CFD or GA-Kriging-CFD system could be applied to optimize other, even more time consuming and complex CFD simulations such as the morphing wing problem or the moving wall problems. The GA-CFD system also needs to be validated with flow control setups which have experimental results. Apart from flow control, there are other numerous other fields where such an efficient system could be applied. Some of these are,

- In-land vehicle body design (Mechanical Engineering)
- Artificial organ (heart, lung, kidney) design (Biomedical Engineering)
- Spray painting (Mechanical and Chemical Engineering).

All these promising research areas require multi-disciplinary knowledge which interweaves the technology and advancement in computational fluid dynamics, genetic optimization algorithms, and nonlinear/linear interpolation schemes.

APPENDIX

A1. FLEXGRID.F90 MODIFICATION FOR STEADY CASE

```
! Single Airfoil Upper Block
       MidUpper1Start=0.05_high
        MidUpper1End  =0.85_high

        man=0
        do man=1, 4
           jete(man)    =  Jet_Pos(man) + 0.025_high
           Jetstart(man)= (Jet_Pos(man) - Midupper1start)/Jetstep+1
           Jetend(man)  = (Jete(man) - Midupper1start)/Jetstep+1
           PRINT *, Jetstart(man), Jetend(man)
        End do

      print*,"niMidUpper1",niMidUpper1
      ALLOCATE(MidUpper1sss(niMidUpper1),MidUpper1xa(niMidUpper1),MidUpper1ya
(niMidUpper1))
ALLOCATE(MidUpper1xxx(niMidUpper1,njmax),MidUpper1yyy(niMidUpper1,njmax))

      ra1=0.0
      ra2=0.0
        call
space(MidUpper1Start,MidUpper1End,JetStep,JetStep,niMidUpper1,MidUpper1sss,RA
1,RA2)

      do j=1,njmax
           do i=1,niMidUpper1
              call splint(s,x,x2,n,MidUpper1sss(i),MidUpper1xa(i),dxdn)
              call splint(s,y,y2,n,MidUpper1sss(i),MidUpper1ya(i),dydn)
              theta=atan(dydn/dxdn)
              IF(dxdn>0) then
                 MidUpper1xxx(i,j)=MidUpper1xa(i)-ra(j)*SIN(theta)
                 MidUpper1yyy(i,j)=MidUpper1ya(i)+ra(j)*COS(theta)
              else IF(dxdn<0)then
                 MidUpper1xxx(i,j)=MidUpper1xa(i)+ra(j)*SIN(theta)
                 MidUpper1yyy(i,j)=MidUpper1ya(i)-ra(j)*COS(theta)
              endif
           end do
        end do

  LeaUpperxxx(niLeaUpper,1:njmax)=Midupper1xxx(1,1:njmax)
  LeaUpperyyy(niLeaUpper,1:njmax)=Midupper1yyy(1,1:njmax)
```

A2. A TYPICAL 'INPUT' FILE

```
/* number_of_zone
  16                          ! Number of zones
 /* zone_number 1
  grid          110          84  4 2 !x and y points, number of BC's
  0.0 0.0
  Back4.dat       ! Below each zone name are the Boundary conditions
  m inlet               left    1       1       1       99999   0
  * patch               right   99999   99999   1       99999   2
  * patch               bottom  -99999  100000  1       1       9
  * patch               top     -99999   100000 99999   99999   6
 /* zone_number 2
 grid           61          84  8 2
  0.0 0.0
  Back1.dat
  * block               0       0       0       0       0       0      1
  * immerse             0       0       0       0       0       0      12
  * immerse             0       0       0       0       0       0      13
  * immerse             0       0       0       0       0       0      14
  * patch               left    1       1       1       99999   1
  * patch               right   99999   99999   1       99999   3
  * overlap               bottom  -99999  100000  1       1       10
  * overlap               top     -99999   100000 99999   99999   7
 /* zone_number 3
 grid           78          84  8 2
  0.0 0.0
  Back2.dat
  * block               0       0       0       0       0       0      1
  * immerse             0       0       0       0       0       0      12
  * immerse             0       0       0       0       0       0      14
  * immerse             0       0       0       0       0       0      15
  * patch               left    1       1       1       99999   2
  * patch               right   99999   99999   1       99999   4
  * overlap               bottom  -99999  100000  1       1       10
  * overlap               top     -99999   100000 99999   99999   7
 /* zone_number 4
 grid           63          84  8 2
  0.0 0.0
  Back3.dat
  * block               0       0       0       0       0       0      1
  * immerse             0       0       0       0       0       0      15
  * immerse             0       0       0       0       0       0      16
  * immerse             0       0       0       0       0       0      12
  * patch               left    1       1       1       99999   3
  * patch               right   99999   99999   1       99999   5
  * overlap               bottom  -99999  100000  1       1       10
  * overlap               top     -99999   100000 99999   99999   7
 /* zone_number 5
 grid          110          84  4 2
  0.0 0.0
  Back5.dat
  * patch               left    1       1       1       99999   4
  * outflow             right   99999   99999   1       99999   0
  * patch               bottom  -99999  100000  1       1       11
  * patch               top     -99999   100000 99999   99999   8
```

```
/* zone_number 6
 grid          110         100  4 2
  0.0 0.0
  BackUpper1.dat
  m inlet         left          1         1          -99999    100000   0
  * patch               right         99999       99999          -99999
100000   7
  * patch          bottom           1       100000      1            1
1
  * freestream    top          -99999       100000      99999    99999  0
 /* zone_number 7
 grid          140         100  6 2
  0.0 0.0
  BackUpper2.dat
  * overlap            bottom      -99999       100000      1     1     2
  * overlap            bottom      -99999       100000      1     1     3
  * overlap            bottom      -99999       100000      1     1     4
  * patch           left        1   1     -99999     100000       6
  * patch                 right       99999     99999 -99999 100000     8
  * freestream       top               -99999    100000  99999     99999
     0
 /* zone_number 8
 grid          110         100  4 2
  0.0 0.0
  BackUpper3.dat
  * patch           left    1       1      -99999  100000   7
  * outflow         right   99999   99999 1       99999     0
  * patch           bottom -99999   100000  1       1       5
  * freestream      top    -99999   100000  99999     99999       0
 /* zone_number 9
 grid          110         100  4 2
  0.0 0.0
  BackLower1.dat
  m inlet           left    1       1      -99999  100000   0
  * patch           right   99999   99999   1      100000   10
  * freestream     bottom  -99999  100000   1      1        0
  * patch            top    1      100000   99999  99999    1
 /* zone_number 10
 grid          140         100  6 2
  0.0 0.0
  BackLower2.dat
  * overlap          top   -99999  100000  99999    99999   2
  * overlap          top   -99999  100000  99999    99999   3
  * overlap          top   -99999  100000  99999    99999   4
  * patch       left        1       1      1       100000   9
  * patch       right       99999   99999  1       100000  11
  * freestream  bottom     -99999  100000  1        1       0

/* zone_number 11
 grid          110         100  4 2
  0.0 0.0
  BackLower3.dat
  * patch           left    1       1      1      100000  10
  * outflow        right    99999   99999  1       99999   0
  * freestream     bottom  -99999  100000  1       1       0
  * patch           top     1      100000  99999   99999  5
```

```
/* zone_number 12
 grid          454          50  6 2
  0.0 0.0
  rot_Lower.dat
  * patch               left    1       1       1       99999   16
  * patch               right   99999   99999   1       99999   13
  * wall                bottom  -99999  100000  1       1       0
  * overlap             top     -99999   100000 99999   99999   2
  * overlap             top     -99999   100000 99999   99999   3
  * overlap             top     -99999   100000 99999   99999   4
 /* zone_number 13
 grid           35          50  4 2
  0.0 0.0
  rot_LeaUpper.dat
  * patch               left    1       1       1       99999   12
  * patch               right   99999   99999   1       99999   14
  * wall                bottom  1       99999   1       1       0
  * overlap             top     -99999   100000 99999   99999   2
 /* zone_number 14
 grid          799          50  10 2
  0.0 0.0
  rot_MidUpper1.dat
  * patch               left    1       1       1       99999   13
  * patch               right   99999   99999   1       99999   15
  * wall                bottom  1  99999  1       1       0
  * overlap             top     -99999   100000 99999   99999   3
  * overlap             top     -99999   100000 99999   99999   2
  * overlap             top     -99999   100000 99999   99999   4
  p inlet               bottom           1          26 1       1     0
  q inlet               bottom         639         664 1       1     0
  r inlet               bottom          57          82 1       1     0
  s inlet               bottom         672         697 1       1     0
 /* zone_number 15
 grid           42          50  5 2
  0.0 0.0
  rot_TraUpper.dat
  * patch               left    1       1       1       99999   14
  * patch               right   99999   99999   1       99999   16
  * wall                bottom  -99999  100000  1       1       0
  * overlap             top     -99999   100000 99999   99999   3
  * overlap             top     -99999   100000 99999   99999   4
 /* zone_number 16
 grid           40          50  4 2
  0.0 0.0
  rot_Tra.dat
  * patch               left    1       1       1       99999   15
  * patch               right   99999   99999   1       99999   12
  * wall                bottom  -99999  100000  1       1       0
  * overlap             top     -99999   100000 99999   99999   4
```

A3. FLEXGRID.F90 MODIFICATION FOR UNSTEADY CASE

```
  MidUpper1Start=0.05_high
  MidUpper1End  =0.85_high

      if(Jet_Pos(1)>Jet_Pos(2))then
         tempjet=Jet_Pos(1)
         Jet_Pos(1)=Jet_Pos(2)
         Jet_Pos(2)=tempjet
      end if

      print*, "Jet Locations", Jet_Pos(1), Jet_Pos(2)

      if(Jet_Pos(1)<=0.065_high)then
         Jet_Pos(1)=0.065_high
      end if

      if(jet_Pos(2)>=0.785_high)then
         Jet_Pos(2)=.0785_high
      end if
      !Jet 1 Locations

      Jet1_gridS=Jet_Pos(1)-0.002_high-0.013_high
      Jet1_gridActS=Jet_Pos(1)-0.002_high
      Jet1_gridActE=Jet_Pos(1)+0.002_high
      Jet1_gridE=Jet1_gridActE+0.013_high

      !Jet 2 Locations

      Jet2_gridS=Jet_Pos(2)-0.002_high-0.013_high
      Jet2_gridActS=Jet_Pos(2)-0.002_high
      Jet2_gridActE=Jet_Pos(2)+0.002_high
      Jet2_gridE=Jet2_gridActE+0.013_high


      niMidUpperStoJ1=(Jet_Pos(1)-(0.002_high)- 0.013_high -0.05_high
)/UpperStep

      niMidUpperJ1=(Jet1_gridActS-Jet1_gridS)/Upperstep+10

      JetStart(1)=niMidUpperStoJ1+niMidUpperJ1

      niMidUpperJ1Act=(Jet1_gridActE-Jet1_gridActS)/JetStep

      Jetend(1)=Jetstart(1)+niMidUpperJ1Act

      niMidUpperJ1Last=(Jet1_gridE-Jet1_gridActE)/UpperStep+10

      niMidUpperBet=(Jet2_gridS-Jet1_gridE)/UpperStep

      niMidUpperJ2=(Jet2_gridActS-Jet2_gridS)/Upperstep+10

      Jetstart(2)=niMidUpperStoJ1+niMidUpperJ1+niMidUpperJ1Act+
niMidUpperJ1Last+niMidUpperBet+niMidUpperJ2
```

```
niMidUpperJ2Act=(Jet2_gridActE-Jet2_gridActS)/JetStep

      Jetend(2)=Jetstart(2)+ niMidUpperJ2Act

      niMidUpperJ2Last=(Jet2_gridE-Jet2_gridActE)/UpperStep+10

      niMidUpperJ2toE=(0.85_high-Jet2_gridE)/UpperStep


niMidUpper1=niMidUpperStoJ1+niMidUpperJ1+niMidUpperJ+niMidUpperBet+niMidUpper
J2toE+niMidUpperJ1Act+ &
      & niMidUpperJ1Last+niMidUpperJ2Act+niMidUpperJ2Last-8


ALLOCATE(MidBsss(niMidUpperStoJ1),MidBxa(niMidUpperStoJ1),MidBya(niMidUpperSt
oJ1))
      ALLOCATE(MidBxxx(niMidUpperStoJ1,njmax),MidByyy(niMidUpperStoJ1,njmax))

      ra1=0.0_high
      ra2=0.0_high

      passp=0

      call
space(MidUpper1Start,Jet1_gridS,JetStep,UpperStep,niMidUpperStoJ1,MidBsss,RA1
,RA2)

      do j=1,njmax
         do i=1, niMidUpperStoJ1
            call splint(s,x,x2,n,MidBsss(i),MidBxa(i),dxdn)
            call splint(s,y,y2,n,MidBsss(i),MidBya(i),dydn)
            theta=atan(dydn/dxdn)
            IF(dxdn>0) then
               MidBxxx(i,j)=MidBxa(i)-ra(j)*SIN(theta)
               MidByyy(i,j)=MidBya(i)+ra(j)*COS(theta)
            else IF(dxdn<0)then
               MidBxxx(i,j)=MidBxa(i)+ra(j)*SIN(theta)
               MidByyy(i,j)=MidBya(i)-ra(j)*COS(theta)
         end do
      end do


ALLOCATE(MidCsss(niMidUpperJ1),MidCxa(niMidUpperJ1),MidCya(niMidUpperJ1))
      ALLOCATE(MidCxxx(niMidUpperJ1,njmax),MidCyyy(niMidUpperJ1,njmax))


call space
(Jet1_gridS,Jet1_gridActS,UpperStep,JetStep,niMidUpperJ1,MidCsss,RA1,RA2)

      do j=1,njmax
         do i=1, niMidUpperJ1
            call splint(s,x,x2,n,MidCsss(i),MidCxa(i),dxdn)
            call splint(s,y,y2,n,MidCsss(i),MidCya(i),dydn)
            theta=atan(dydn/dxdn)
            IF(dxdn>0) then
               MidCxxx(i,j)=MidCxa(i)-ra(j)*SIN(theta)
               MidCyyy(i,j)=MidCya(i)+ra(j)*COS(theta)
```

```
            else IF(dxdn<0)then
               MidCxxx(i,j)=MidCxa(i)+ra(j)*SIN(theta)
               MidCyyy(i,j)=MidCya(i)-ra(j)*COS(theta)
            endif
         end do
      end do


ALLOCATE(MidDsss(niMidUpperJ1Act),MidDxa(niMidUpperJ1Act),MidDya(niMidUpperJ1
Act))
      ALLOCATE(MidDxxx(niMidUpperJ1Act,njmax),MidDyyy(niMidUpperJ1Act,njmax))

call space
(Jet1_gridActS,Jet1_gridActE,JetStep,JetStep,niMidUpperJ1Act,MidDsss,RA1,RA2)

      do j=1,njmax
         do i=1, niMidUpperJ1Act
            call splint(s,x,x2,n,MidDsss(i),MidDxa(i),dxdn)
            call splint(s,y,y2,n,MidDsss(i),MidDya(i),dydn)
            theta=atan(dydn/dxdn)
            IF(dxdn>0) then
               MidDxxx(i,j)=MidDxa(i)-ra(j)*SIN(theta)
               MidDyyy(i,j)=MidDya(i)+ra(j)*COS(theta)
            else IF(dxdn<0)then
               MidDxxx(i,j)=MidDxa(i)+ra(j)*SIN(theta)
               MidDyyy(i,j)=MidDya(i)-ra(j)*COS(theta)
            endif
         end do
      end do


ALLOCATE(MidEsss(niMidUpperJ1Last),MidExa(niMidUpperJ1Last),MidEya(niMidUpper
J1Last))

ALLOCATE(MidExxx(niMidUpperJ1Last,njmax),MidEyyy(niMidUpperJ1Last,njmax))


call space
(Jet1_gridActE,Jet1_gridE,JetStep,UpperStep,niMidUpperJ1Last,MidEsss,RA1,RA2)

      do j=1,njmax
         do i=1, niMidUpperJ1Last
            call splint(s,x,x2,n,MidEsss(i),MidExa(i),dxdn)
            call splint(s,y,y2,n,MidEsss(i),MidEya(i),dydn)
            theta=atan(dydn/dxdn)
            IF(dxdn>0) then
               MidExxx(i,j)=MidExa(i)-ra(j)*SIN(theta)
               MidEyyy(i,j)=MidEya(i)+ra(j)*COS(theta)
            else IF(dxdn<0)then
               MidExxx(i,j)=MidExa(i)+ra(j)*SIN(theta)
               MidEyyy(i,j)=MidEya(i)-ra(j)*COS(theta)
            endif
         end do
      end do


ALLOCATE(MidFsss(niMidUpperBet),MidFxa(niMidUpperBet),MidFya(niMidUpperBet))
```

```
      ALLOCATE(MidFxxx(niMidUpperBet,njmax),MidFyyy(niMidUpperBet,njmax))


call space
(Jet1_gridE,Jet2_gridS,UpperStep,UpperStep,niMidUpperBet,MidFsss,RA1,RA2)

      do j=1,njmax
         do i=1, niMidUpperBet
            call splint(s,x,x2,n,MidFsss(i),MidFxa(i),dxdn)
            call splint(s,y,y2,n,MidFsss(i),MidFya(i),dydn)
            theta=atan(dydn/dxdn)
            IF(dxdn>0) then
               MidFxxx(i,j)=MidFxa(i)-ra(j)*SIN(theta)
               MidFyyy(i,j)=MidFya(i)+ra(j)*COS(theta)
            else IF(dxdn<0)then
               MidFxxx(i,j)=MidFxa(i)+ra(j)*SIN(theta)
               MidFyyy(i,j)=MidFya(i)-ra(j)*COS(theta)
            endif
         end do
      end do


ALLOCATE(MidGsss(niMidUpperJ2),MidGxa(niMidUpperJ2),MidGya(niMidUpperJ2))
      ALLOCATE(MidGxxx(niMidUpperJ2,njmax),MidGyyy(niMidUpperJ2,njmax))


call space
(Jet2_gridS,Jet2_gridActS,UpperStep,JetStep,niMidUpperJ2,MidGsss,RA1,RA2)

      do j=1,njmax
         do i=1, niMidUpperJ2
            call splint(s,x,x2,n,MidGsss(i),MidGxa(i),dxdn)
            call splint(s,y,y2,n,MidGsss(i),MidGya(i),dydn)
            theta=atan(dydn/dxdn)
            IF(dxdn>0) then
               MidGxxx(i,j)=MidGxa(i)-ra(j)*SIN(theta)
               MidGyyy(i,j)=MidGya(i)+ra(j)*COS(theta)
            else IF(dxdn<0)then
               MidGxxx(i,j)=MidGxa(i)+ra(j)*SIN(theta)
               MidGyyy(i,j)=MidGya(i)-ra(j)*COS(theta)
            endif
         end do
      end do


ALLOCATE(MidHsss(niMidUpperJ2Act),MidHxa(niMidUpperJ2Act),MidHya(niMidUpperJ2
Act))

ALLOCATE(MidHxxx(niMidUpperJ2Act,njmax),MidHyyy(niMidUpperJ2Act,njmax))


call space
(Jet2_gridActS,Jet2_gridActE,JetStep,JetStep,niMidUpperJ2Act,MidHsss,RA1,
RA2)

      do j=1,njmax
         do i=1, niMidUpperJ2Act
```

```
            call splint(s,x,x2,n,MidHsss(i),MidHxa(i),dxdn)
            call splint(s,y,y2,n,MidHsss(i),MidHya(i),dydn)
            theta=atan(dydn/dxdn)
            IF(dxdn>0) then
               MidHxxx(i,j)=MidHxa(i)-ra(j)*SIN(theta)
               MidHyyy(i,j)=MidHya(i)+ra(j)*COS(theta)
            else IF(dxdn<0)then
               MidHxxx(i,j)=MidHxa(i)+ra(j)*SIN(theta)
               MidHyyy(i,j)=MidHya(i)-ra(j)*COS(theta)
            endif
            if (j==1) then
            end if
         end do
      end do


ALLOCATE(MidIsss(niMidUpperJ2Last),MidIxa(niMidUpperJ2Last),MidIya(niMidUpper
J2Last))

ALLOCATE(MidIxxx(niMidUpperJ2Last,njmax),MidIyyy(niMidUpperJ2Last,njmax))


call space
(Jet2_gridActE,Jet2_gridE,etStep,UpperStep,niMidUpperJ2Last,MidIsss,RA1,RA2)

      do j=1,njmax
         do i=1, niMidUpperJ2Last
            call splint(s,x,x2,n,MidIsss(i),MidIxa(i),dxdn)
            call splint(s,y,y2,n,MidIsss(i),MidIya(i),dydn)
            theta=atan(dydn/dxdn)
            IF(dxdn>0) then
               MidIxxx(i,j)=MidIxa(i)-ra(j)*SIN(theta)
               MidIyyy(i,j)=MidIya(i)+ra(j)*COS(theta)
            else IF(dxdn<0)then
               MidIxxx(i,j)=MidIxa(i)+ra(j)*SIN(theta)
               MidIyyy(i,j)=MidIya(i)-ra(j)*COS(theta)
            endif
            if (j==1) then
            end if
         end do
      end do


ALLOCATE(MidJsss(niMidUpperJ2toE),MidJxa(niMidUpperJ2toE),MidJya(niMidUpperJ2
toE))
      ALLOCATE(MidJxxx(niMidUpperJ2toE,njmax),MidJyyy(niMidUpperJ2toE,njmax))


call space
(Jet2_gridE,MidUpper1End,UpperStep,JetStep,niMidUpperJ2toE,MidJsss,RA1,RA2)

      do j=1,njmax
         do i=1, niMidUpperJ2toE
            call splint(s,x,x2,n,MidJsss(i),MidJxa(i),dxdn)
            call splint(s,y,y2,n,MidJsss(i),MidJya(i),dydn)
            theta=atan(dydn/dxdn)
            IF(dxdn>0) then
```

```
               MidJxxx(i,j)=MidJxa(i)-ra(j)*SIN(theta)
               MidJyyy(i,j)=MidJya(i)+ra(j)*COS(theta)
            else IF(dxdn<0)then
               MidJxxx(i,j)=MidJxa(i)+ra(j)*SIN(theta)
               MidJyyy(i,j)=MidJya(i)-ra(j)*COS(theta)
            endif
            if (j==1) then
            end if
         end do
      end do

      write(6,*)"MidUpper1",niMidUpper1


Allocate(MidUpper1xxx(niMidUpper1,njmax),MidUpper1yyy(niMidUpper1,njmax))
      midcount=1
      do i=1,niMidUpperStoJ1-1
         do j=1,njmax
            MidUpper1xxx(midcount,j)=MidBxxx(i,j)
            MidUpper1yyy(midcount,j)=MidByyy(i,j)
         end do
         midcount=midcount+1
      end do

      temp_int1=niMidUpperStoJ1+1
      temp_int2=niMidUpperStoJ1+niMidUpperJ1-1

      write(6,*) midcount, temp_int1, temp_int2

      do i=temp_int1, temp_int2
         do j=1,njmax
            MidUpper1xxx(midcount,j)=MidCxxx(i+1-temp_int1,j)
            MidUpper1yyy(midcount,j)=MidCyyy(i+1-temp_int1,j)
         end do
         midcount=midcount+1
      end do

      temp_int1=niMidUpperStoJ1+niMidUpperJ1+1
      temp_int2=niMidUpperStoJ1+niMidUpperJ1+niMidUpperJ1Act-1

      do i=temp_int1, temp_int2
         do j=1,njmax
            MidUpper1xxx(midcount,j)=MidDxxx(i+1-temp_int1,j)
            MidUpper1yyy(midcount,j)=MidDyyy(i+1-temp_int1,j)
         end do
         midcount=midcount+1
      end do


      temp_int1=niMidUpperStoJ1+niMidUpperJ1+niMidUpperJ1Act+1

temp_int2=niMidUpperStoJ1+niMidUpperJ1+niMidUpperJ1Act+niMidUpperJ1Last-1

      do i=temp_int1, temp_int2
         do j=1,njmax
            MidUpper1xxx(midcount,j)=MidExxx(i+1-temp_int1,j)
            MidUpper1yyy(midcount,j)=MidEyyy(i+1-temp_int1,j)
```

```
         end do
         midcount=midcount+1
      end do


temp_int1=niMidUpperStoJ1+niMidUpperJ1+niMidUpperJ1Act+niMidUpperJ1Last+1

temp_int2=niMidUpperStoJ1+niMidUpperJ1+niMidUpperJ1Act+niMidUpperJ1Last+niMid
UpperBet-1

      do i=temp_int1, temp_int2
         do j=1,njmax
            MidUpper1xxx(midcount,j)=MidFxxx(i+1-temp_int1,j)
            MidUpper1yyy(midcount,j)=MidFyyy(i+1-temp_int1,j)
         end do
         midcount=midcount+1
      end do


temp_int1=niMidUpperStoJ1+niMidUpperJ1+niMidUpperJ1Act+niMidUpperJ1Last+niMid
UpperBet+1

temp_int2=niMidUpperStoJ1+niMidUpperJ1+niMidUpperJ1Act+niMidUpperJ1Last+niMid
UpperBet+niMidUpperJ2-1

      do i=temp_int1, temp_int2
         do j=1,njmax
            MidUpper1xxx(midcount,j)=MidGxxx(i+1-temp_int1,j)
            MidUpper1yyy(midcount,j)=MidGyyy(i+1-temp_int1,j)
         end do
         midcount=midcount+1
      end do


temp_int1=niMidUpperStoJ1+niMidUpperJ1+niMidUpperJ1Act+niMidUpperJ1Last+niMid
UpperBet+niMidUpperJ2+1

temp_int2=niMidUpperStoJ1+niMidUpperJ1+niMidUpperJ1Act+niMidUpperJ1Last+niMid
UpperBet+niMidUpperJ2+niMidUpperJ2Act-1

      do i=temp_int1, temp_int2
         do j=1,njmax
            MidUpper1xxx(midcount,j)=MidHxxx(i+1-temp_int1,j)
            MidUpper1yyy(midcount,j)=MidHyyy(i+1-temp_int1,j)
         end do
         midcount=midcount+1
      end do


temp_int1=niMidUpperStoJ1+niMidUpperJ1+niMidUpperJ1Act+niMidUpperJ1Last+niMid
UpperBet+niMidUpperJ2+niMidUpperJ2Act+1

temp_int2=niMidUpperStoJ1+niMidUpperJ1+niMidUpperJ1Act+niMidUpperJ1Last+niMid
UpperBet+niMidUpperJ2+niMidUpperJ2Act+niMidUpperJ2Last-1

      do i=temp_int1, temp_int2
         do j=1,njmax
```

```
            MidUpper1xxx(midcount,j)=MidIxxx(i+1-temp_int1,j)
            MidUpper1yyy(midcount,j)=MidIyyy(i+1-temp_int1,j)
          end do
          midcount=midcount+1
       end do


temp_int1=niMidUpperStoJ1+niMidUpperJ1+niMidUpperJ1Act+niMidUpperJ1Last+niMid
UpperBet+niMidUpperJ2+niMidUpperJ2Act+niMidUpperJ2Last+1

temp_int2=niMidUpperStoJ1+niMidUpperJ1+niMidUpperJ1Act+niMidUpperJ1Last+niMid
UpperBet+niMidUpperJ2+niMidUpperJ2Act+niMidUpperJ2Last+niMidUpperJ2toE

        do i=temp_int1, temp_int2
          do j=1,njmax
            MidUpper1xxx(midcount,j)=MidJxxx(i+1-temp_int1,j)
            MidUpper1yyy(midcount,j)=MidJyyy(i+1-temp_int1,j)
          end do
          midcount=midcount+1
end do
```

REFERENCES

1. Lewis, Jeffrey C., Agarwal, and Ramesh K., "Airfoil design via control theory using full-potential and Euler equations", ASME Fluids Eng Div Publ FED, v 232, 1995, p 53-60.

2. Singh, K.P., Murali Krishna, K., Saha, S., and Mukharjea, S.K., "Application of an Euler Code on a modern combat aircraft configuration", ADA, Bangalore, India, Lecture Notes in Physics, n 453, 1995, p 535-539.

3. Huang, L., Huang, P.G, LeBeau, R.P., and Hauser, Th., "Optimization of Blowing and Suction Control on NACA 0012 Airfoil Using a Genetic Algorithm", AIAA-2004-0423, 2004.

4. Chaudhry, S.S., "Application of Genetic Algorithms in production and operations management: a review", International Journal of Production Research, v 43, n 19, 1 Oct. 2005, p 4083-101.

5. Gonzalez, L.F., E.J. Whitney, K. Srinivas, and J. Periaux, "Optimum Multidisciplinary and Multi-Objective Wing Design in CFD Using Evolutionary Techniques", International Conference on Computational Fluid Dynamics 3, Toronto, Canada, July 2004.

6. Kalyanmoy, D., and Tiwari S., "Multi-objective optimization of a leg mechanism using Genetic Algorithms', Engineering optimization, Vol. 37, No. 4, June 2005, p 325-350.

7. Chang, C.K., "Genetic Algorithms for project management", Annals of Software Engineering, v 11, 2001, p 107-39.

8. Hauser, Karina, "Simulation and Optimization of a crossdocking operation in a just-in-time environment", PhD Dissertation, University of Kentucky, 2002.

9. Douglas I. G., A Review of Unsteady Aerodynamic Modeling for Flight Dynamics of Maneuverable Aircraft, AIAA Atmospheric Flight Mechanics Conference and Exhibit, p16 - 19 August 2004, Providence, Rhode Island.

10. LeBeau, R.P., Beliganur, N, and Th. Hauser, "Flow Control Optimization Using Neural Networks and Genetic Algorithms", International Conference of Computational Fluid Dynamics 4, Ghent, Belgium, July 10-14, 2006.

11. Collis, S., Joslin, R.D, Seifert, A, and Theofilis, V, "Issues in active flow control: theory, control, simulation and experiment", Prog. In Aero. Sci., Vol. 40, No 4-5, pp 237-289, 2004.(earlier version as AIAA paper 2002-3277).

12. Katam, V., "Simulation of low-Re flow over a modified NACA 4415 airfoil with oscillating camber", Master's thesis, University of Kentucky, 2005.

13. Gad-el-Hak, Pollard, and Bonnet (Eds.), "*Flow Control: Fundamentals and Practices*", Springer-Verlag Berlin Heidelberg, 1998.

14. Gad-el-Hak, M., "Flow Control: The Future", Journal of Aircrafts, Vol. 38, No. 3, May–June 2001.

15. Gad-el-Hak, M., "*Flow Control: Passive, Active and Reactive Flow Management*", Cambridge University Press, 2000

16. Anderson, D. John, "*Computational Fluid Dynamics: The basics with applications*", McGraw-Hill International Editions, 1995.

17. Thomas Back, "*Evolutionary Algorithms in Theory and Practice*", Oxford University Press, Inc., New York, 1996.

18. J. H. Holland, "*Adaptation in Natural and Artificial Systems*", University of Michigan Press, Ann Arbor, 1975

19. Gen, M., and Cheng, R., "*Genetic Algorithms & Engineering Optimization*", John Wiley & Sons, Inc., New York, 2000.

20. Huang, Liang, "Optimization of Blowing and Suction Control on NACA0012 Airfoil using Genetic Algorithm with Diversity Control", PhD Thesis, University of Kentucky, 2004.

21. Haupt R. L and Haupt S. E., "*Practical Genetic Algorithms", Second Edition,* John Wiley & Sons, Inc., New York, 2004.

22. L. Prandtl, in *Verhandlungen des dritten internationalen Mathematiker-Kongresses in Heidelberg* 1904, A. Krazer, ed., Teubner, Leipzig, Germany (1905), p. 484. English trans. in Early Developments of Modern Aerodynamics, J. A. K. Ackroyd, B. P. Axcell, A.I. Ruben, eds., Butterworth-Heinemann, Oxford, UK (2001), p.77.

23. B. Thwaites, "Approximate Calculation of the Laminar Boundary Layer", Aeronautical Quarterly, Vol. 1, 1949, p 245-280.

24. B. S. Stratford, "The Prediction of Separation of the Turbulent Boundary Layer", Journal of Fluid Mechanics, Vol. 5, 1959, p 1-16.

25. Curle, N., and Skan. S., "Approximate Methods for Predicting Properties of Laminar Boundary Layers", Aeronautical Quarterly, Vol. 8, 1957, p 257-268.

26. L. Crabtree, "Prediction of Transition in the Boundary Layer of An Aerofoil", Journal of Royal Aeronautical Society, Vol. 62, 1958, p. 525-537.

27. E. C. Maskell, "Approximate Calculation of the Turbulent Boundary Layer In Two Dimensional Incompressible Flow", M. O. S. Report, 1958.

28. Eli Reshotko and Maurice Tucker, "Approximate calculation of the compressible turbulent boundary layer with heat transfer and arbitrary pressure gradient", NACA TN-4154, Lewis Flight Propulsion Laboratory, December 1957.

29. Julian Nitzberg Allen, and E. Gerald, "The effect of compressibility on the growth of the laminar boundary layer on low-drag wings and bodies", NACA TN-1255, July 1947.

30. John Stack, “Tests of airfoils designed to delay the compressibility burble”, NACA TN-976, Langley Memorial Aeronautical Laboratory, Langley Field, VA, December 1944.

31. McCullough, G. B., and Gault, D.E., “ Examples of Three Representative Types of Airfoil-Section Stall at Low Speed,” NACA TN-2502, Washington, DC, 1951.

32. Meuller, T. J., and Burns, T.F., “Experimental Studies of the Eppler 61 Airfoil at Low Reynolds Numbers,” 1982, AIAA Paper 82-0345.

33. Sunada, S., Sakaguchi, A., Kawachi, K., “Airfoil section characteristics at a low Reynolds number.”, Journal of fluids engineering, Transactions of the ASME, Vol. 119,1997, p 129-135.

34. L. Bahi, J.M. Ross and H.T. Nagamatsu, “Passive Shock Wave/Boundary Layer Control for Transonic Airfoil Drag Reduction”, 1983, AIAA Paper 1983-0137.

35. G. Savu and O. Trifu, “Porous Airfoils in Transonic Flow”, AIAA Journal, Vol. 22, 1984, p. 989-991.

36. H. D. Taylor, “Application of Vortex Generator Mixing Principles to Diffusers”, Research Department Concluding Report No. R-15064-5, United Aircraft Corporation, East Hartford, 1948.

37. H. H. Pearcey, “Shock Induced Separation and Its Prevention by Design and Boundary Layer Control”, Boundary layer and Flow Control, Vol. 2, Pergamon Press, Oxford, England, 1961, p 1166-1344.

38. J. D. Nickerson, “A Study of Vortex Generators at Low Reynolds Numbers”, 1986, AIAA Paper 1986-0155.

39. M.B. Bragg and G.M. Gregorek, “Experimental Study of Airfoil Performance with Vortex Generators”, Journal of Aircraft, Vol. 24, 1987, p 305-309.

40. Gu, W., Robinson, O., and Rockwell, D., "Control of vortices on a delta wing by leading-edge injection", AIAA Journal, v 31, 1993, p 1177-1186.

41. Saeed, F., and Selig, M. S., "Multipoint inverse airfoil design method for slot-suction airfoils", Journal of Aircraft, v 33, 1996, p 708-715.

42. Wright, M.C.M. and Nelson, P.A., "Wind tunnel experiments on the optimization of distributed suction for laminar flow control", Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering, v 215, n 6, p 343-354, 2001.

43. C. Wong and K. Konstantinos, "Flow control by spanwise blowing on a NACA 0012", 24th AIAA Applied Aerodynamics Conference, San Francisco, CA, p 2215-2228, 2006.

44. Greenblatt, D., and Wygnanski, I. J., "Control of flow separation by periodic excitation", Progress in Aerospace Sciences, v 36, 2000, p 487-545.

45. Bushnell, D.M., and McGinley, C.B. (1989) "Turbulence Control in Wall Flows," Ann. Rev. Fluid Mech. 21, p 1-20.

46. Fiedler, H.E., and Fernholz, H.-H. (1990) "On Management and Control of Turbulent Shear Flows," Prog. Aerospace Sci. 27, p 305-387.

47. Gad-el-Hak, M., and Bushnell, D.M. (1991) "Separation Control: Review," J. Fluids Eng. 113, p 5-30.

48. Moin, P. and Bewley, T. (1994) "Feedback Control of Turbulence," Appl. Mech. Rev. 47, p S3-S13.

49. Gad-el-Hak, M. (1994) "Interactive Control of Turbulent Boundary Layers: A Futuristic Overview," AIAA J. 32, p 1753-1765.

50. Glezer, A., Allen, M. G., Coe, D. J., Barton, S. L., Trautman, M. A., and Wiltse, J. W., "Synthetic Jet Actuator and Applications Thereof", U.S. Patent 5,758,823, June 2, 1998.

51. Barton L. Smith and A. Glezer, "The Formation and Evolution of Synthetic Jets", Physics of Fluids, Vol. 10, No. 9, Sep 1998, pp. 2281- 2297.

52. A. Glezer, "Shear Flow Control Using Fluidic Actuator Technology", Proceedings of the 1st Symposium on Smart Control of Turbulence, Tokyo, Japan, 1999.

53. D. C. McCormick, S. Lozyniak, D. G. MacMartin, and P. F. Lorber, "Compact High Power Boundary Layer Separation Control Actuation Development", ASME Fluids Engineering Division Summer Meeting, New Orleans, ASME FEDSM2001-18279, May 2001.

54. J. L Gilarranz and O.K. Rediniotis, "Compact, High-Power Synthetic Jet Actuators for Flow Separation Control", 39th AIAA Aerospace Sciences Meeting and Exhibit, AIAA Paper 2001-0737, 2001.

55. J. L. Gilarranz, X. Yue, and O. K. Rediniotis, "PIV Measurements and Modeling of Synthetic Jet Actuators for Flow Control," Proceedings of FEDSM'98, ASME Fluids Engineering Meeting, 1998.

56. Coe, D.L., Allen, M.G., Smith, B.L., and Glezer, A., "Addressable Micro-machined Jet Arrays," Transducers '95, Stockholm, Sweden, June 25-29, 1995.

57. Smith, B.L. and Glezer, A., 'Vectoring and Small-Scale Motions Effected in Free Shear Flows Using Synthetic Jet Actuators,' AIAA-97-0213, 35th Aerospace Sciences Meeting and Exhibit, Jan. 6-10, 1997, Reno, NV.

58. Amitay, M., Honohan, A, Trautman, M., and Glezer, A., 'Modification of the Aerodynamic Characteristics of Bluff Bodies Using Fluidic Actuators,' AIAA-97-2004, 28th AIAA Fluid Dynamics Conference, June 29 – Jul. 2, 1997, Snowmass Village, CO.

59. Jason Kiddy, Peter Chen, John Niemczuk, Don DeVoe, and Ken Kiger, "Active Flow Control using Micro-electromechanical Systems", Systems Planning and Analysis Inc., AIAA SDM Conference, 2000.

60. Munday, D. and Jacob, J.D., “Active Control of Separation on a Wing with Oscillating Camber.” AIAA Journal of Aircraft, 39, No. 1, 2002.

61. Kota, S., Hetrick, J., Osborn, R., Paul, D., Pendleton, Ed, Flick, P., and Tilmann, C., "Design and application of compliant mechanisms for morphing aircraft structures", Proceedings of SPIE - The International Society for Optical Engineering, v 5054, 2003, p 24-33 .

62. Martin, T., Guitton, A., Schmit, R, and Glauser, M. N., "Development of a morphing micro air vehicle wing using the combined POD and LSE technique", InfoTech at Aerospace: Advancing Contemporary Aerospace Technologies and Their Integration, AIAA 2005-7158, 2005.

63. Fischer, Michael C. and Vemuru, Chandra S., "Application of laminar flow control to the high speed civil transport. The NASA Supersonic Laminar Flow Control program", SAE Technical Paper Series, 1991, p 1-12.

64. L. D. Kral, J. F. Donovan, A. B. Cain, and A. W. Cary, "Numerical Simulation of Synthetic Jet Actuators," AIAA 28th Fluid Dynamics Conference, AIAA Paper 1997-1824, 1997

65. D. P. Rizzetta, M. R. Visbal, and M. J. Stanek, "Numerical Investigation of Synthetic Jet Flowfields," AIAA Journal, Vol. 37, No. 8, August 1999

66. Rumsey, C. L., Lee-Rausch, E. M., and Watson, R. D, "Three-dimensional effects in multi-element high lift computations", Computers and Fluids, v 32, 2003, p 631-657.

67. Jie-Zhi Wu, Xi-Yun Lu, Andrew G. Denny, Meng Fan and Jain-Ming Wu, “Post stall Flow Control On An Airfoil By Local Unsteady Forcing”, Journal of Fluid Mechanics, Vol. 371, 1998, p 21-58.

68. Catalin Nae, “Synthetic Jets Influence on NACA 0012 Airfoil at High, Angles of Attack”, AIAA Atmospheric Flight Mechanics Conference and Exhibit, Boston, Massachusetts, August 10-12, 1998.

69. A. Hassan, and R. D. Janakiram, "Effects of Zero-Mass Synthetic Jets on the Aerodynamics of the NACA 0012 Airfoil", Journal of the American Helicopter Society, Vol. 43, No. 4, Oct, 1998

70. Hassan, A.A., Martin, P.B., Tung, C., Cerchie, D., and Roth, J., "Active flow control measurements and CFD on a transport helicopter fuselage", AHS International, v 1, 61st Annual Forum Proceedings - AHS International, p 349-369, 2005.

71. R. Duvigneau, A. Hay, and M. Visonneau, "Study on the Optimal Location of a Synthetic Jet for Stall Control", AIAA-2006-3679 3rd AIAA Flow Control Conference, San Francisco, California, June 5-8, 2006.

72. Za’er Salem Abo-Hammour, “Advanced Continuous Genetic Algorithms and their Applications in the Motion Planning of Robot Manipulators and in the Numerical Solution of Boundary Value Problems”, PhD Thesis, Pakistan Institute of Engineering and Applied Sciences, Quaid-i-Azam University, 2002.

73. Rao S. S., "*Engineering Optimization – Theory and Practice*", Revised Third Edition, New Age International (P) Ltd., Publishers, 2005.

74. Reeves, C. R., "*Modern Heuristic Techniques for Combinatorial Problems*", Orient Longman, 1993.

75. Samii, Y. R. and Michielssen E., "*Electromagnetic Optimization by Genetic Algorithms*", Wiley-Interscience, 1999.

76. Goldberg D. E., "*Genetic Algorithms in search, optimization, and machine learning*", Addision-Wisley Company Inc: Reading, MA, 1989.

77. Wang L., and Zheng D. Z., "An effective hybrid optimization strategy for job-shop scheduling problems", Computers and Operations Research, 28 (6), p 585-596, 2001.

78. Kobayashi, R., and Nakanishi, I., "Application of Genetic Algorithms to focal mechanism determination", Geophysical Research Letter, Vol. 21 (8), 1994, p 729-732.

79. Wong, S. C., Wong, C. K, and Tong, C.O., "A parallelized Genetic Algorithm for the calibration of Lowry Model", Parallel Computing, Vol. 27 (12), 2001, p 1523-1536.

80. Hajela. P, Lin C. Y, "Real versus binary coding in Genetic Algorithms: A comparative study, in computational engineering using metaphors from nature, by Topping B. H. V, 2000, p. 77-83.

81. Duvigneau, R., and Visonneau, M., "Hybrid Genetic Algorithms and artificial neural networks for complex design optimization in CFD", Int. J. Numerical Methods in Fluids 2004, Vol. 44, p 1257–1278.

82. Jameson, A., Martinelli, L., and Pierce, N.A., "Optimum aerodynamic design using the Navier-Stokes equations", Theoretical and Computational Fluid Dynamics, Vol. 10, p 213-37, 1998.

83. Anderson, W. K., and Venkatakrishnan, V., "Aerodynamic design optimization on unstructured grids with a continuous adjoint formulation", Computers and Fluids, Vol. 28, 1999, p 443-480.

84. Anderson WK, and Nielsen E., "Aerodynamic design optimization on unstructured meshes using the Navier–Stokes", equations. AIAA Journal, Vol. 37, 1999, p 1411–1419.

85. Giunta A, Dudley J, Narducci R, Grossman B, Haftka R, Mason W, and Watson L., "Noisy aerodynamic response and smooth approximations in HSCT design", AIAA Paper 94-4316, 1994.

86. Hosder S, Grossman B, Haftka R, Mason W, and Watson L., "Observations on CFD simulation uncertainties", AIAA Paper 2002-5531, 2002.

87. Peigin, S., and Epstein, B., "Robust handling of non-linear constraints for GA optimization of aerodynamic shapes", International Journal for Numerical Methods in Fluids, Vol. 45, 2004, p 1339-1362.

88. Sengupta, T. K., Deb, K., and Talla, S. B., "Control of flow using Genetic Algorithm for a circular cylinder executing rotary oscillation", Computers and Fluids, Vol. 36, p 578-600.

89. Gutowski, M. W., “Smooth Genetic Algorithms”, Journal of Physics A-Mathematical and General, 27, 7893-7904, 1994.

90. Raymond P. LeBeau, Jr., Liang Huang, and Thomas Hauser, "Application of Evolutionary Algorithms to Small Jet Arrays for Flow Control", 17th AIAA Computational Fluid Dynamics Conference, 6 - 9 June 2005, AIAA 2005-4859, Toronto, Ontario Canada.

91. Palki, A., “Cache optimization and performance evaluation of a structured cfd code – GHOST”, MS Thesis, University of Kentucky, 2006.

92. Y. B. Suzen, P. G. Huang, "Numerical Simulation of Wake Passing on Turbine Cascades", 41st Aerospace Sciences Meeting and Exhibit, AIAA Paper 2003-1256, 2003.

93. Y. B. Suzen, P.G. Huang, R. J. Volino, T. C. Corke, F. O. Thomas, J. Huang, J. P. Lake and P. I. King, “A Comprehensive CFD Study of Transitional Flows In Low-Pressure Turbines Under a Wide Range of Operation Conditions”, 33rd AIAA Fluid Dynamic Conference, AIAA Paper 2003-3591, 2003.

94. Y. B. Suzen and P. G. Huang, “Predictions of Separated and Transitional Boundary Layers Under Low-Pressure Turbine Airfoil Conditions Using an Intermittency Transport Equation”, Journal of Turbomachinery, Vol. 125, No.3, 2003, pp. 455-464

95. Suzen, Y., Huang, G., Jacob, J. D., and Ashpis, D. “Numerical Simulations of Plasma Based Flow Control Applications.” AIAA 2005-4633, AIAA 35th Fluid Dynamics Conference and Exhibit, Toronto, CA, June 6-9, 2005.

96. Menter F. R., “Two-Equation Eddy-Viscosity Turbulence Models For Engineering Applications”, AIAA Journal, Vol. 32, No. 8, 1994, p 1598-1605.

97. Bardina, J. E., P. G. Huang and T. J. Coakley, “Turbulence Modeling Validation, Testing and Development”, NASA TM-110446, 1997.

98. Beliganur, N., R.P. LeBeau, and Th. Hauser, “Application of Genetic Algorithms and Neural Networks to Unsteady Flow Control Optimization”, 18th AIAA Computational Fluid Dynamics Conference, 25 - 28 Jun 2007, Miami, FL.

99. C. M. Rhie and W. L. Chow, “Numerical study of the turbulent flow past an airfoil with tailing edge separation”, AIAA Journal, Vol. 21, 1983, pp. 1523-1532.

100. Robert E. Dannenberg and James A. Weiberg, “Section Characteristics Of A 10.5-Percent Thick Airfoil With Area Suction As Affected By Chordwise Distribution Of Permeability”, NASA Technical Note 2847, Ames Aeronautical Laboratory, Moffett Field, CA, 1952.

101. M. Amitay, V. Kibens, D. Parekh, and A. Glezer, ‘‘The dynamics of flow reattachment over a thick airfoil controlled by a synthetic jet actuator, ’’ AIAA Paper. 99-1001, 1999.

102. A. Crook, A. M. Sadri, and N. J. Wood, ‘‘The development and implementation of synthetic jets for the control of separated flow,’’ AIAA Paper 99-3173, 1999.

103. C. Y. Lee and D. B. Goldstein, ‘‘DNS of microjets for turbulent boundary layer control,’’ AIAA Paper 2001-1013, 2001

104. B. L. Smith and A. Glezer, ‘‘Vectoring and small-scale motions affected in free shear flows using synthetic jet actuators,’’ AIAA Paper 97-0213, 1997.

105. Y. Chen, S. Liang, K. Aung, A. Glezer, and J. Jagoda, ‘‘Enhanced mixing in a simulated combustor using synthetic jet actuators,’’ AIAA Paper 99-0449, 1999.

106. A. Seifert, T. Bachar, D. Koss, M. Shepshelovich, and I. Wygnanski, ‘‘Oscillatory blowing: A tool to delay boundary layer-separation,’’ AIAA J. vol. 31, 2052, 1993.

107. R. Mittal and P. Rampunggoon, "On the virtual aeroshaping effect of synthetic jets", American Institute of Physics, Physics of Fluids, Vol. 14, Number 4, 2002.

108. HTTP://HOME.GWU.EDU/~RENI/RESEARCH.HTM

109. D.S. Blank, D. Kumar, L. Meeden, and H. Yanco. Pyro: A python-based versatile programming environment for teaching robotics. Journal of Educational Resources in Computing, 2004.

110. Smith BL, Glezer A., "Vectoring of a high aspect ratio air jet using zero-net-mass-flux control jet", Bull. Am. Phys. Soc. 39-1894, 1994.

111. Smith BL, Trautman MA, Glezer A., "Controlled interactions of adjacent synthetic jets. AIAA 37th Aerospace Science Meeting 99-0669, 1999.

112. A. G. Journel and CH. J. Huijbregts " *Mining Geostatistics*", Academic Press 1981

VITA

Narendra K Beliganur was born on May 10, 1981 in Donimalai, Karnataka, India. He received his bachelors' degree from Visweshwariah Technological University, Karnataka, India in July of 2002. Post bachelors' He worked as a *Software Engineer* at *GE* and *IBM* from 2002 to 2005. He joined the University of Kentucky in the Fall of 2005 to pursue his MS in Mechanical Engineering and successfully completed in the May of 2007. As a graduate student he served as a *Teaching Assistant* for ME321 (Engineering Thermodynamics II) and ME325 (Elements of Heat Transfer), and also as a *Research Assistant* at the UK CFD lab. Upon graduation he began work at *Corvid Technologies Inc.*, as a *Computational Analyst.*

Scholastic Honors

Kentucky Graduate Scholarship, Fall 2005 - Spring 2007
University of Kentucky, Lexington, KY.

Papers and Conferences

Beliganur, N., R.P. LeBeau, and Th. Hauser, "Application of Genetic Algorithms and Neural Networks to Unsteady Flow Control Optimization", Accepted, 18th AIAA Computational Fluid Dynamics Conference, 25 - 28 Jun 2007, Miami, FL

Beliganur, N., and R.P. LeBeau, "Application of Evolutionary Algorithms to Flow Control Optimization", Accepted, AIAA Region III Student Conference, 30 – 31 Mar 2007, University of Notre Dame, Notre Dame, IN – *Awarded Third-Place in the Master's best paper and presentation category*

Beliganur, N, R.P. LeBeau, and Th. Hauser, “Unsteady Flow Control Optimization using Evolutionary Algorithms”, Accepted, 32nd Annual Dayton-Cincinnati Aerospace Science Symposium, Mar 6th 2007, Dayton, OH

LeBeau, R.P., D.A. Reasor, X. Deng, S. Panguluri, Beliganur, N., and T. Hauser, "Performance Assessment of Fluid Dynamics codes on Different Computer Architectures" , 32nd Annual Dayton-Cincinnati Aerospace Science Symposium, Dayton, OH, March 6, 2007

Beliganur, N, LeBeau, R.P., D.G. Schauerhamer, and Th. Hauser, “Application of Genetic Algorithms to Complex Computational Fluid Dynamics Simulations”, 45th AIAA Aerospace Sciences Meeting and Exhibit, 8 - 11 Jan 2007, Reno, NV

LeBeau, R.P., Beliganur, N, Th. Hauser, “Flow Control Optimization Using Neural Networks and Genetic Algorithms”, International Conference of Computational Fluid Dynamics 4, July 10 - 14 Jul 2006, 2006, Ghent, Belgium

Beliganur, N., LeBeau, R.P., and Th. Hauser, “Combining Neural Networks and Genetic Algorithms for Flow Control Optimization”, 31st Annual Dayton-Cincinnati Aerospace Science Symposium, Mar 7th 2006, Dayton, OH