Control Power Optimization using Artificial Intelligence for Forward Swept Wing and Hybrid Wing Body Aircraft

Moustaine Adegbindin

Thesis submitted to the faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
In
Aerospace Engineering

Joseph A. Schetz
Rakesh K. Kapania
Mayuresh Patil

December 5th, 2016
Blacksburg, VA

Keywords: Optimization, Artificial Intelligence, Neural Network, Genetic Algorithm, Forward Swept Wing, Hybrid Wing Body, Hinge Moment

Control Power Optimization using Artificial Intelligence for Forward Swept Wing and Hybrid Wing Body Aircraft

Moustaine Adegbindin

ABSTRACT

Many futuristic aircraft such as the Hybrid Wing Body have numerous control surfaces that can result in large hinge moments, high actuation power demands, and large actuator forces/moments. Also, there is no unique relationship between control inputs and the aircraft response. Distinct sets of control surface deflections may result in the same aircraft response, but with large differences in actuation power. An Artificial Neural Network and a Genetic Algorithm were used here for the control allocation optimization problem of a Hybrid Wing Body to minimize the Sum of Absolute Values of Hinge Moments for a 2.5-G pull-up maneuver. To test the versatility of the same optimization process for different aircraft configurations, the present work also investigates its application on the Forward Swept Wing aircraft. A method to improve the robustness of the process is also presented. Constraints on the load factor and longitudinal pitch rate were added to the optimization to preserve the trim constraints on the control deflections. Another method was developed using stability derivatives. This new method provided better results, and the computational time was reduced by two orders of magnitude. A hybrid scheme combining both methods was also developed to provide a real-time estimate of the optimum control deflection schedules to trim the airplane and minimize the actuation power for changing flight conditions (Mach number, altitude and load factor) in a pull-up maneuver. Finally, the stability derivatives method and the hybrid scheme were applied for an antisymmetric, steady roll maneuver.

Control Power Optimization using Artificial Intelligence for Forward Swept Wing and Hybrid Wing Body Aircraft

Moustaine Adegbindin

GENERAL AUDIENCE ABSTRACT

Many futuristic aircraft such as the Hybrid Wing Body have numerous control surfaces that can result in large actuation power. An Artificial Neural Network and a Genetic Algorithm were used here to minimize the actuation power on the Hybrid Wing Body. To test the versatility of the same optimization process for different aircraft configurations, the present work also investigates its application on the Forward Swept Wing aircraft. A method to improve the robustness of the process is also presented. A completely different method was developed, and it provided better results with the computational time reduced by two orders of magnitude. A hybrid scheme combining both methods was also developed to provide a real-time estimate of the optimum control deflection schedules to trim the airplane and minimize the actuation power for changing flight conditions (Mach number, altitude and load factor) in a pull-up maneuver.

Acknowledgement

I owe a debt of gratitude to all those who have helped throughout my time as a graduate research assistant.

First and foremost, I would like to thank Dr. Joseph A. Schetz for seeing something in me and giving me the opportunity to work on such an important project. Without his supervision and guidance, this project would not have been possible. His insights, attention to detail and unending quest for excellence have in large part contributed to this work. Most importantly, his kindness and understanding make him easily approachable. I would like to extend my sincere gratitude to Dr. Rakesh K. Kapania for his co-supervision through this entire project. This research was made far more successful because of his intuitions, which motivated me to pursue critical thinking. He is always available whenever I need his assistance. He also has a real passion for teaching, which made me love his structure classes. I owe a big part of my success to him. I would also like to thank Dr. Mayuresh Patil for serving on my committee and for his support.

I must thank Nathan Love for always being available and willing to respond to all my questions, and for the numerous discussions, which have in a big part resulted in the successful completion of this work. It has been a pleasure working with Rupanshi Chhabra, who explained to me much of the research this work was developed from.

This work has been supported by NASA's Aeronautics Research and Mission Directorate (ARMD) through the ARMD Seedling Fund project 'Artificial Intelligence Based Control power Optimization on Tailless Aircraft'. Thanks to the project monitor and principal investigator, Frank H. Gern (NASA Langley Research Center). I would also like to thank Jesse Quinlan (NASA Langley Research Center), Martin Waszak (NASA Langley Research Center) and Nhan Nguyen (NASA Ames Research Center) for their helpful suggestions and supports in the progress of the present research.

Finally, I am deeply indebted to my parents for their huge support and unwavering love throughout the pursuit of this degree. I could not have done it without you.

Contents

ACKNOWLEDGEMENTS iii

LIST OF FIGURES vii

LIST OF TABLES x

NOMENCLATURE xii

1 Introduction 1

1.1 Background 1

1.2 Problem Definition 2

1.3 Proposed Solution 3

1.4 Literature Survey 4

1.5 Objective of this Thesis 4

2 Aircraft Configurations Studied 5

2.1 Hybrid Wing Body (HWB) 5

2.2 Forward Swept Wing (FSW) 7

3 Artificial Neural Networks 9

3.1 Introduction 9

3.2 Artificial Neural Network Structure 11

3.2.1 Learning 11

3.2.2 Perceptron 12

3.2.3 Multi-Layer Perceptron 13

3.2.4 Feedforward Network 14

3.2.5 Backpropagation 14

3.3 Artificial Neural Networks in MATLAB 16

3.3.1 Workflow for Neural Network Design 16
3.3.2 Architectures 17
3.3.3 Generalization Error 20
3.3.4 Training 21
3.3.5 Limitations 25
4 Genetic Algorithm 26
4.1 Introduction 26
4.2 Genetic Algorithm Operators 27
4.3 Genetic Algorithm Process 28
4.4 Genetic Algorithm vs. Classical Algorithm 30
4.5 Genetic Algorithm Toolbox in MATLAB 30
5 Static Aeroelasticity 32
5.1 Longitudinal and Lateral stability 33
5.1.1 Longitudinal Stability 34
5.1.2 Lateral and Directional Stability 35
5.2 Static Aeroelasticity in MSC Nastran 35
5.2.1 Introduction 35
5.2.2 Aerodynamic Analysis in MSC Nastran 36
5.3 Hinge Moments 40
6 Proposed Approach 43
6.1 Hypotheses 43
6.2 Initial Optimization Process 43
6.2.1 Dataset Generation using MSC Nastran 43
6.2.2 Data Segregation 45
6.2.3 Surrogate Model using an Artificial Neural Network 45

6.2.4 Optimization with Genetic Algorithm 46
6.2.5 Validation 46
6.3 Improvements 46
7 Results 48
7.1 Forward Swept Wing Aircraft Case 48
7.2 Hybrid Wing Body Case 59
7.3 FSW Case Optimization using Stability Derivatives 61
7.4 HWB Case Optimization using Stability Derivatives 65
7.5 HWB Case: Optimization with Stability derivatives vs. ANN+GA vs. SOL 200 66
7.6 Hinge Moment Optimization for HWB for a Pull-up Maneuver at Various Flight Conditions 68
7.7 Control Surface Deflection Schedule Optimization for HWB for a Steady Roll Maneuver using Stability Derivatives 72
7.8 Hinge Moment Optimization for Various Flight Conditions in a Roll Maneuver 77
8 Discussions 80
References 83

List of Figures

Figure 1.1: X-48B developed by Boeing Phantom team and NASA..... 2
Figure 1.2: The Boeing OREIO with up to 25 control surfaces and high lift devices. 3
Figure 2.1: Model Specifications of Boeing X-48B HWB OREIO aircraft 6
Figure 2.2: Control Surfaces in Boeing X-48B OREIO aircraft..... 7
Figure 2.3: FSW Half model with AILERON split into Inboard (INB) and Outboard (OUTB) sections 7
Figure 3.1: Human brain [13] 9
Figure 3.2: Structure of Neuron [14] 10
Figure 3.3: Artificial neuron [15]..... 10
Figure 3.4: Multi-layer Perceptron [19]..... 13
Figure 3.5: Order for correction of weights showing the 'bottom-top' oriented in the error back-propagation learning [16]..... 15
Figure 3.6: Example of a Neural Network Architecture..... 17
Figure 3.7: Linear Transfer Function..... 17
Figure 3.8: Log-Sigmoid Transfer Function..... 18
Figure 3.9: Neural Network Training Window in MATLAB [27]..... 22
Figure 3.10: Example of regression plot..... 23
Figure 4.1: Single-point crossover..... 27
Figure 4.2: Two-point crossover..... 27
Figure 4.3: Cut and Splice Crossover 28
Figure 4.4: Mutation 28
Figure 4.5: Plot of populations at iterations 60, 80, 95, and 100. 29
Figure 5.1: Aeroelasticity..... 32
Figure 5.2: Equilibrium flight 33
Figure 5.3: Statically unstable airplane..... 33
Figure 5.4: Neutral static stability..... 34
Figure 5.5: Longitudinal stability: Pitch equilibrium..... 34

Figure 5.6: Static directional stability 35
Figure 5.7: Floating tendency of trailing edge control surfaces 41
Figure 5.8: Restoring tendency of trailing edge control surfaces 41
Figure 6.1: Initial Optimization Schedule 44
Figure 6.2: New Optimization schedule with Data Set Updating and Retraining of Neural Network 47
Figure 7.1: Control surface deflections and corresponding Hinge Moments for the first case on FSW aircraft. 49
Figure 7.2: Scheme to determine the number of neurons in the Neural Network. 50
Figure 7.3: Root-mean-square error vs. Number of neurons. 51
Figure 7.4: Estimated Sum of Absolute Values of Hinge Moments vs. True Sum of Absolute Values of Hinge Moments from MSC Nastran. 51
Figure 7.5: Training ANN between the control deflections and all the components of Hinge Moments. 52
Figure 7.6: Initial Methodology. 53
Figure 7.7: Optimization using AELINK cards vs. optimization using control surface deflections as design variables. 54
Figure 7.8: Optimization results for FSW using the AELINK cards 55
Figure 7.9: Extended optimization process including Load factor and Pitch rate constraints. 57
Figure 7.10: Optimization results using Load factor and Pitch rate constraints for FSW case. 58
Figure 7.11: Optimization results using Load factor and Pitch rate constraints for the HWB Full model case. 60
Figure 7.12: Optimization approach using stability derivatives. 63
Figure 7.13: FSW case optimization results using stability derivatives. 63
Figure 7.14: HWB case optimization results using stability derivatives. 65
Figure 7.15: Hinge moment optimization for variable flight conditions in a Pull-up maneuver. 68
Figure 7.16: Flight envelope of long range mission of 777-200 LR 69

Figure 7.17: Training regression plot for 60 neurons for Sum of Absolute Values of Hinge Moments Training. 70
Figure 7.18: Training regression plot for 150 neurons for control surface deflections training. 70
F**igure 7.19**: HWB optimization in a steady roll maneuver. 76
Figure 7.20: Hinge moment optimization for variable flight conditions in a roll maneuver. 77
Figure 7.21: Training regression plot for 50 neurons for Sum of Absolute Values of Hinge Moments Training in a roll maneuver. 78
Figure 7.22: Training regression plot for 300 neurons for control surface deflections training in a roll maneuver 78

List of Tables

Table 4.1: Classical Algorithm vs. Genetic Algorithm 30
Table 4.2: GA options in MATLAB 30
Table 4.3: Mutlitobjective GA Option 31
Table 7.1: Optimization results using AELINK cards. 55
Table 7.2: AELINK cards within [-1.2 1.2] vs. AELINK cards within [-5 5]. 56
Table 7.3: Optimization results using Load factor and Pitch rate constraints for FSW case. 58
Table 7.4: Optimum control deflections for both half- and full-span models of the HWB. 59
Table 7.5: Optimization results using Load factor and Pitch rate constraints for the HWB Full model case. 60
Table 7.6: Constant values of control derivatives. 61
Table 7.7: Constant values of control derivatives. 62
Table 7.8: FSW case optimization results using stability derivatives. 64
Table 7.9: Computational time with optimization process using ANN+GA. 64
Table 7.10: Computational time with optimization process using stability derivatives. . 65
Table 7.11: HWB case optimization results using stability derivatives. 66
Table 7.12: HWB: Optimization with Stability derivatives vs. Neural Network vs. SOL 200 67
Table 7.13: Estimated results from the trained Neural Networks and validation with Genetic Algorithm for Mach = 0.65, Altitude = 18300 and Load factor = 1.66. 71
Table 7.14: Control surface deflections and corresponding hinge moments for the first set of AELINK cards. 73
Table 7.15: Hinge moments from MSC Nastran vs. Hinge moments from Eq. 30. 73
Table 7.16: Control surface deflections and corresponding hinge moments for the second set of AELINK cards. 74
Table 7.17: Hinge moments from MSC Nastran vs. Hinge moments from Eq. 30. 74
Table 7.18: HWB optimization in a steady roll maneuver. 76

Table 7.19: Estimated results from the trained Neural Networks and validation with Genetic Algorithm for Mach = 0.78, Altitude = 30200 ft. and Roll Rate = 20 deg/s. 79

Nomenclature

Symbols

C_{HM}	=	Hinge moment coefficient
HM	=	Hinge moment
C_L	=	Lift coefficient
C_{m_0}	=	Pitching moment
w_j	=	Downwash
A_{ij}	=	Aerodynamic influence coefficient matrix
f_j	=	Pressure on lifting element
q	=	Flight dynamic pressure
w_j^g	=	Static aerodynamic downwash
k	=	Reduced frequency
ω	=	Angular frequency
V	=	Free-stream velocity.
u_k	=	Displacement at aerodynamic grid points
D_{jk}^1 , D_{jk}^2	=	Real and imaginary parts of substantial differential matrix
S_{kj}	=	Integration matrix.
V	=	Velocity vector flow field
V_∞	=	Freestream velocity vector field
$q(x,y)$	=	Disturbance velocity due to the modeled surface.
$\varphi(x,y)$	=	Perturbation velocity potential
$\{u_k\}$	=	Structural grid point deflections
$\{u_g\}$	=	Aerodynamic grid point deflections
$[G_{kg}]$	=	Interpolation matrix
$\{F_k\}$	=	Aerodynamic forces
$\{F_g\}$	=	Structurally equivalent forces
$\{w_j\}$	=	Aerodynamic degrees of freedom

$\{u_x\}$	=	Extra aerodynamic points used to describe aerodynamic control surface deflections and overall rigid body motions
$[D_{jk}]$	=	substantial derivative matrix for the aerodynamic displacements
$[D_{jx}]$	=	substantial derivative matrix for the extra aerodynamic points.
Q_{aa}	=	A-set aerodynamic influence coefficient matrix
K_{aa}	=	Structural stiffness matrix
M_{aa}	=	Structural mass matrix
P_a	=	Vector of applied loads
C_y	=	Side force
C_l	=	Rolling moment
C_n	=	Yawing moment
β	=	Sideslip angle
$\frac{p\ b}{2\ V}$	=	Roll rate

Abbreviations

AELINK	:	Aeroelastic Linkage Coefficients
ANN	:	Artificial Neural Network
AoA	:	Angle of Attack
ARMD	:	Aeronautics Research and Mission Directorate
BDF	:	Bulk Data File
DLM	:	Doublet-Lattice Method
DMI	:	Direct Matrix Input
FSW	:	Forward Swept Wing
GA	:	Genetic Algorithm
HM	:	Hinge Moments

HWB	:	Hybrid Wing Body
LHS	:	Latin Hypercube Sampling
LM	:	Levenberg-Marquardt
MATLAB	:	Matrix Laboratory developed by MathWorks
MLP	:	Multi-Layer Perceptron
MSC	:	MacNeal - Schwendler Corporation
NASA	:	National Aeronautics and Space Administration
OREIO	:	Boeing Open Rotor Engine Integration on an HWB
SAHM	:	Sum of Absolute Value of Hinge Moments
SOL	:	Solution (MSC Nastran)
VLM	:	Vortex-Lattice Method

1 Introduction

1.1 Background

"Although airlines are safer and more profitable than any time in history, the industry must innovate much more rapidly in order to secure its environmental and financial viability in the future." [1]

— Samantha Shankman

Aviation is helping improve the development of the global economy, especially in many countries where citizens are traveling abroad for the first time. Besides the more than 58 million jobs and $2.4 trillion annual economic activity that aviation supports [1], it has several advantages. It brings people together, including families, friends and colleagues. It helps minds to meet and exchange ideas. In other words, it has transformed our gigantic planet into a tiny world of tremendous and magnificent opportunities. However, as it moves toward the future, aviation will face major challenges amongst which are safety, convenience, and environmental and financial challenges.

Flying today is much safer than the early days of aviation. In 2013 there were some 36.4 million flights and 16 fatal accidents [1]. But, accidents do happen, and the memory of those lost in those accidents must be honored by more efforts on safety. However, there can be nothing guaranteed about the future of aviation if the industry is not sustainable. The two most important dimensions of sustainability include environmental impact and profitability.

Sustainability is challenging for airlines that burn fuel to propel their aircraft. Nonetheless, the industry has committed to cutting, by 2050, the emissions to half the levels emitted in 2005. The easiest way to reduce emissions is to reduce fuel consumption. Airports throughout the world are encouraged to participate in an acceleration program that recognizes their efforts in managing and reducing their carbon emissions.

The other vector of sustainability is profitability. Over the last years, the world has witnessed great variations in oil prices with some sharp peaks. For both of these reasons, the main objective of research and development in aeronautics design will be the reduction of fuel consumption. This will require the use of new aircraft concepts, on the one hand, and new design methods, on the other hand, in order to exploit to the highest potential the synergies between propulsion systems, aerodynamics, aeroelasticity and control systems. Efficient control systems and control allocation can minimize both drag and control power requirements. That is the focus of this work.

1.2 Problem Definition

In an effort to explore new aircraft concepts, NASA, the Boeing Phantom team and the Air Force Research Laboratory have developed a Hybrid Wing Body aircraft concept: the X-48 (see Fig. 1.1)

Figure 1.1: X-48B developed by Boeing Phantom team and NASA

Hybrid Wing Body (HWB) aircraft, also known as Blended Wing Body, have a wide airfoil-shaped body, which helps to generate more lift as well as a higher lift-to-drag ratio than a conventional airplane. Because of these characteristics, they can be more environmentally friendly than the popular "tube and wings" design. The design reduces fuel consumption, offers potential to reduce emissions, has community noise benefits, and consequently counters the effects of the rapid increase in future air traffic volume. A comparison of HWB performance with that of a conventional aircraft for the same design point proved that the HWB fuel burn per seat mile is 27% lower that of the conventional aircraft [2]. HWB platforms also have large redundant control surfaces (see Fig. 1.2), which

may allow the aircraft to achieve a specific maneuver with many different control surface deflection schedules. However, the large control surfaces may sometimes lead to high hinge moments and required actuation power. Thus, an optimization problem must be solved to determine the combination of control surface deflections with the minimum actuation power to achieve a given maneuver. Because of the increased number of control surfaces and fundamentally different configuration as compared to a conventional design, traditional methods may not be effective for HWB control allocation optimization. [3, 4]

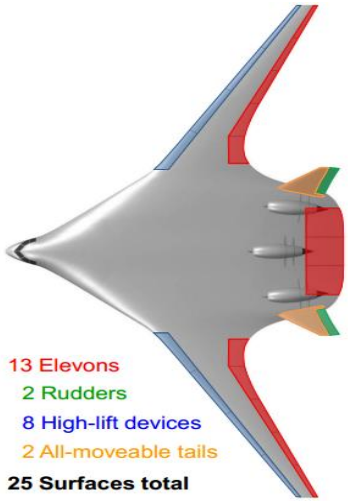

Figure 1.2: The Boeing OREIO with up to 25 control surfaces and high lift devices.

For transonic and supersonic speeds, there are no reliable theories to estimate hinge moments. The aerodynamic hinge moment depends on control deflections and aerodynamic quantities that are themselves dependent on the aircraft motion and the corresponding relationships are not always linear [5]. Therefore, an innovative method is required to solve the optimization problem.

1.3 Proposed Solution

As stated above, reduction of fuel consumption requires not only new aircraft designs, but also new design methods. The main objective of this work was to test the potential of using an Artificial Neural Network (ANN) and a Genetic Algorithm (GA) for the control allocation optimization of the HWB at the conceptual design level.

ANNs are generally used to estimate costly objective functions [6 - 8], and they are a way of creating a surrogate model for a response. ANNs are inspired by biological

nervous systems, and are used to estimate functions that are usually dependent on a large number of inputs. This makes them suitable for a control surface scheduling optimization of an aircraft, such as a HWB, with many control surfaces. ANNs have not been used to a great extent in the literature to overcome the weaknesses of conventional methods for control allocation problems.

1.4 Literature Survey

The available prior work on the use of ANNs and GA for control allocation has been performed by researchers at NASA and Virginia Polytechnic Institute and State University through a NASA Aeronautics Research Mission Directorate (ARMD) Seedling Fund Project. Gern *et al*. [3] designed a proof-of-concept process to demonstrate the potential of using neurocomputing to optimize actuation power for aircraft with multiple independently actuated control surfaces. This process was applied to a half-span model of an HWB aircraft for an initial 2.5-G pitch maneuver analysis, and the optimization results for the Sum of Absolute Values of Hinge Moments were improved by 12% over the best case in the initial sample set. The project employed a finite element-based aeroelastic HWB model inspired by Boeing OREIO aircraft (see Fig. 1.2) along with accompanying flight test and wind tunnel data to construct a database used to develop an ANN. Chhabra *et al*. [10] applied the same process on the full-span model of the same aircraft, and the results showed an improvement of 14% over the best case in the initial sample set. The proposed concept applies to the design and development stages of future aircraft.

1.5 Objective of this Thesis

In order to test the potential and robustness of the aforementioned process, the present research studies its application for additional maneuvers and another aircraft configuration, a Forward Swept Wing (FSW) aircraft.

2 Aircraft Configurations Studied

2.1 Hybrid Wing Body (HWB)

A Hybrid Wing Body aircraft is a configuration where the wing and fuselage are integrated, which essentially results in a large flying wing. HWB aircraft were previously called 'tailless airplanes' or 'Flying-Wing aircraft'. The shape of the aircraft is the first thing that attracts one's attention. The aircraft's appearance is like a single extended wing because of the HWB design. [11]

Some 85 years after the Wright Flyer, McDonnell Douglas aerodynamicist, Robert Liebeck proposed the concept of Blended Wing Body aircraft. But, the HWB aircraft concept has been on the drawing board even longer. Even though the HWB configuration has shown promise in terms of aerodynamic efficiency, and also shown quite remarkable advantages against the conventional "tube and wings" airframe, which included improved fuel efficiency, longer flight range, reliability and even lower manufacturing costs, such a concept has only been applied to military aircraft to obtain a low radar cross-section. The HWB's shape also reduces total wetted area. In this imaginative layout there is no need for a conventional tail.

In recent years, the Boeing Phantom team, NASA and the Air Force Research Laboratory (AFRL) have developed the X-48 as a HWB concept aircraft, and it has been observed very closely in its flight testing phase since 2006. In 2012, the remotely operated X-48C HWB aircraft, designed by Boeing and built by Cranfield Aerospace Limited in partnership with NASA, took its first flight from Rogers Dry Lake at Edwards Air Force Base, California. The X-48C model, was formerly the X-48B HWB aircraft shown here in Fig. 2.1, which was modified to evaluate the low-speed stability and control of a low-noise version of an HWB aircraft design. April 9, 2013 marked the end of the flight testing of X-48C. [11]

Hybrid Wing Body (HWB) platforms feature multiple control surfaces, with large control surface geometries leading to large hinge moments and high control power

demands. Due to the large number of control surfaces on a HWB aircraft, there is no unique relationship between control inputs and resulting aircraft response, i.e. different combinations of control surface deflections may result in the same maneuver, but with large differences in control power. [11]

The model specifications of THE HWB X-48B Half Span model are:

Span: 2552.1 in

Area: 576720 in^2

Chord: 818.35 in

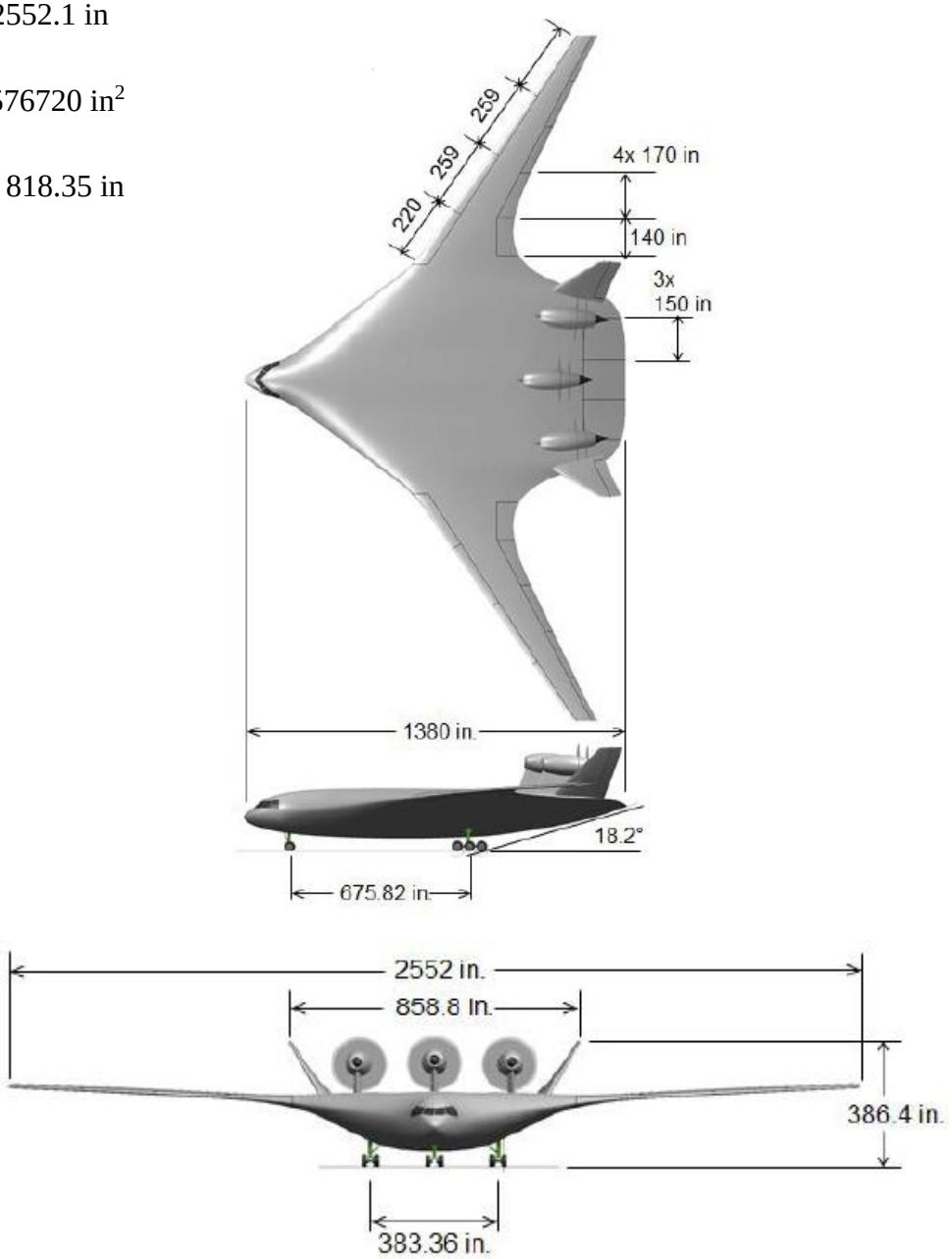

Figure 2.1: Model Specifications of Boeing X-48B HWB OREIO aircraft

Figure 2.2 below shows the control surfaces on a Boeing X-48B OREIO aircraft.

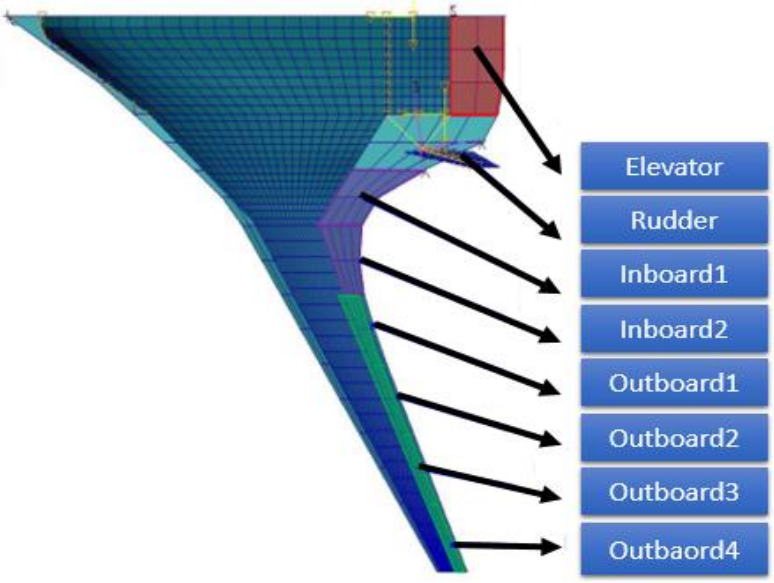

Figure 2.2: Control Surfaces in Boeing X-48B OREIO aircraft

2.2 Forward Swept Wing (FSW)

A Forward Swept Wing (FSW) aircraft was selected as a simpler case to study various analysis frameworks. It is an idealized model (see Fig. 2.3) from the MSC Nastran Aeroelastic Analysis User's Guide [12].

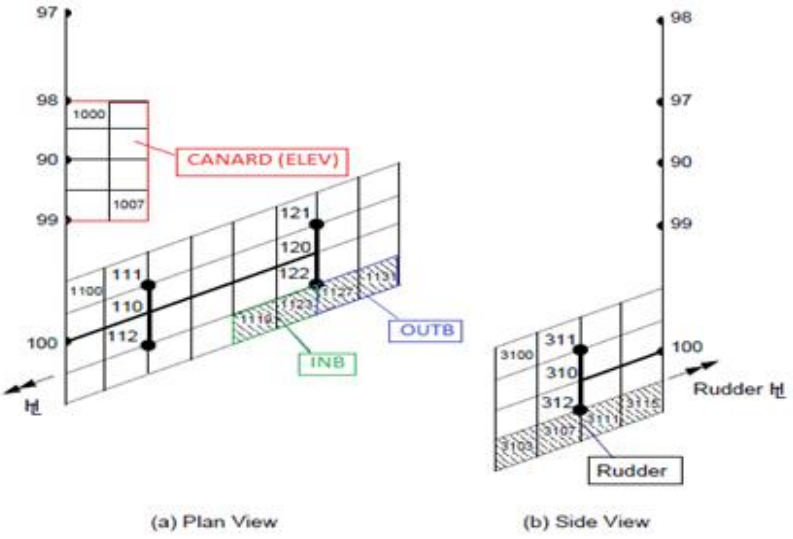

Figure 2.3: FSW Half model with AILERON split into Inboard (INB) and Outboard (OUTB) sections

Its wing has an aspect ratio of 4.0 and an incidence of 0.1 degrees with respect to the fuselage and a forward sweep angle of 30 degrees. Both the wing and the canard have no taper, twist, or camber. The reference chord is 10 ft., the reference area is 400 sq. ft., and the reference span is 40 ft. The FSW initially had one canard (elevator), one rudder and one aileron. To investigate the efficient deployment of multiple control surfaces, the aileron was split down the middle into two ailerons, the inboard (INB) and outboard (OUTB), resulting in a total of four control surfaces (see Fig. 2.3) for this research.

3 Artificial Neural Networks

3.1 Introduction

Recently, research in the field of neural networks has increased dramatically. Artificial neural networks constitute a part of Artificial Intelligence which focuses on making computers behave in a more intelligent way. They are an effort at emulating the information processing potentials of biological neural systems, and they are used in a wide range of applications including but not limited to robotics, pattern recognition, data processing, classification and function approximation (surrogate models). [13]

Figure 3.1: Human brain

An Artificial Neural Network's structure and functioning are motivated by the human brain. Human nervous systems are made up of billions of interconnected neurons (see Fig. 3.1), each of which is a complex organization capable of transmitting nerve impulses in several different ways. Although neurons are much slower than electronic components in reacting to a stimulus, the human brain is still able to deal with problems which no computer can effectively solve. The neuronal cells are the basic building blocks which constitute the architecture of the human nervous systems. A human brain is composed of different types of neurons, but they all share a similar structure (see Fig. 3.2).

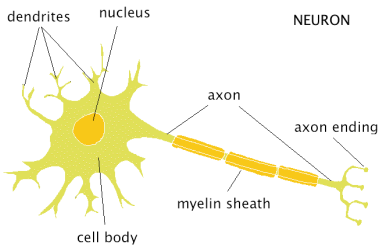

Figure 3.2: Structure of Neuron [14]

The *cell body* or *soma* is the main part of the cell. It is the bulbous part of a neuron, which contains the nucleus. The soma in turn contains the genetic information and connects to the *dendrites*, which are the primary elements that receive chemical signals from other neurons. The *axon* is a long fiber of a nerve cell which carries information from the soma to other neurons and muscles. The *myelin sheath* is the insulating substance that protects the axon and facilitates the transmission of impulses along a neuron. The *nerve ending* is the terminal structure of an axon, where the electro-chemical message from the axon is converted into chemical signal which is transmitted to the next neuron [14].

A natural neural network is composed of natural neurons which receive signals through *synapses* located on dendrites. If the signals received reach a *threshold*, the neuron is activated, and an *action potential* is initiated before traveling through the axon [14]. An artificial neural network functions in a similar but more abstract way. There are several types of networks, but they are all made of a set of nodes and connections between the nodes. Artificial neurons consist of *inputs* (similar to synapses), each transmitting a value x_i, which is multiplied by an associated *weight* w_i (see Fig. 3.3). The strength of the input is directly related to the value of the weight. Also, the signal is increased by positive weights and inhibited by negative ones. The transmitted information is then computed by a primitive function *f* which determines the activation of the neuron. The primitive function can be chosen arbitrarily. [15]

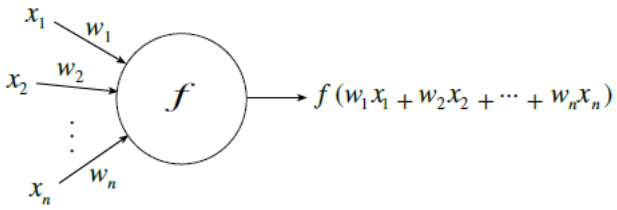

Figure 3.3: Artificial neuron

The computation of the neuron is dependent on the weight. So, a specific output can be targeted by adjusting the weight of a neuron. The more neurons there are, the more complicated it is to estimate the required weights to achieve a specific output. As a result, specific algorithms are developed to adjust the weights accordingly through a process known as *learning* or *training*. [15]

The model of an artificial neural network depends on three important elements:

- the structure of the nodes
- the topology of the network
- the training algorithm used to find the weights of the interconnections

3.2 Artificial Neural Network Structure

3.2.1 Learning

One of the first things to take into account when using ANNs is the type of the problem and the type of learning algorithm required to solve the problem. A number of learning algorithms are available, and they can be classified as unsupervised and supervised. Usually, problems requiring handling data are initially associated with unsupervised learning. Supervised algorithms are employed only after familiarization with the design space of inputs and outputs, and the characteristics of the responses. [16]

3.2.1.1 Unsupervised learning

In unsupervised problems, the data which one deals with have no specific or associated outputs. No external agent is involved in performing adjustments to the network weights, since in most cases the solution expected from the ANN is unknown. There are two main types of unsupervised learning: reinforcement learning and competitive learning. In reinforcement learning such as the Hebbian learning, each input will create a reinforcement of the network weight so as to improve the reproduction of the desired output [17]. In competitive learning, the nodes of the network compete with each other to provide the output related to an input vector. The unsupervised learning algorithms commonly used are the adaptive resonance theory and the self-organizing map.

3.2.1.2 Supervised learning

In supervised learning, known outcomes for specific inputs are given to the ANN during training. The network then compares the processed outputs against the targeted outputs. Adjustment of the weights is then performed based on the calculated error between the processed and targeted outputs [20]. The most common error used is the mean-squared error, which attempts to minimize the average squared error between the calculated and targeted outputs. Supervised learning is usually used for pattern recognition (classification) and regression (function approximation). Pattern recognition is the study of how machines can classify an unknown pattern into a category of pre-defined classes. Regression algorithms deal with constructing the relationship between continuous variables using the square error as the loss function.

3.2.2 Perceptron

Developed by Rosenblatt in 1958, the Perceptron is among the earliest ANNs and constitutes the simplest form of almost all ANNs. The basic model operates just as a biological model with input/output relations, and its main revolution was the introduction of numerical weights and unusual interconnections between the nodes. The summation formula to determine whether or not the threshold is met is:

$$Z = \sum_i w_i \, x_i \tag{1}$$

where x_i is the input and w_i the weight corresponding to each input.

The activation function is

$$y = f_N(z) \tag{2}$$

where z is the node output and f_N the activation function.

The activation function is a nonlinear function whose role is to keep the output within certain limits. Different functions are used as activation functions: Sigmoid, Threshold, Gaussian and Piecewise Linear, etc. [15]

The Perceptron is only a single neuron, but it can also be a single-network. Single-layer networks can be used only to solve problems with linearly separate patterns. Linearly separable problems are problems with input patterns that can be classified using a single hyperplane. The single-layer network has several limitations. For example, it cannot solve a 2-state Exclusive-Or (XOR) problem, or the 2-contradiction problem (XNOR) [18] and

other classes of problems that cannot be represented with any linear separation. To overcome the limitations of single-layer networks, multi-layer Perceptrons were introduced.

3.2.3 Multi-Layer Perceptron

The multi-layer Perceptron (see Fig. 3.4 below) is an ANN that consists of an input layer, one or more hidden layers for computation nodes, and an output layer of nodes [19]. One of the most important differences between single-layer and multi-layer Perceptrons is that the first ones are made up of parallel Perceptrons, making them limited in the kind of mappings they can represent. The multi-layer Perceptron is used either for unsupervised learning with *auto-associative* structure or for supervised learning with the *backpropagation* algorithm, which will be discussed in the next section.

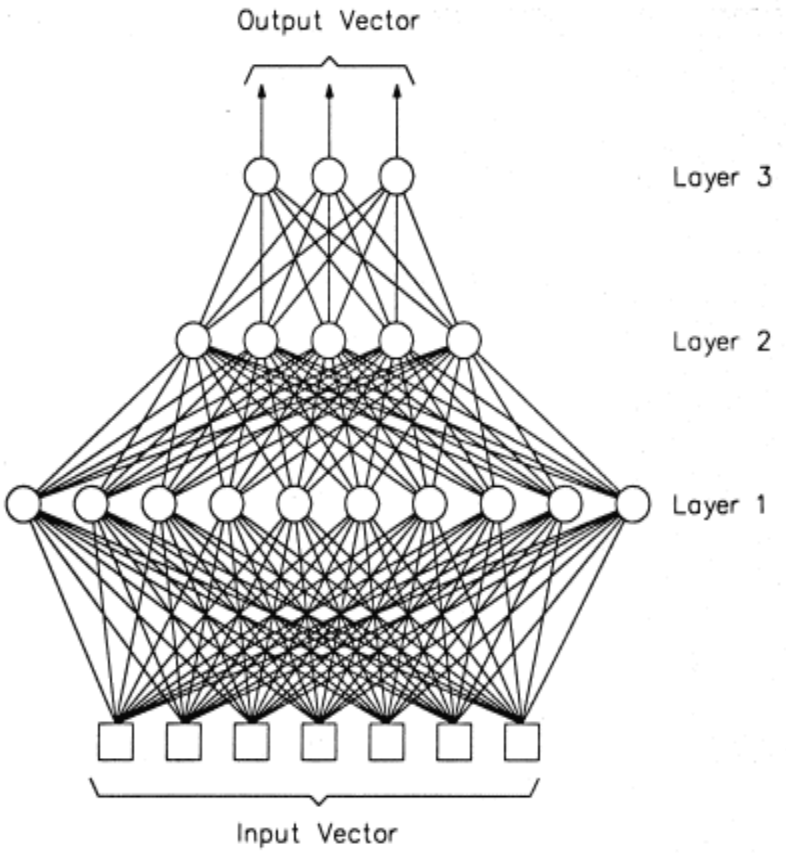

Figure 3.4: Multi-layer Perceptron

3.2.4 Feedforward Network

A feedforward neural network is a network consisting of an input layer, one or more hidden layers, and an output layer. In this network, the input of one layer is the output of the previous layer. As a result, no cyclic connections exist between the nodes and information always flows in the forward direction. There is no connection between the nodes of hidden layers and the external environment and no connection between the nodes of the same layer. One advantage of the feedforward neural network lies in its capability to recognize input patterns [13]. Also, any mapping problem can be fitted by a feedforward ANN with only one hidden layer and sufficient neurons in the hidden layer. The Feedforward Network was used in this research.

3.2.5 Backpropagation

Introduced by Rumelhart, Williams and Hinton in 1986 [20], the back propagation algorithm has been used for training multi-layer networks by weights adjustment. It is mainly used within feedforward networks, meaning that the signal travels forward while the error moves backward. Because the backward propagation employs supervised learning, both the inputs and corresponding outputs must be provided to the algorithm. The error, which is the difference between the calculated and targeted output is then calculated, and the goal of the backpropagation algorithm is to adjust the weights until the error is minimized below a certain threshold. [20]

The error function of the neural network can be defined as:

$$E(x, w, d) = \sum_j (O_j(x, w) - d_j)^2 \tag{3}$$

$$O_j(x, w) = \frac{1}{1+e^{A(x,w)}} \tag{4}$$

$$A_j(x, w) = \sum_{i=0}^{n} x_i w_{ji} \tag{5}$$

A_j is the activation function of neural networks using backpropagation which depends upon the inputs and weights. O_j is the sigmoidal output function, which only depends on *A*. The total error function E then depends only on the inputs and weights. By computing how the

error varies with respect to the weights ($\frac{\partial E}{\partial O_j}$), the algorithm can find the appropriate weights to minimize the error and have the computed output as close as possible to the targeted output [20]. Because the outputs of the intermediate layers are not available, the algorithm starts by computing the output layer, which is the only one where desired outputs are available. The weights of neurons are first corrected in the output layer, then in the last hidden layer (if available), and at the end in the first hidden layer that receives the signals directly from the input. This order of weight correction allows to know the exact error on each output node. Figure 3.5 below displays the order of weight correction in backpropagation.

In backpropagation, the time required for training varies exponentially with respect to the number of layers. It is then preferable to use this type of algorithm for networks with few layers. [16]

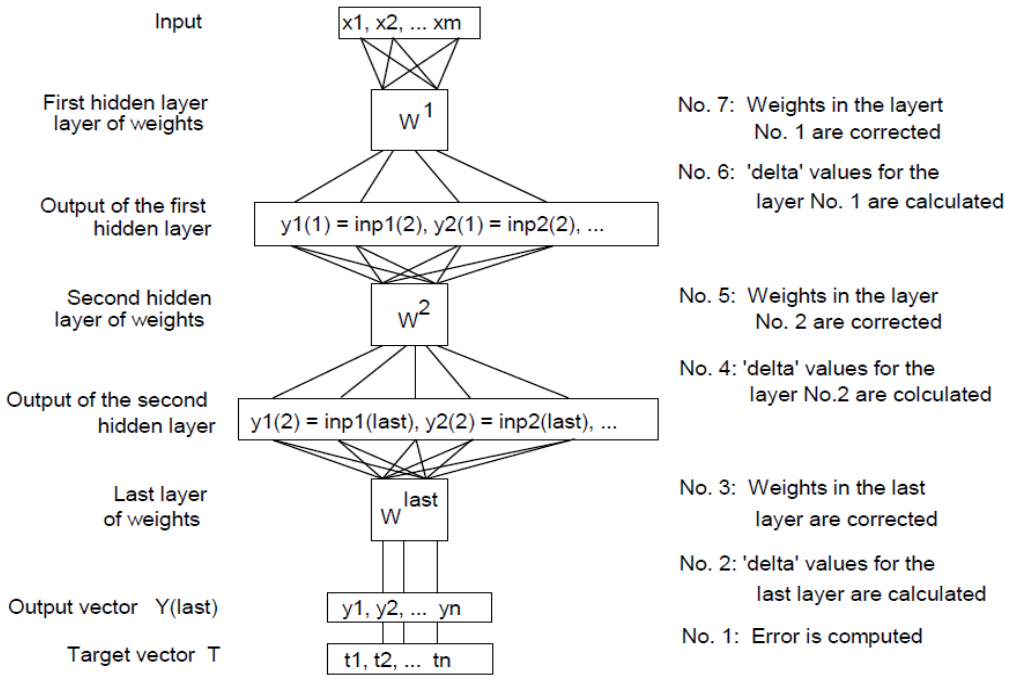

Figure 3.5: Order for correction of weights showing the 'bottom-top' oriented in the error backpropagation learning

For this work, both inputs and outputs were available for Neural Network training, so it was decided to use the Backpropagation algorithm.

3.3 Artificial Neural Networks in MATLAB

The Neural Network Toolbox in MATLAB [21] includes different functions and algorithms to create, train, visualize and simulate neural networks. It is also possible to perform regression, classification, clustering, time-series forecasting, and dynamic system modeling and control. The toolbox also provides tools for accelerated training of large data sets by dividing computations and data across multicore processors, GPUs, and computer clusters using the Parallel Computing Toolbox. The Neural Network Toolbox's key features include [21]:

- Deep learning, including convolutional neural networks and autoencoders
- Parallel computing and GPU support for accelerating training
- Supervised learning algorithms, including multilayer, radial basis, learning vector quantization (LVQ), time delay, nonlinear autoregressive (NARX), and recurrent neural network (RNN)
- Unsupervised learning algorithms
- Apps for data-fitting, pattern recognition, and clustering.

3.3.1 Workflow for Neural Network Design

In MATLAB, the process of a neural network design has seven steps [22]:

1- Collect data
2- Create the network
3- Configure the network
4- Initialize the weights and biases
5- Train the network
6- Validate the network
7- Use the network

Important parts of the network creation and configuration include choosing the transfer functions, defining the network architecture, the number of neurons and hidden layers.

3.3.2 Architectures

A wide variety of supervised and unsupervised architectures is supported by Neural Network Toolbox. Using the toolbox's modular approach to building networks, one can customize network architecture according to a problem. An example of a network architecture with the inputs, outputs, and interconnections is visible on Fig. 3.6 below [23].

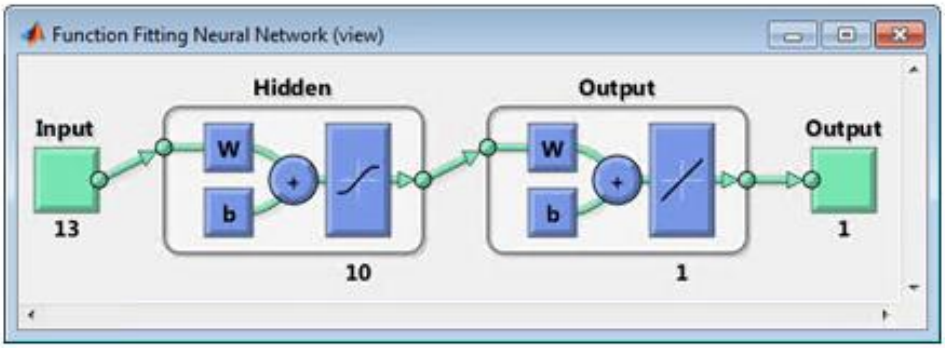

Figure 3.6: Example of a Neural Network Architecture

Figure 3.6 displays a two-layer feedforward network with sigmoid hidden neurons and linear output neurons.

3.3.2.1 Transfer Functions

Many transfer functions are included in the Toolbox, but the two most commonly used include the linear transfer function (see Fig. 3.7) and the Log-Sigmoid Transfer function (see Fig. 3.8). [24]

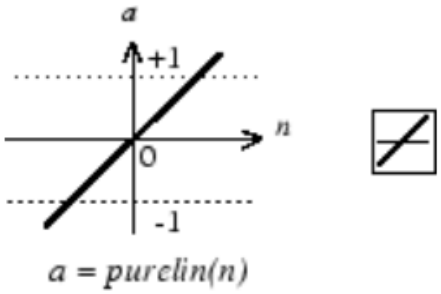

Figure 3.7: Linear Transfer Function

The linear transfer function is used for neurons in the final layer of multilayer networks that are used as function approximators.

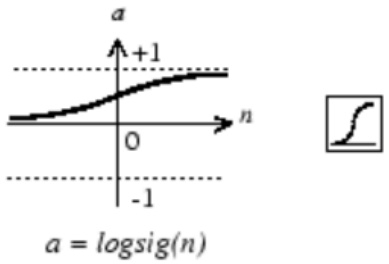

Figure 3.8: Log-Sigmoid Transfer Function

The Log-Sigmoid Transfer is usually used in the hidden layers of multilayer networks. Other functions can also be used as transfer functions.

3.3.2.2 Number of Hidden Units

The choice of the number of hidden units requires many considerations including [25]:

- The number of input and outputs units
- The number of training cases
- The amount of noise in the targets
- The complexity of the function or classification to be learned
- The architecture
- The type of hidden unit activation function
- The training algorithm
- Regularization

The only way to determine the best number of units is to train several networks and estimate the generalization error (see Sect. 3.3.3) of each. With too few hidden units, there will be underfitting and high statistical bias, which will result in high training error and high generalization error. With too many hidden units, there may be low training error, but there may still have overfitting and high variance.

Some rules of thumb for choosing the number of units include:

- The size of the hidden layer is somewhere between the sizes of the input layer and the output layer.
- The number of hidden nodes can be calculated using the following formula:

$$Number\ of\ hidden\ nodes = \frac{2}{3} * (Number\ of\ inputs + outputs) \quad (6)$$

- The hidden layer should never be more than twice as large as the input layer.

- A good choice of the number of hidden units depends on whether *early stopping* or other forms of regularization (see Sect. 3.3.4.3) are used.
- With conventional optimization algorithms, the weights used should not be more than the training cases. Otherwise, *overfitting* will result.
- With the standard backpropagation algorithm, it is difficult to reduce training error to a level near the globally optimal value, even when using more weights than training cases.

These rules of thumb do not guarantee the best training, because they do not take into account the number of training cases, the amount of noise in the targets, and the complexity of the function.

3.3.2.3 Number of Hidden Layers

Some problems do not require hidden layers. Certain rules can be used to choose the number of hidden layers:

- In Multilayer Perceptron with step or threshold of Heaviside activation functions, two layers are needed for full generality.
- In Multilayer Perceptron with any of a wide variety of continuous nonlinear hidden-layer activation functions, one hidden layer with an arbitrarily large number of units can be used. [25]
- With only one input, there is no advantage in using more than one hidden layer.
- In certain architectures, such as Cascade Correlation, more than two hidden layers can be used. Cascade Correlation Learning Architecture is a supervised learning algorithm which, instead of adjusting the weights in a network of fixed topology, begins with a minimal network, then automatically trains and adds new hidden units one by one, creating a multi-layer structure. [48]
- Two hidden layers may worsen the problem of local minima and it is important to use random initializations for global optimization.
- Radial Basis Function networks, which are traditionally associated with radial functions in a single-layer network [49], usually require a single hidden layer. But an extra, linear hidden layer before the radial hidden layer can enable the network to ignore irrelevant inputs.

3.3.3 Generalization Error

The goal of using a neural network is to generalize, i.e. to have the outputs of the net approximate the target values for inputs that are not necessarily in the training set. As stated earlier, the only way to determine the best number of units and hidden layers is to train several neural networks and choose the one with the minimum generalization error. There are different ways to estimate the generalization error:

- Cross-validation: The goal is to use all of the data for training by refining the split-sample validation. But, the network has to be trained several times.
- Bootstrapping: It usually works better than cross-validation. It involves repeatedly analyzing subsamples of data instead of subsets of data, each subsample being a random sample with replacement from the full sample. Bootstrapping requires more computational time than cross-validation.
- Split-sample of hold-out validation: This is the most frequently used method to estimate generalization error. It works by putting aside a chunk of available data as a test set, which must not be used for training. The error on the test set after training the network gives a good estimate of the generalization error, presuming that the test set is arbitrarily chosen and is a good illustrative sample of the data to generalize. A drawback of this method is that less than the available data will be used to train the network.
- Simple sample statistics: Statistical theory gives many estimators of the generalization error under different sampling assumptions in linear models. The estimators tune the training error for the number of weights being estimated. The same statistics can also be used as simple estimates of the generalization error in nonlinear cases when there is a large training set.

It is important to note that generalization is not always possible and it requires prior knowledge of the inputs, which must carry enough information related to the target. This is because the network cannot learn a nonexistent function. Also, the function must be to some extent smooth, meaning that a small change in the input must cause only a small change in the output. Finally a good generalization requires the training sample to be large enough to represent the set of all cases to be generalized. This is to avoid the need of

extrapolation, which is a type of generalization that is at once both unreliable and usually results into errors. [25, 26]

3.3.4 Training

The training starts with initialization of the network weights and biases. The network inputs and target outputs are required to start the training. The process requires adjusting the values of the weights and biases to optimize a network performance, represented by a network performance function *net.performFcn.* For feedforward networks, *mean square error* is the default performance function and it is defined as:

$$F = mse = \frac{1}{N}\sum_{i=1}^{N} {e_i}^2 = \frac{1}{N}\sum_{i=1}^{N}(t_i - a_i)^2 \quad (7)$$

Training can be performed in two different ways: *batch mode* and *incremental mode*.

- In batch training, weights and biases are not adjusted before all the inputs and targets are presented. It can be applied to both static and dynamic networks.
- In the incremental mode, however, the weights and biases are updated after application of each input to the network. In this case, the inputs and targets are presented as sequences.

For multilayer feedforward networks, any numerical optimization training algorithm can be used to optimize the performance function, but there are a few that have resulted in excellent performance for neural network training.

3.3.4.1 Training Algorithm

Many of the training algorithms available in the Neural Network Toolbox use gradient- or Jacobian-based methods. The simplest optimization algorithm is the gradient descent, which adjusts the weights and biases in the direction in which the performance function decreases most rapidly. The fastest training function is the Levenberg-Marquardt (LM), which tends to be less efficient with networks with more than thousands of weights. For large networks, the Scaled Conjugate Gradient and Resilient Back Propagation are the best choices. [27]

The simplest commands used to initiate the training process in MATLAB are the following:

Load dataset

Net= feedforward (20);

[net, tr] = train (net, inputs, Targets);

Dataset contains the input and target data. The training window that appears during training is shown in Fig. 3.9 below.

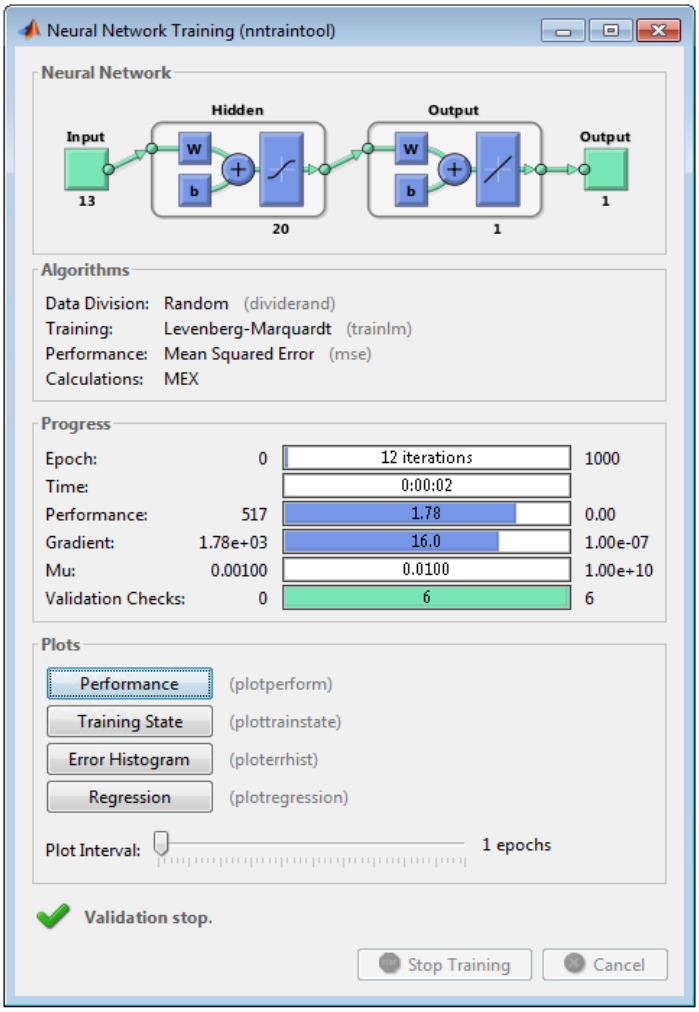

Figure 3.9: Neural Network Training Window in MATLAB [27]

The progress of the training is constantly updated in the training window. The most important pieces of information are the performance, the magnitude of the gradient of performance and the number of validation checks. The magnitude of the gradient of performance and the number of validation checks are used to stop the training. As the training reaches a minimum of performance, the gradient becomes very small. The number of validation checks is the number of successive iterations that the validation performance fails to decrease. Other criteria can also be used to stop the network training: minimum

gradient magnitude, maximum training time, maximum number of validation increases and minimum performance value.

3.3.4.2 Analysis and Testing

Four plots can be accessed from the training window seen in Fig. 3.9:

- Performance: It shows the value of the performance function with respect to the iteration number. It plots training, validation and test performances.
- Training state: It shows other training variables such as the number of validation checks, the gradient magnitude, etc.
- Error histogram: It shows the distribution of the network errors.
- Regression: It shows a regression between network outputs and network targets. Figure 3.10 shows an example.

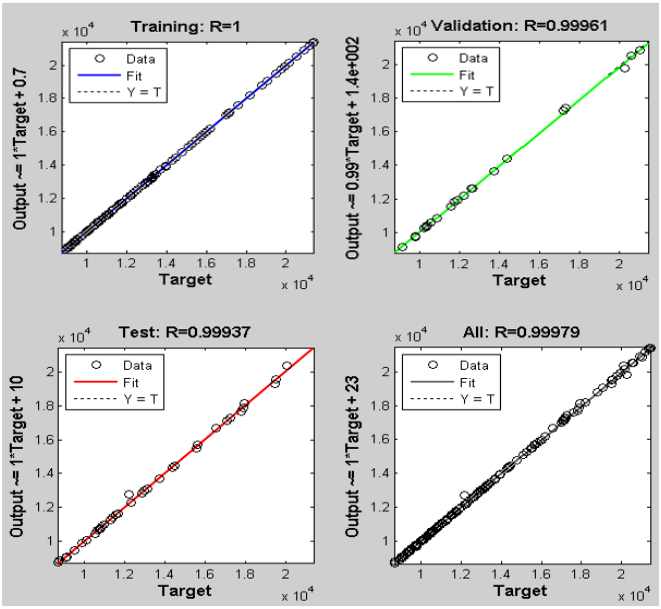

Figure 3.10: Example of regression plot

Figure 3.10 above shows the regression plot for a neural network training where 75% of the samples were used for training, 15% for testing and 10% for validation. The regression number is close to 1 for Training, Validation and Testing, showing how well trained the Neural Network is.

The trained neural network can then be used to calculate the output corresponding to any input. It is important to mention that different neural networks trained for the same

problem can result in different outputs for the same input. This is due to different initial weight and bias values and different divisions of data into training, validation and tests sets each time a neural network is trained. [27]

3.3.4.3 Improve Neural Network Training and Avoid Overfitting

Overfitting is one of the problems that may occur during Neural Network training. It is characterized by the trained neural network resulting in a large error when new data is given as input even though the error on the training set is very small. This means that the network was successful in learning the training samples, but failed to learn the generalization to a new situation. Three methods are used to improve network generalization:

- Large *enough* network: The larger the network is, the more complex functions the network can create. A small *enough* network is not powerful enough to overfit the data. As a result, reducing the size of a network can prevent overfitting. Also, there is no risk of overfitting if the number of parameters in the network is much smaller than the total number of points in the training set.
- Regularization: This involves using a performance function other than the *sum of squares of the network errors* on the training set. The performance function can be modifed by adding a term that consists of the mean of the sum of squares of the network weights and biases: [28]

$$msereg = \gamma * msw + (1 - \gamma) * mse \quad (8)$$

Where γ is the performance ratio and,

$$msw = \frac{1}{N}\sum_{i=1}^{N} w_j{}^2 \quad (9)$$

- Early Stopping: Another method to improve generalization is *early stopping.* This technique is available for all supervised algorithms. In this technique, the available samples are divided into 3 groups. The first is used for calculating the gradient and adjusting the network weights and biases. This is the training set. The second is the validation set. The error on this set is tracked during the training process. When the validation error increases for a specific number of iterations, the training stops, and the weights and biases at the minimum of the validation error are returned. The third set is the test set. This set is used to compare different models. If the error in

the test set reaches a minimum at a significantly different iteration number than the validation set error, this might be a consequence of a poor division of the data set. [28]

3.3.5 Limitations

Neural Network training with the MATLAB Toolbox has several limitations:

- Neural Networks training requires the availability of a large number of previous cases.
- There is no way to determine the best neural network topology for a specific problem.
- The reliability of the results decreases as the complexity of the neural network increases. Several experiments with various network architectures must be performed to overcome this limitation.
- Neural network training is highly sensitive to the number of neurons and the number of hidden layers. Too few neurons can lead to underfitting; too many to overfitting. It is also sensitive to the training algorithm, the number of points initially provided, division of data into training, validation and tests and many other parameters. This makes it difficult to decide on the most effective choice of parameters to train the neural network. [29]
- The Levenberg-Marquardt (LM) training is only for small and medium size networks if there is enough memory available.
- Backpropagation and its variants are not always able to find a solution.
- Because nonlinear transfer functions in multilayer networks create many local minima in the error surface, the error surface of a nonlinear network is more complex than the error surface of a linear network. Being trapped in a local minimum is bad if it is far from the global minimum. As a result, the backpropagation algorithm might not always be able to find the correct weights for the optimum solution. A Neural Network must then be trained several times to ensure the best solution. [28]

A method to determine the Neural Network with the best number of neurons is presented in the results.

4 Genetic Algorithm

4.1 Introduction

Every organism has a set of genes describing how that organism is built up from the smallest building blocks of life. The genes are connected together into chromosomes. The transmission of gene's offspring is the foundation of the inheritance of phenotypic traits, such as eye colour or hair colour. When two organisms breed, they share genes and the resultant progeny may have genes from both parents in a process called *recombination.* Sometimes, there can be *mutation*, where the gene does not affect the development of the phenotype [30]. The evolution of life on earth is based on the processes of natural selection, recombination and mutation.

Genetic Algorithms were introduced by John Holland in 1970 to simulate some of these processes. As such, they constitute an intelligent exploitation of a random search to solve both constrained and unconstrained optimization problems. The basic idea of the GA is to simulate Charles Darwin's principle of "survival of the fittest" [31]. A GA can be applied to solve problems that are not well suited for standard optimization algorithms, including problems in which the objective function is discontinuous, nondifferentiable, stochastic, or highly nonlinear.

GAs mimic the survival of the fittest among individuals over consecutive generations for solving a problem. Each generation includes a population of strings that are comparable to the chromosomes in the DNA. Each individual constitutes a point in a search space and a possible solution, and it goes through a process of evolution. The foundations of GA are [32]:

- Individuals in a population compete for resources and mates.
- The most successful individuals will produce more offspring.
- Genes from good individuals proliferate throughout the population so that better offspring can be produced.

- As a result, each successive generation will become more suitable for its environment.

4.2 Genetic Algorithm Operators

The simplest form of genetic algorithm involves three types of operators:

- Selection: It is equivalent to survival of the fittest. It is the stage of a GA where the chromosomes are selected in the population for reproduction. The fitter the chromosome, the most likely it will be selected. Keeping the best individuals in a generation unchanged in the next generation is termed *elitism* of *elitist selection*. It is a successful form of the general process of constructing a new population.
- Crossover: It represents mating between individuals. It selects randomly a locus and exchanges the subsequences before and after that locus between two chromosomes to create two offspring [31]. This emulates biological recombination between two single haploid organisms. There are several variants of crossover:
    - Single-point crossover (see Fig. 4.1): It generates a cut-point on each parent and recombines the first part of the first parent with the second part of the second parent to generate one offspring. The second offspring is generated by recombining the second part of the first parent with the first part of the second parent.

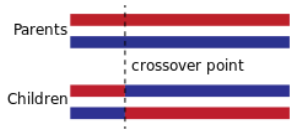

Figure 4.1: Single-point crossover [9]

    - Two-point crossover (see Fig. 4.2): Two crossover points are selected randomly.

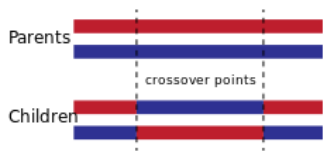

Figure 4.2: Two-point crossover [9]

- Cut and splice (see Fig. 4.3): It results in change in length of the children strings. This is due to the fact that each parent has a different choice of crossover point.

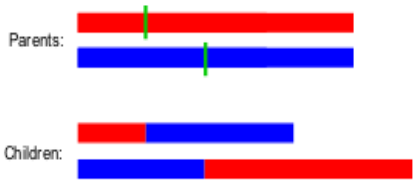

Figure 4.3: Cut and Splice Crossover

- Mutation (see Fig. 4.4): It randomly inverts some of the bits in a chromosome. This operator is used to maintain genetic diversity from one generation of a population of genetic algorithm chromosomes to the next. [34]

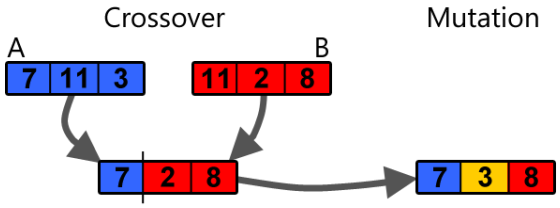

Figure 4.4: Mutation [51]

Some facts are important to know when using GA operators [33]:

- Using only the selection operator will tend to fill the population with copies of the best individual from the population.
- Using mutation alone causes a random walk through the search space
- Using both selection and crossover operators will tend to push the algorithm to converge on a local optimum.
- Using both selection and mutation creates a parallel, noise-tolerant, hill climbing algorithm.

4.3 Genetic Algorithm Process

A simple GA works as follows:

1- Start with a randomly generated population of *n* candidate solutions to a problem.
2- Calculate the fitness function of each candidate solution.
3- Repeat the following until *n* offspring have been created:
    a. Select a pair of parent chromosomes from the current population, the probability of selection being an increasing function fitness. Selection is done with replacement, meaning that the same chromosome can be selected more than once to become a parent.
    b. Using a crossover rate, cross over the pair at a randomly chosen point to create two offspring. If there is no crossover, form two offspring that are exact copies of their respective parents.
    c. Mutate the two offspring at each locus, and place the resulting chromosomes in the new population.
4- Replace the current population with the new population
5- Return to step 2

Each iteration of this process is called a *generation*. The number of generations is usually between 50 and 500. Figure 4.5 below shows an example of a generation plot with the populations at iterations 60, 80, 95, and 100.

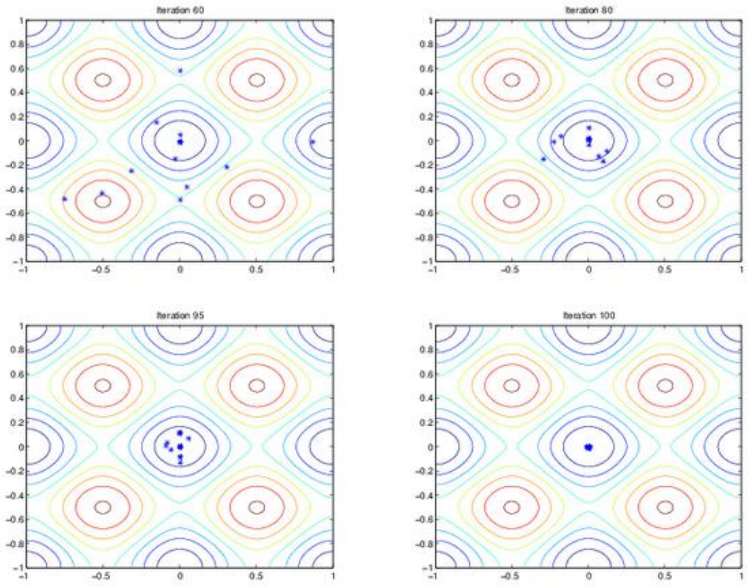

Figure 4.5: Plot of populations at iterations 60, 80, 95, and 100.

As the number of generations increases, the individuals in the population get closer to each other and approach the minimum.

A *run* is the entire set of generations. One or more highly fit chromosomes remain in the population at the end of a run.

The simple process described above is the basis for most GAs. Other details must be taken care of, such as the size of the population, the rates of crossover and mutation.

4.4 Genetic Algorithm vs. Classical Algorithm

The GA differs from a classical, derivative-based optimization algorithm in two main ways that can be summarized in Table 4.1 below. [35]

Table 4.1: Classical Algorithm vs. Genetic Algorithm

Derivative-based Algorithm	**Genetic Algorithm**
Creates a single point at each iteration. The sequence of points approaches an optimal solution.	Generates a population of points at each iteration. The best point in the population approaches an optimal solution.
Selects the next point in the sequence by a deterministic computation.	Selects the next population by computation which uses random number generator.

4.5 Genetic Algorithm Toolbox in MATLAB

The GA options provided by the toolbox in MATLAB are summarized in Table 4.2 below.

Table 4.2: GA options in MATLAB

Step	**Genetic Algorithm Option**
Creation	Uniform, feasible
Fitness scaling	Rank-based, proportional, top (truncation), shift linear
Selection	Roulette, stochastic uniform selection (SUS), tournament, uniform, remainder
Crossover	Arithmetic, heuristic, inter-mediate, scattered, single-point, two-point
Mutation	Adaptive feasible, Gaussian, uniform
Plotting	Best fitness, best individual, distance among individuals, diversity of population, expectation of individuals, max constraint, range, selection index, stopping conditions

It is possible to specify:

- Population size
- Number of elite children
- Crossover fraction
- Migration among subpopulations
- Bounds, linear, and nonlinear constraints for an optimization problem.

It is also possible to carry multiobjective optimization using the GA Toolbox in MATLAB, and the following options can be specified along with the ones specified above:

- Pareto fraction
- Distance measure across individuals

Table 4.3 below shows the available options for multiobjective optimization using GA.

Table 4.3: Mutlitobjective GA Option

Step	**Multiobjective Genetic Algorithm Option**
Creation	Uniform, feasible
Fitness scaling	Rank-based, proportional, top (truncation), linear scaling, shift
Selection	Tournament
Crossover	Arithmetic, heuristic, inter-mediate, scattered, single-point, two-point
Mutation	Adaptive feasible, Gaussian, uniform
Plotting	Average Pareto distance, average Pareto spread, distance among individuals, diversity of population, expectation of individuals, Pareto front, rank histogram, selection index, stopping conditions

In this project, a GA is used to optimize control surface deflections to minimize the Sum of Absolute Values of Hinge Moments. The GA provides the optimum combination of control surface deflections and extracts its expected Sum of Absolute Values of Hinge Moments from the trained ANN.

5 Static Aeroelasticity

Aeroelasticity is the study of the interaction between inertial, aerodynamic and structural forces (see Fig. 5.1) on aircraft, other vehicles and buildings [36]. If airplane structures were perfectly rigid, there would be no aeroelastic problems. Because modern aircraft are very flexible, they are subject to various undesirable phenomena such as *divergence*, *flutter, limit cycle oscillations, vortex shedding*, etc...

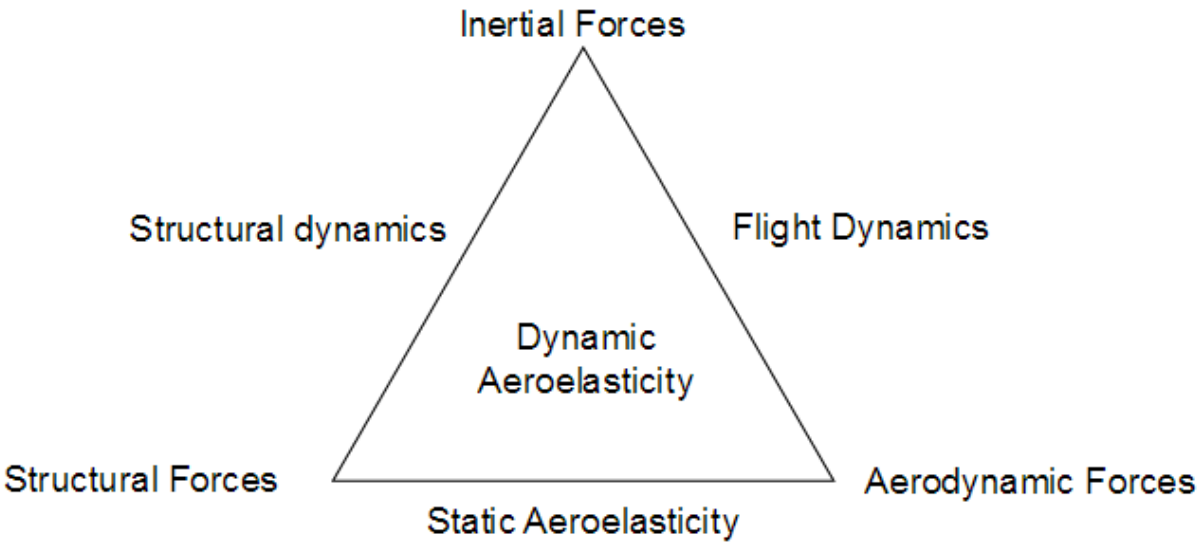

Figure 5.1: Aeroelasticity [36]

Static Aeroelasticity is the study of mutual interactions between elastic and aerodynamic forces and their influence on the airplane. It includes:

- Load distribution: The influence of elastic deformations of the structure on the distribution of aerodynamic pressure over the aircraft structure.
- Divergence: It occurs at the *divergence speed* above which a lifting surface deflects under condition where no statically stable equilibrium condition exists; the deformation increases to a point of structural failure.
- Control system reversal: A condition occurring in flight at a speed called the control reversal speed, at which the intended effect of displacing a given component of the

control system are nullified or reversed by elastic deformations of the structure. [37]

5.1 Longitudinal and Lateral stability

An HWB, or FSW, or any aircraft put in a state of equilibrium by adjusting control inputs is flying in a *trimmed* condition. For an airplane to be in equilibrium for a particular flight condition, the sum of all the forces and moments on it must be equal to zero [38]. For an airplane in a straight and *level flight* (see Fig. 5.2), the lift equals the weight, the thrust equals the drag, and there is no net rotating moment.

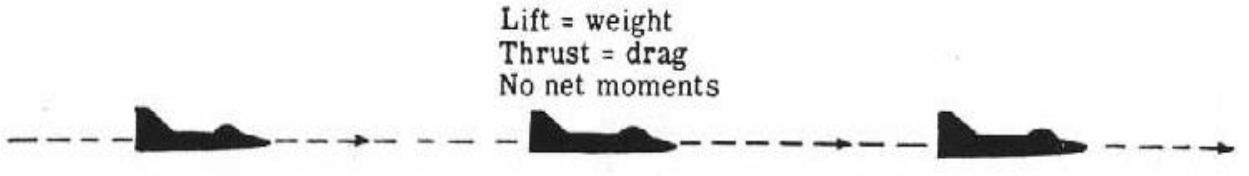

Figure 5.2: Equilibrium flight [38]

If there is an atmospheric turbulence so that the aircraft is disturbed, the airplane will no longer be in equilibrium. If the forces and moments on the aircraft in this condition tend to increase the angle of attack, the aircraft will be statically unstable and its motion will *diverge* from equilibrium (see Fig. 5.3). However, if the aircraft tends to hold the disturbed position, then the aircraft has *neutral static equilibrium* (see Fig. 5.4).

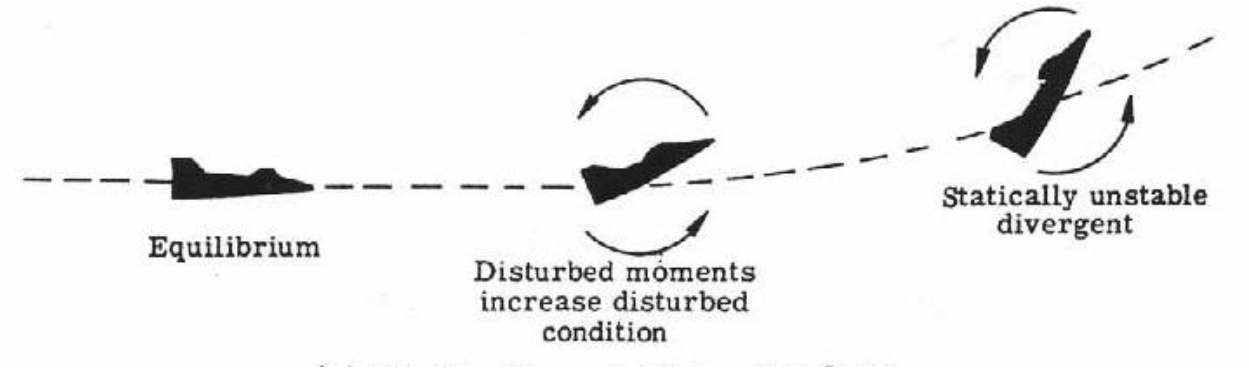

Figure 5.3: Statically unstable airplane [38]

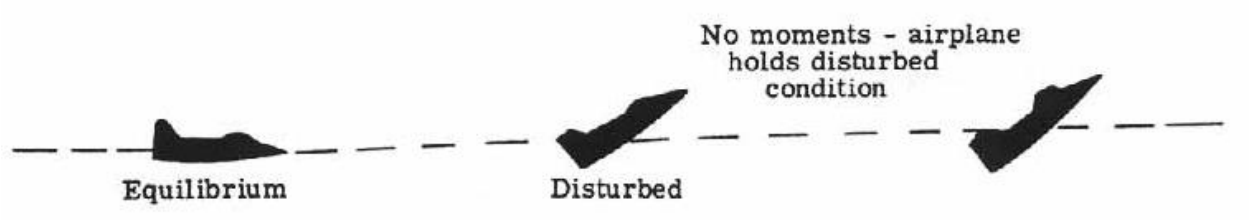

Figure 5.4: Neutral static stability [38]

Finally, if restoring forces and moments are generated by the airplane that tend initially to bring it back to its equilibrium straight and level condition, the aircraft will be *statically stable*.

5.1.1 Longitudinal Stability

Longitudinal stability and control is concerned with an airplane's pitching motion. Figure 5.5 shows how pitch equilibrium is achieved for an airplane. [38]

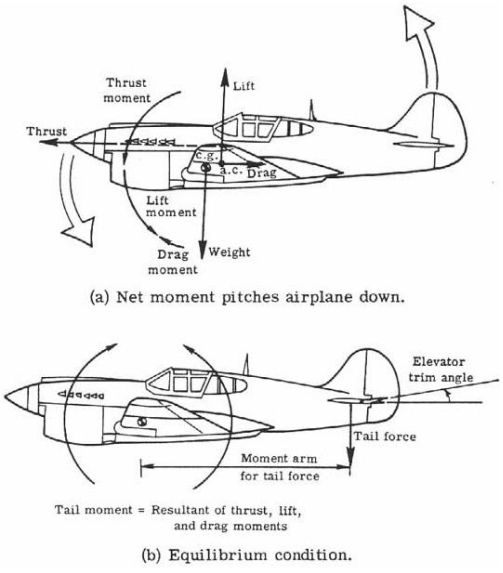

Figure 5.5: Longitudinal stability: Pitch equilibrium [38]

To fly in particular equilibrium condition, the control surfaces are trimmed to particular angles, such that the total moment about the airplane center of gravity is zero.

5.1.2 Lateral and Directional Stability

Lateral stability relates to an airplane's rolling motion, and directional stability is concerned with an airplane's yawing motion. Because lateral and directional stability are interrelated, they are sometimes referred to as lateral stability [38]. In the usual directional equilibrium condition, an airplane flies so that the yaw angle is zero as shown in Fig. 5.6.

An airplane is said to possess lateral static stability if after being subject to a disturbance that rolls it to some bank angle, it generates forces and moments that tend to reduce the bank angle and restore the equilibrium flight condition.

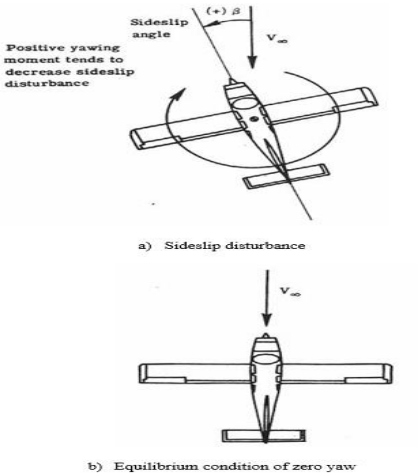

Figure 5.6: Static directional stability [38]

5.2 Static Aeroelasticity in MSC Nastran

5.2.1 Introduction

Static aeroelastic problems involve the interaction of aerodynamic and structural forces on a flexible structure that leads to a redistribution of the aerodynamic loading as a function of airspeed. This aerodynamic load redistribution and resulting internal structural load and stress redistributions are important to the structural analyst. Also important is the probability of a static aeroelastic instability and divergence. The control systems analyst and the aerodynamicist are also concerned by the aerodynamic load redistribution and resulting modifications to aerodynamic stability and control derivatives.

These concerns can be treated by the static aeroelastic capability in MSC Nastran through computation of aircraft trim conditions, with retrieval of structural responses, aeroelastic

stability derivatives, and static aeroelastic divergence dynamic pressures [39]. The static aeroelastic solution sequence (SOL 144) capabilities in MSC Nastran are:

- The finite element models for the definition of the structure and aerodynamic loading, including information on the flight condition can be supplied by the user.
- Stability and control derivatives are printed in the output (f06) file for every single flight condition (Mach number and dynamic pressure).
- Aircraft trim conditions: A trim analysis is conducted to calculate unknown trim values and then carry out standard data recovery for each TRIM subcase defined in the Case Control section of the input data file. AEROF and APRES Case Control Commands can be used to acquire aerodynamic forces and pressures on the aerodynamic elements respectively.
- A DIVERG Case Control command allows for static aeroelastic divergence analysis. The divergence is carried out for Mach numbers specified on the DIVERG Bulk Data entry.
- Three matrices are available for altering the theoretically predicted aerodynamics. Correction factors can be input using WKK; experimental pressures can be input using FA2J and adjustments to the downwash; the effects of camber and twist, can be input using matrix W2GJ. [40]
- Rigid/Flexible mesh concept.

5.2.2 Aerodynamic Analysis in MSC Nastran

MSC Nastran has implemented seven internal aerodynamic theories: [40]

1. Doublet-Lattice subsonic lifting surface theory (DLM): It can be used for interfering lifting surfaces in subsonic flow. The theory was presented by Albano and Rodden in 1969. The theoretical basis of the DLM is linearized aerodynamic potential theory. The DLM is an extension of the steady Vortex-Lattice method to unsteady flow.

2. ZONA51 supersonic lifting surface theory: It is a supersonic lifting surface theory that accounts for the interference among multiple lifting surfaces. It is an optional feature in MSC Nastran (available as the Aero II option). It is similar to the Doublet-Lattice method (DLM) in that both are acceleration potential methods that need not account for flow characteristics in any wake.

3. Constant Pressure Method for supersonic lifting surface theory: MSC's acquisition of UAI brought with it the availability of another supersonic aerodynamic method entitled CPM (Constant Pressure Method). This method is available as an alternative to ZONA51 in that it provides the same basic functionality and uses the same user interface as the ZONA51 code.

4. Subsonic wing-body interference theory (DLM with slender bodies)

5. Mach Box method

6. Strip Theory

7. Piston Theory

The following work, however focuses on Vortex-Lattice subsonic lifting surface theory. The relationships required to define a set of aerodynamic influence coefficients are the basic relationships between the lifting pressure and the dimensionless vertical or normal velocity induced by the inclination of the surface to the airstream. They include:

- The downwash,

$$\{w_j\} = [A_{ij}]\{f_j/q\} \tag{10}$$

- The substantial differential matrix of the deflections to obtain downwash,

$$\{w_j\} = \left[D_{jk}^1 + ikD_{jk}^2\right]\{u_k\} + \{w_j^g\} \tag{11}$$

- And the integration of the pressure to obtain forces and moments,

$$\{P_k\} = [S_{kj}]\{f_j\} \tag{12}$$

Where,

w_j = downwash

$A_{ij}(m,k)$ = aerodynamic influence coefficient matrix as a function of Mach number (m) and reduced frequency (k)

f_j = pressure on lifting element j

q = flight dynamic pressure

w_j^g = static aerodynamic downwash which includes the static incidence distribution that may arise from an initial angle of attack, camber, or twist.

k = reduced frequency, $k = \omega b/V$ where ω is the angular frequency, b is a reference length and V is the free-stream velocity.

u_k = displacement at aerodynamic grid points

D_{jk}^1 , D_{jk}^2 = real and imaginary parts of substantial differential matrix

S_{kj} = integration matrix

The above three equations can be combined to give an aerodynamic influence coefficient matrix:

$$[Q_{kk}] = [S_{kj}][A_{ij}]^{-1}[D_{jk}^1 + ikD_{jk}^2] \quad (13)$$

For different user-supplied Mach numbers and reduced frequencies, all aerodynamic methods can be used to compute *S*, D^1 and D^2 matrices. The *A* matrix is calculated by the Doublet-Lattice and ZONA51 theories. Then, the computation of the *Q* matrix is performed by matrix decomposition and forward and backward substitution. The remaining methods compute A^{-1} directly and use matrix multiplications to form *Q*. For static aeroelastic analysis, reduced frequencies are 0 as analysis is time independent. Equation 13 becomes:

$$[Q_{kk}] = [S_{kj}][A_{ij}]^{-1}[D_{jk}] \quad (14)$$

The above aerodynamic equations form the basis of the aerodynamic computations required for static aeroelastic analysis with some special purpose modifications made for the MSC Nastran implementation.

For static aeroelasticity, the downwash relation of Eq. 11 becomes

$$\{w_j\} = [D_{jk}]\{u_k\} + [D_{jx}]\{u_x\} + \{w_j^g\} \quad (15)$$

Where:

$\{w_j\}$ = vector of aerodynamic degrees of freedom (e.g. angles of attack)

$\{u_k\}$ = vector of aerodynamic deformations (displacements)

$\{u_x\}$ = vector of "extra aerodynamic points" used to describe aerodynamic control surface deflections and overall rigid body motions

$\{w_j^g\}$ = an initial static aerodynamic downwash. It primarily includes the static incidence distribution that may arise form initial angle of attack, camber or washout.

$[D_{jk}]$ = substantial derivative matrix for the aerodynamic displacements

$[D_{jx}]$ = substantial derivative matrix for the extra aerodynamic points.

The theoretical aerodynamic pressures are given by:

$$\{f_j\} = \bar{q}[A_{jj}]^{-1}\{w_j\} \tag{16}$$

And, the aerodynamic forces can be written as:

$$\{P_k\} = \bar{q}[W_{kk}][S_{kj}][A_{jj}]^{-1}\{w_j\} + \bar{q}[S_{kj}]\{f_j^e/\bar{q}\} \tag{17}$$

The aerodynamic forces (from Eq. 17) can be transferred to the structural forces using the spline matrix and can be reduced to *a-set* to form the aerodynamic influence coefficient matrix Q_{aa} as,

$$[Q_{aa}] = [G_{ka}]^T[W_{kk}][S_{kj}][A_{jj}]^{-1}[D_{jk}][G_{ka}] \tag{18}$$

The second aerodynamic influence coefficient matrix Q_{ax} for forces at the structural grid points due to unit deflections of the aerodynamic extra points can be written as:

$$[Q_{ax}] = [G_{ka}]^T[W_{kk}][S_{kj}][A_{jj}]^{-1}[D_{jx}] \tag{19}$$

The complete equations of motion in the a-set degrees of freedom can be written as:

$$[K_{aa} - qQ_{aa}]\{u_a\} + [M_{aa}]\{\ddot{U}_a\} = q[Q_{ax}]\{u_x\} + \{P_a\} \quad (20)$$

Where,

K_{aa} = Structural stiffness matrix

M_{aa} = Structural mass matrix

P_a = Vector of applied loads (Including mechanical, thermal, and gravity loads plus aerodynamic terms due to user input pressures and/or downwash velocities).

Equation 20 is the basic set of equations used for static aeroelastic analysis. In the general case, rigid body motions are included in the equations to represent the free-flying characteristic of an air vehicle. This is addressed in MSC Nastran by a requirement that the user identify reference degrees of freedom equal in number to the number of rigid body motions using the SUPORT Bulk Data entry.

In the current work, MSC Nastran's structural finite elements were used to build the structural model. Geometric, structural, inertial and damping data were provided to MSC Nastran, and the structural stiffness, mass and damping matrices were generated for the aeroelastic analyses. Aircraft stability derivatives are then obtained from the loads and deflections.

5.3 Hinge Moments

During the preliminary design of a hydraulic or electric actuator system, it is necessary to estimate the hinge moments due to control surface deflections. The hinge moment is the moment acting about the hinge line of a control surface. In other words, it is the resistance that must be overcome to deflect a specific control surface. For a given airspeed and dynamic pressure, the hinge moment varies with the angle of attack and the amount of control surface deflection.

A free control surface will float, in the static case, to the position where the control surface hinge moment is zero.

$$HM_{cs} = 0 \quad (21)$$

A control surface hinge moment is usually expressed as the *hinge moment coefficient* or *hinge moment derivative.*

$$C_{HM} = \frac{HM}{Q*S*c} \tag{22}$$

where *S* is the reference area, *Q* the dynamic pressure, and c the moment arm of the control surface aft of the hinge line.

If we assume that the hinge moment is a linear function of angle of attack and the control surface deflections then,

$$C_{HM_e} = C_{HM_{e_0}} + C_{HM_\alpha}\alpha + C_{HM_{\delta_e}}\delta_e + C_{HM_{\delta_t}}\delta_t \tag{23}$$

Here, α is the angle of attack (from angle for zero vehicle lift), δ_e is the elevator deflection, and δ_t is the deflection of the control tab. C_{HM_α} is the hinge moment derivative due to the changes in the angle of attack. It is called the *floating tendency*, because an increase in the angle of attack usually causes the control surface to float upward. [41]

The derivative $C_{HM_{\delta_e}}$ represents the hinge moment by a deflection of the control and is called the *restoring tendency*, since the nose-down hinge moment generated by a positive control deflection tends to restore the control to its original position. The aerodynamic forces responsible for generating the hinge moments in the floating and restoring tendencies can be visualized in Figs. 5.7 and 5.8 below.

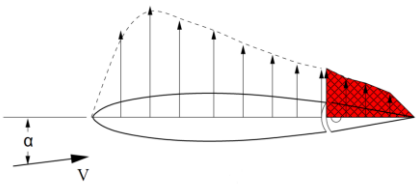

Figure 5.7: Floating tendency of trailing edge control surfaces [41]

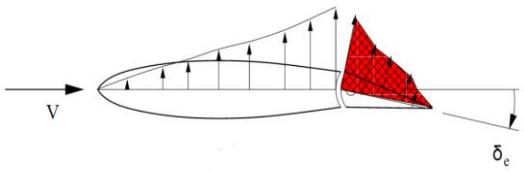

Figure 5.8: Restoring tendency of trailing edge control surfaces [41]

The floating (or restoring) tendency represents the moment about the hinge line of the lift distribution acting on the control surface per unit angle of attack (or control deflection). Only the red-shaded portion of the lift distribution in Figs. 5.7 and 5.8 acts on the control surface and contributes to the hinge moments.

Equation 23 above was derived assuming a linear system, however hinge moments are very difficult to determine. In fact, several factors can influence hinge moments including: [42]

- Nonlinearities in the wing-alone normal-force curve (versus angle of attack).
- Nonlinearities in the wing-alone longitudinal center of pressure curve (versus angle of attack).
- Fin-fin interference.
- Canard vortices (effects on aft surfaces)
- Adverse fin-body interference
- Gap effects.
- Aeroelastic effects.
- Fin-body interference (no body vortices).
- Fin-body interference due to body vortices.
- Fin trailing-edge shock-wave/boundary-layer interaction.
- Fin choking

One effect that is of importance is that of airfoil thickness ratio and thickness distribution which can have a significant effect on the center of pressure of fin normal force. Furthermore, vortices such as those of the body or canard fins passing in close proximity to the all-movable fin can significantly change its center-of-pressure position and normal force and hence its hinge moment. The nonlinearities resulting from higher angle of attack operation make the prediction of the control characteristics even more difficult. As a result, hinge moments are almost always measured in wind tunnels during the design stage to assure that the control surfaces will generate the desired trim and maneuverability within the hinge moment capabilities of the control actuators.

As an initial figure of merit, the Sum of Absolute Value of Hinge Moment was chosen here, for control power to be minimized.

6 Proposed Approach

The methodology used with the HWB [11] summarized below was also applied on the FSW. Because the HWB has a large number of control surfaces, the optimization process for appropriate allocation of control deflections to achieve minimum Sum of Absolute Values of Hinge Moments uses the following hypotheses.

6.1 Hypotheses

- Hypothesis 1

MSC Nastran is used to generate a dataset, which includes the control surface deflections and the corresponding hinge moment values. With this data, an ANN is trained, and it accurately represents the static aeroelastic model behavior.

- Hypothesis 2

The trained ANN is then used as a surrogate model in the optimization process using a GA. The optimum control allocation is provided by the GA, and the Sum of Absolute Values of Hinge Moments corresponding to that control allocation is given by the trained ANN.

- Hypothesis 3

The Sum of Absolute Values of Hinge Moments and the control allocation from the optimization process is the optimum, which can be validated only by MSC Nastran.

6.2 Initial Optimization Process

The initial optimization process is displayed in Fig. 6.1 and described below.

6.2.1 Dataset Generation using MSC Nastran

The steps for generating the dataset using MSC Nastran are as follows:

- Provide the BDF file with the multiple sets of deflections for all the control surfaces, except for the elevator and angle of attack. It needs to be kept in mind that the control deflections should not have values which are difficult or impossible to

attain in practice. Therefore, the control surface deflections were kept between -37.5 and +37.5 degrees.

- Save those deflections in a separate file, as these are the design variables.
- MSC Nastran performs trim analysis using SOL 144 for those specific deflections.
- SOL 144 gives its output (including elevator deflection and AoA) in a F06 file along with all the other data.
- Hinge moments are not direct outputs of MSC Nastran. Therefore, the values for hinge moments are extracted manually from the F06 file.
- The objective function for the optimization is Sum of Absolute Values of Hinge Moments for all the control surfaces.

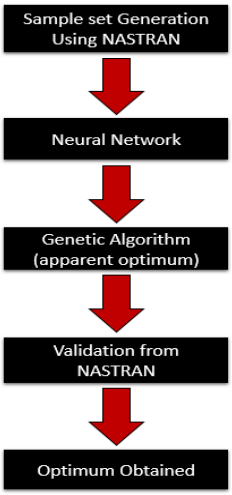

Figure 6.1: Initial Optimization Schedule

An issue with this process is that there can be any combinations of deflections for the 3 control surfaces on the modified FSW. Therefore, a randomly distributed dataset is generated using Latin Hypercube Sampling (LHS) [43] over AELINK cards. AELINK [44] defines relationships among aeroelastic static trim variables (which specify rigid body motions to be used as trim variables in static aeroelasticity), and aerodynamic control surface deflections such that:

$$u^D + \sum_{i=1}^{n} Ci\, u_i^I = 0.0 \tag{24}$$

Where:

u^D = dependent variable

u_i^I = independent variable

The LHS method has a smaller variance with respect to other sampling methods such as Primitive Monte Carlo (PMC) sampling. These AELINK cards are generated using *lhsdesign* in MATLAB, which creates randomly spaced combinations with values between 0 and 1. These values are then mapped to values between -1.2 and 1.2. The control surface deflections (INB and OUT) would be directly related to the elevator deflection and indirectly dependent on the AoA. Therefore, the BDF file with the AELINK cards, in lieu of the deflections, along with the BDF file of the FSW model were provided to MSC Nastran. A MATLAB script then reads the trim variables, and hinge moment for each individual control surface, storing the AELINK Cards, control surface deflections and hinge moments for post-processing and optimization.

6.2.2 Data Segregation

Because the relationship between the control surface deflections and Sum of Absolute Values of Hinge Moments is noisy, a small change in the control surface deflections could cause a significant change in Sum of Absolute Values of Hinge Moments, especially for the elevator. This makes it difficult to train the ANN between the inputs and the targets with small number of samples. However, to avoid training the ANN with unnecessary samples, one can simply target a space with an adequate number of samples. The goal of this work is to determine the best combination of control surface deflections to minimize the Sum of Absolute Values of Hinge Moments. So, the ANN needs to be accurate for samples with low Sum of Absolute Values of Hinge Moments. Therefore, instead of using the whole dataset for training the ANN, only those samples were used where the Sum of Absolute Values of Hinge Moments values were lower than the mean of Sum of Absolute Values of Hinge Moments for the complete dataset.

6.2.3 Surrogate Model using an Artificial Neural Network

Artificial intelligence with an ANN is used for the control allocation problem to determine the best set of control surface deflections for the Sum of Absolute Values of Hinge Moments minimization. The Sum of Absolute Values of Hinge Moments, which is the objective function is obtained from MSC Nastran SOL 144. However, SOL 144 is

computationally expensive and cannot be used during the optimization. Since ANN is a way of creating a surrogate model for costly objective functions that are usually dependent on a large number of inputs, it is suitable for estimating the objective function for the GA. In order to choose the best number of neurons to train the ANN, the root-mean-square error will be evaluated, and the ANN with the minimum error will be chosen.

6.2.4 Optimization with Genetic Algorithm

The trained ANN is used as the objective function by a GA to find an optimum. GA starts by making small changes in the individuals in the initial population to generate children using mutation, which allows for genetic diversity. The resulting optimum is corroborated against an MSC Nastran computation.

6.2.5 Validation

Because a surrogate model (ANN) is used as the objective function during the optimization, it is necessary to check whether the results obtained from the GA are feasible and physical or not. Thus, the Sum of Absolute Values of Hinge Moments obtained from the ANN must be validated. The optimum obtained from the optimization is then fed back into MSC Nastran, and the resulting Sum of Absolute Values of Hinge Moments is compared with the Sum of Absolute Values of Hinge Moments from the optimization.

6.3 Improvements

During the optimization, it is possible that the minimum might not be within the data set space used to train the ANN. This results in extrapolation by the GA and usually happens when the global minimum Sum of Absolute Values of Hinge Moments is lower than the minimum present in the data set. It is then necessary to improve the accuracy of the ANN outside the bounds of the dataset initially used to train the ANN.

Our strategy is to increase the bounds of the data set gradually, as the process moves forward. This prevents the ANN from going outside its bounds every time an optimum set of control deflections is generated by GA. This process also helps to increase the number of samples for training ANN, which may even improve the network further.

As a result, the initial optimization process (Fig. 6.1) was improved to the one shown in Fig. 6.2.

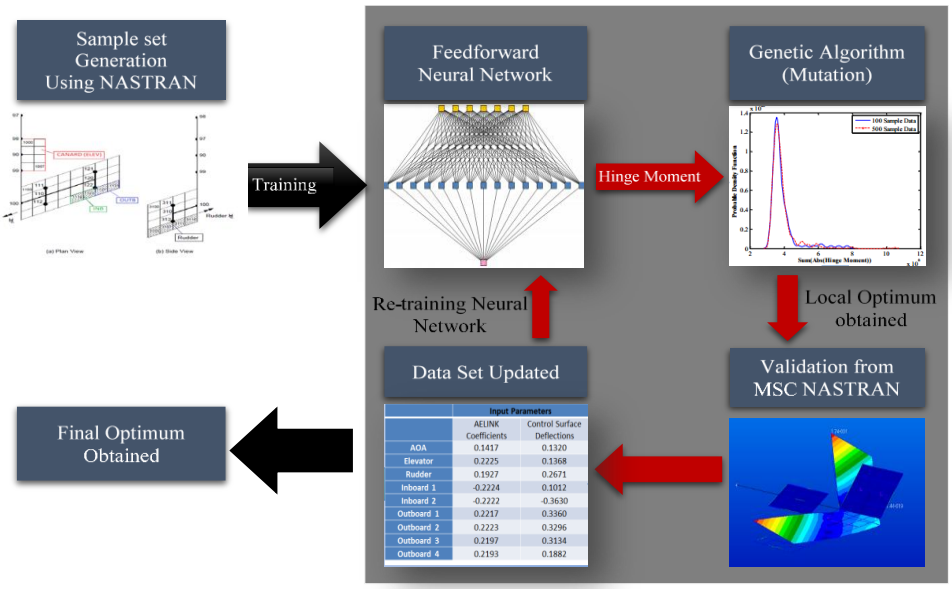

Figure 6.2: Improved Optimization schedule with Data Set Updating and Retraining of Neural Network

In the new and improved optimization process, the minimum obtained from the GA is presumed to be just a local minimum that is highly dependent on the nature of the initial population, number of generations and the type of modifications GA makes to the initial population. After validation of the local minimum using MSC Nastran, the specific control surface deflections and their respective Sum of Absolute Values of Hinge Moments are added to the initial set of data used to train the ANN. So, now the new data set has *(n+1)* samples, where n is number of samples in the original data set. Then, using this new expanded data set, the neural network is trained again. This training will improve the bounds of the network, so that it can be more accurate in calculating the Sum of Absolute Values of Hinge Moments for the given control surface deflections. GA is then re-run using the solution from the last iteration as the new initial values of the variables. The process is continued until a final optimum is obtained.

7 Results

The modified FSW case (see Sect. 2.2) was selected as a simple test case to learn more about the whole process. This proved to be a wise approach.

7.1 Forward Swept Wing Aircraft Case

The first analysis conducted on the modified FSW was to trim the airplane. SOL 144 is MSC Nastran's solution executive sequence used for static aeroelastic trim analysis. It computes deflections, stresses, loads and hinge moments. Since only longitudinal motion was of interest in this study, the rudder was locked. The load case considered is a pull-up maneuver at 2.5-G, Mach number of 0.9 at sea level. The aeroelastic degrees of freedom required to trim the airplane include: angle of attack (AoA), pitch rate, normal load factor and pitch acceleration. Note that there is no pitch rate considered here [47]. Since only two free variables were needed to trim the airplane, the other variables were fixed to zero or to a specific value. The second step of the analysis involved the use of AELINK cards to trim the airplane. Instead of fixing the other control surfaces to a specific value, they were linked to one of the free variables (independent variables) by the relationship in Eq. 24 above.

Two cases were studied. For the first case, the AoA and Elevator were free, and the ailerons were linked to the Elevator. In the second case, the AoA and inboard aileron were free; Elevator and outboard aileron were linked to Inboard aileron. For each of these cases, a set of 20 AELINK cards were randomly generated from Latin Hypercube Sampling within the range [-5, 5]. The control surface deflections and corresponding hinge moments were then computed using SOL144, and the results for the first case are displayed in Fig. 7.1 for ascending values of the AoA.

Figure 7.1 shows that the Sum of the Absolute Values of the Hinge Moments increases as the control surface deflections increase. To determine the set of control surface deflections to minimize the Sum of the Absolute Values of the Hinge Moments, the optimization process applied to the HWB [11] and described in Sect. 6.3 was also applied

to the FSW. A total of 1000 samples were generated with the AELINK cards using Latin Hypercube Sampling within the range [-1.2 1.2]. Each sample includes the control surface

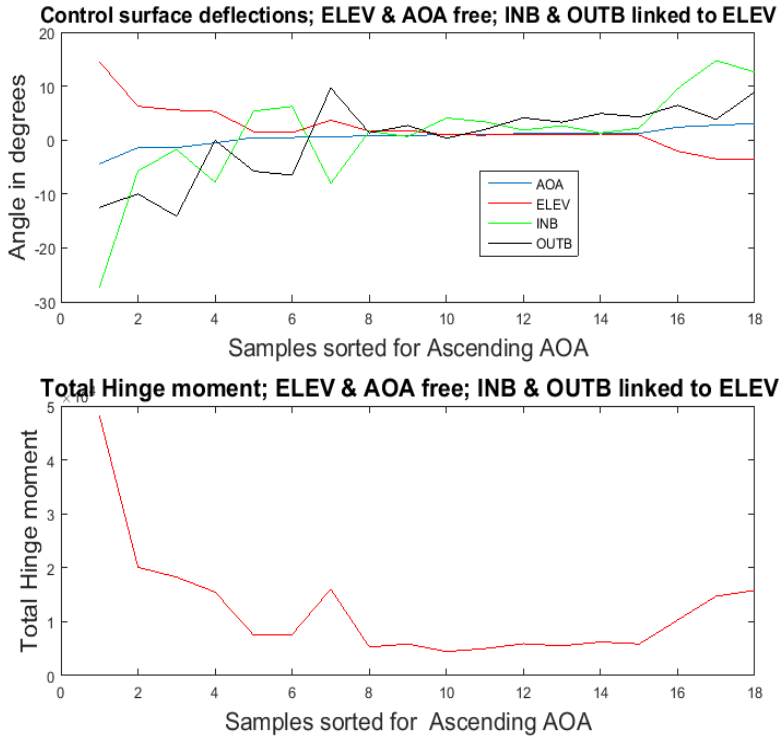

Figure 7.1: Control surface deflections and corresponding Hinge Moments for the first case on FSW aircraft: AoA and Elevator free

deflections (ELEV, INB, OUTB) & AoA, and the corresponding Sum of Absolute Values of Hinge Moments. A segregation was conducted, removing the highest 50% of Hinge Moments, and the remaining 696 samples were used to train the ANN as described in Sect. 6.2.2. Because there is no way to know *a priori* what number of neurons should be used for ANN training, a simple scheme was designed to determine this number. Only when a surrogate model with enough accuracy is designed can a reliable optimum be found with the GA. A common way to test a surrogate model is to evaluate its error at test points [45]. Here, the root-mean-square error defined in Eq. 25 and 26 is used.

$$\sigma = \sqrt{(\sum_{i=1}^{N} e(i)^2)/N}\,, \tag{25}$$

Where N is the number of test points, and

$$e(i) = \left| \frac{\text{SAHM\_nastran} - \text{SAHM\_estimated}}{\text{SAHM\_nastran}} \right| \quad (26)$$

The lower the root-mean-square error, σ, the more accurate the surrogate. The basic idea of the scheme is to train the ANN for different numbers of neurons (*n*) and choose the one with the minimum root-mean-square error, σ. There are two steps (see Fig. 7.2):

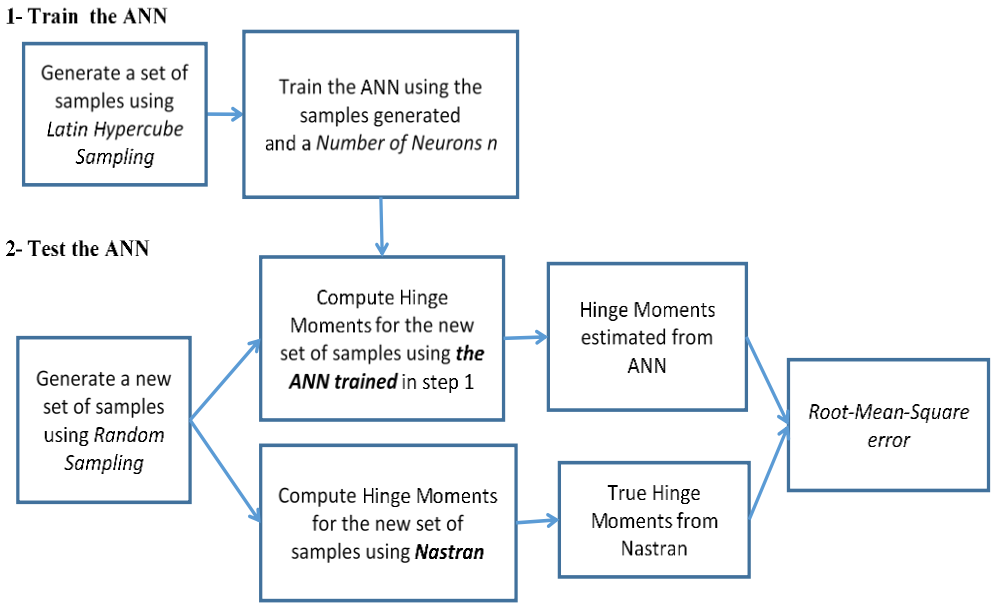

Figure 7.2: Scheme to determine the number of neurons in the Neural Network.

- Step 1: Train the ANN

The remaining samples (696) after segregation were used to train the ANN with *n* number of neurons. The input data for the ANN was the three control surface deflections for each of the 696 samples, and the target was their respective Sum of Absolute Values of Hinge Moments.

- Step 2: Test the ANN

A new set of 500 samples was generated for testing, and the Sum of Absolute Values of Hinge Moments for the new set of samples were computed using both the ANN trained in Step 1 and MSC Nastran. The estimated Sum of Absolute Values of Hinge Moments from the ANN and the true Sum of Absolute Values of Hinge Moments from MSC Nastran were then used to compute the root-mean-square error. These two steps were carried out for different numbers of neurons ranging from 5 to 200 with a step size of 5. The plot for the

errors as a function of number of neurons is shown in Fig. 7.3. The minimum root-mean-square error is obtained for 40 neurons. With this number of neurons, the training of the ANN is expected to be accurate. Figure 7.4 shows a comparison between the Sum of Absolute Values of Hinge Moments estimated from the trained ANN and the true Sum of Absolute Values of Hinge Moments from MSC Nastran. It can be seen that both values are in agreement.

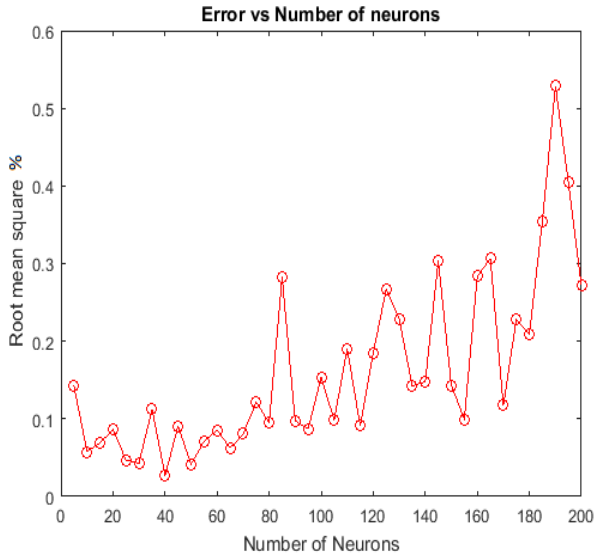

Figure 7.3: Root-mean-square error vs. Number of neurons.

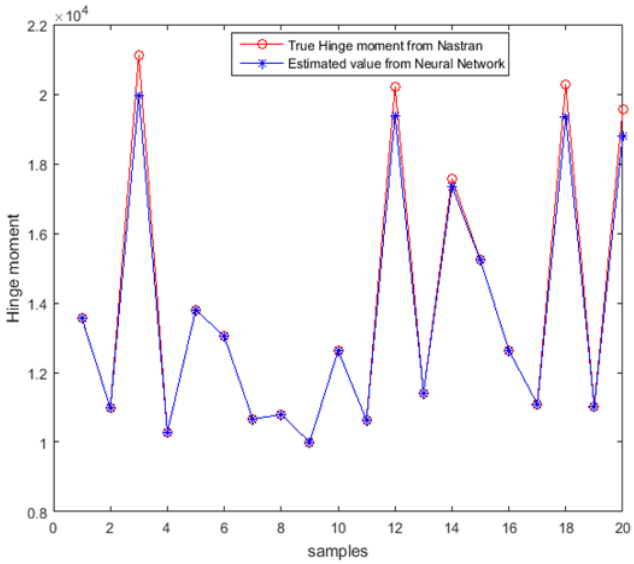

Figure 7.4: Estimated Sum of Absolute Values of Hinge Moments from ANN with 40 neurons vs. True Sum of Absolute Values of Hinge Moments from MSC Nastran.

With the accurate ANN, the optimization described below was carried out by the GA with:

$$Objective:\ minimize\ (|SAHM_{ELEV}| + |SAHM_{INB}| + |SAHM_{OUT}|)$$

$$Subject\ to\ LB \leq [ELEV,\ INB,\ OUB] \leq UB$$

The Sum of Absolute Values of Hinge Moments did not decrease over the course of the optimization as expected. Also, the Sum of Absolute Values of Hinge Moments from ANN+GA did not match the Sum of Absolute Values of Hinge Moments from MSC Nastran during the validation step. More surprisingly, the Sum of Absolute Values of Hinge Moments from ANN+GA were all negative. This shows that the optimization failed as the Sum of Absolute Values of Hinge Moments used to train ANN were all positive, and the expected optimum was also supposed to be positive. The ANN was well trained with the best number of neurons (the one with the minimum error). Thus, the reason of the optimization failure is not related to the ANN, but to the optimization problem itself. After a detailed investigation, the cause of the optimization failure was found to be due to two different reasons. The first is that the training of ANN was directly between control surface deflections and Sum of Absolute Values of Hinge Moments. When different sets of control deflections used to train the ANN were given as input to the trained ANN, the output Sum of Absolute Values of Hinge Moments were positive. However, when the opposites of the same sets of control deflections were given as input, the output Sum of Absolute Values of Hinge Moments were negative. The trained ANN was supposed to always output a positive value, as the ANN target was the Sum of Absolute Values of Hinge Moments. To fix this problem, it was decided to train the ANN between the control surface deflections and all the individual components of Hinge Moments before summing their absolute values (see Fig. 7.5).

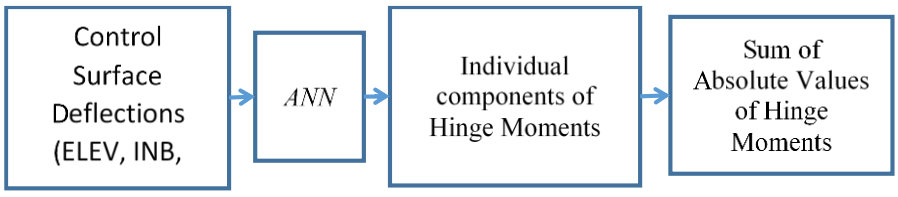

Figure 7.5: Training ANN between the control deflections and all the components of Hinge Moments.

With the new way of training the ANN, the optimization objective (Sum of Absolute Values of Hinge Moments) was always positive as expected. However, when the optimization was conducted again, the Sum of Absolute Values of Hinge Moments and the control surface deflections from the optimization did not match the ones from MSC Nastran during the validation step. The reason was tracked down; the cause turned out to be the lack of satisfying the trim constraints in the optimization process.To understand that, it is necessary to have a different view at the steps involved in the optimization process shown in Fig. 7.6.

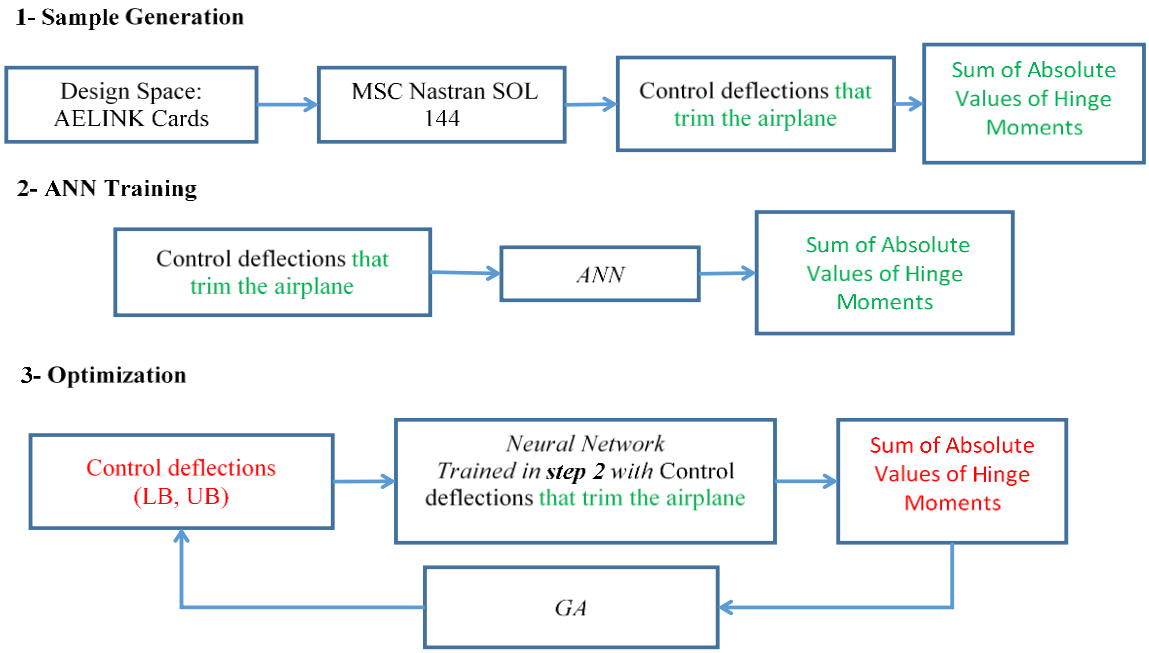

Figure 7.6: Initial Methodology with control deflections as design variables.

In Step 1, a set of AELINK cards is given as input to MSC Nastran SOL 144, which output the control surface deflections *that trim the airplane* and the corresponding Hinge Moments. In Step 2, the control surface deflections that *trim the airplane* and the corresponding Hinge Moments are used to train an ANN. In Step 3, the optimization is initialized. The control surface deflections are used as the optimization design variables with lower (LB= -30^o) and upper bounds (UB= 30^o) as constraints imposed on them. The ANN trained in Step 2 with the control surface deflections that *trim the airplane* is used to output the objective (Hinge Moments).

It was noticed that when an ANN is trained with a set of data in a certain range and with a specific feature, it can only be used for that kind of data. Because the ANN was trained with the control surface deflections that *trim the airplane,* the output Hinge Moments from that ANN would be reasonable if, and only, if the control surface deflections given as input *trim the airplane.* In this optimization process, there are only LB and UB constraints on control surface deflections and there is no way to constrain the GA to ensure that the optimum control surface deflections trim the airplane. So trim constraints must be added explicitly to the optimization process when control surface deflections are used as design variables so that the optimizer does not lead to a non-physical solution.

Another way to make the optimization effective is to use a different set of optimization variables which have no constraints, for example the so-called AELINK variables in MSC Nastran. They are generated randomly using Latin Hypercube Sampling within [-1.2, 1.2]. From the previous remarks on the ANN behavior, if AELINK cards are used as design variables, the output Hinge Moments will be reasonable as long as the same LB and UB constraints ([-1.2, 1.2]) are applied on the variables. There is no need to add additional constraints, since there are none on the AELINK cards, as opposed to control surface deflections which trim the airplane. When using AELINK cards as design variables, the ANN must be trained between AELINK cards and Hinge Moments. Figure 7.7 below shows both optimization processes.

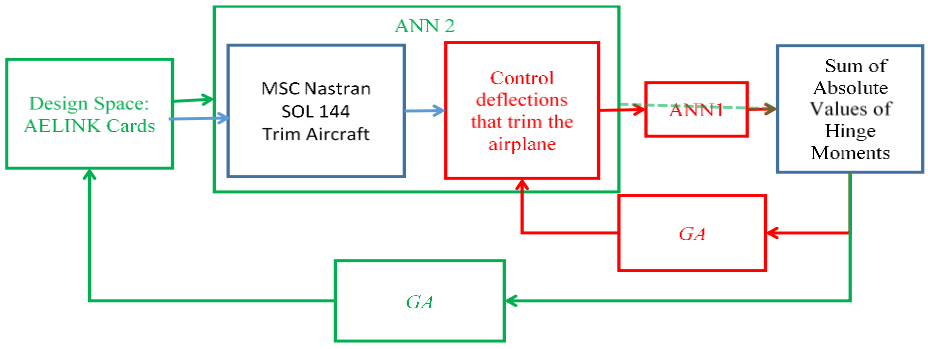

Figure 7.7: Optimization using AELINK cards (in green) vs. optimization using control surface deflections as design variables (in red).

In the previous optimization process, ANN1 was trained between control surface deflections and Hinge Moments, whereas in the new optimization ANN2 is trained using the AELINK cards as the input and Hinge Moments as the output. There is no constraint

on AELINK cards as the control surface deflections trim the airplane. Using this new process, the optimization was conducted, and the results after 20 iterations are displayed in Fig. 7.8 and Table 7.1.

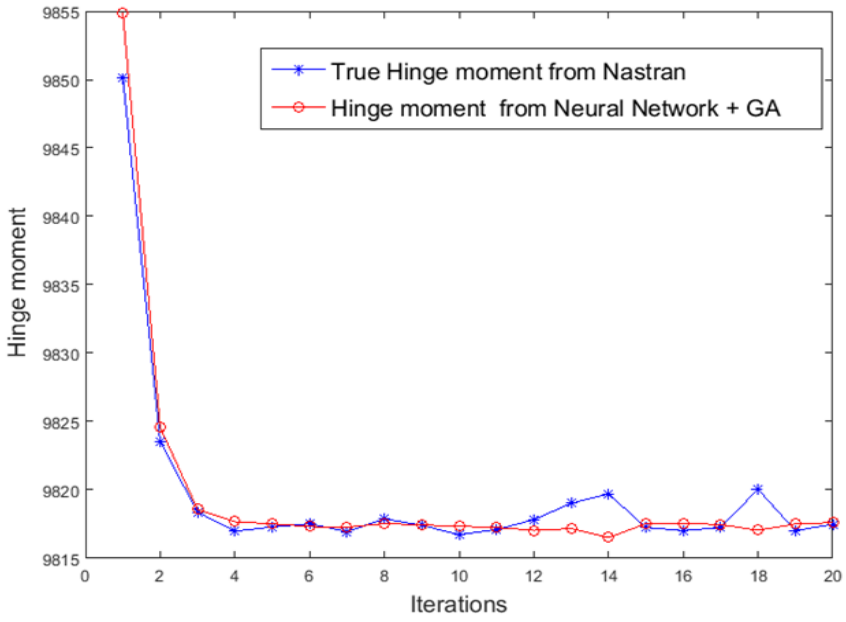

Figure 7.8: Optimization results for FSW using the AELINK cards

Table 7.1: Optimization results using AELINK cards.

	Sum of Absolute Values of Hinge Moments (lb. ft)
Minimum in the initial sample set	9931
Optimized Sum of Absolute Values of Hinge Moments (ANN+GA)	9817.4
MSC Nastran Validation	9816.7
%ERROR (between ANN and MSC Nastran)	7.13e-003%

Figure 7.8 shows that the Sum of Absolute Values of Hinge Moments estimated from the ANN is in agreement with the Sum of Absolute Values of Hinge Moments obtained from MSC Nastran. All the Sum of Absolute Values of Hinge Moments are positive, which was not the case when the control surface deflections were used as the design variables without imposing the trim constraints. This is one of the key findings of

this work. The minimum in the initial set of samples is 9931 lb. ft, and the optimized Hinge Moments from ANN+GA is 9816.7 lb. ft, which results in an improvement of 1.16%. Also, the error between ANN and MSC Nastran is 7.13e-003%, showing that the ANN is both accurate and well trained. The AELINK cards were generated within [-1.2, 1.2], and the optimization was conducted within the same limits. To try and improve the result, the design space was increased to [-5, 5]. To keep the same density in the design space, more samples (5000) were generated. After segregation, 3465 samples remained, and they were used to train the ANN. The optimization was performed again, and the results are displayed in Table 7.2.

Table 7.2: AELINK cards within [-1.2, 1.2] vs. AELINK cards within [-5 5].

	Aelink cards within [-1.2, 1.2]	*Aelink cards within* [-5, 5]
Minimum in the Initial Sample Set	9931	8530
Optimized Hinge Moment (ANN+GA)	9817.4	8510
MSC Nastran Validation	9816.7	8503
%Error (ANN and MSC Nastran)	7.13e-003%	0.03%
Improvement (final optimum vs. minimum in initial sample set)	1.16%	0.31%

When the range of samples is increased (i.e. AELINK cards within [-5 5] instead of [-1.2 1.2]) a better optimum, 8503 lb. ft, is found. The optimum for the previous design space (i.e. AELINK cards within [-1.2, 1.2]) was 9387 lb. ft. The Sum of Absolute Values of Hinge Moments decreased from 9817 lb. ft to 8503 lb. ft, an improvement of 9.41%. If the AELINK cards range were increased further to [-30, 30], and if ELEV is equal to 5^o, then INB will be equal to 150^o, which is not physically feasible. Consequently, the whole design space cannot be explored using AELINK cards. The control surface deflections, which unlike the AELINK cards are physical variables, have to be used as the design variables to cover the entire space, but the trim constraints must be added in the optimization problem.

To run MSC Nastran SOL 144 for a symmetric pull-up maneuver, one requires two free variables which correspond to the two unconstrained degrees of freedom, vertical

acceleration and pitch rate. As a result, two different constraints must be enforced in the optimization to trim the airplane. Since the load factor (2.5-G) in the vertical (z) direction and the pitch rate (0.0) are known, they can be used to enforce the trim constraints. The resulting extended optimization process is described in Fig. 7.9.

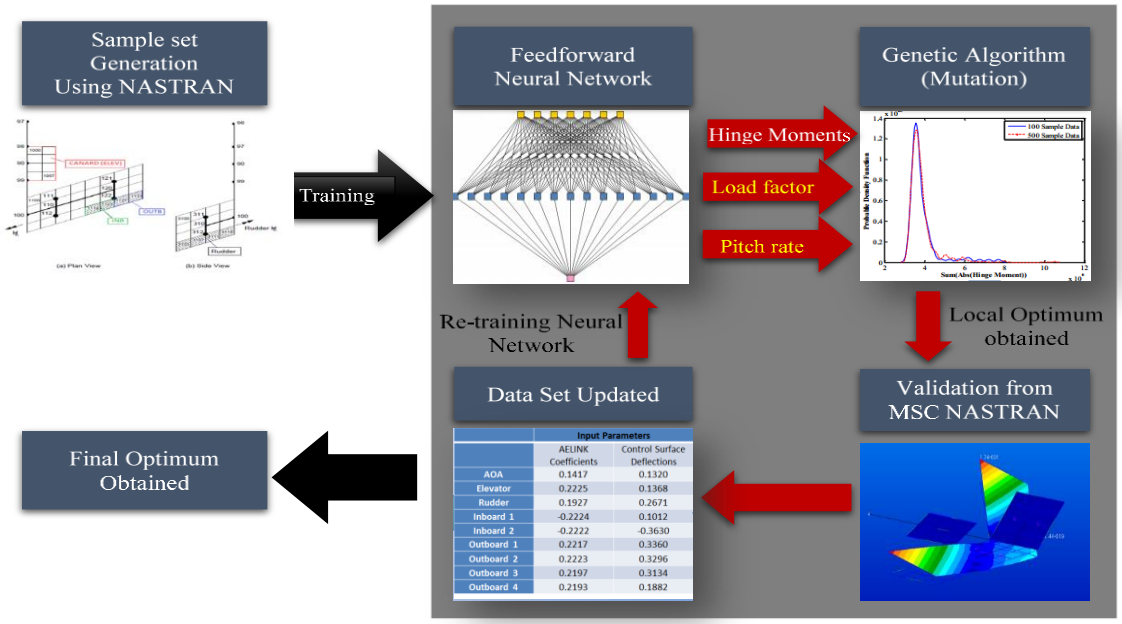

Figure 7.9: Extended optimization process including Load factor and Pitch rate constraints.

In Step 1, a set of control surface deflections and an AoA (generated with Latin Hypercube Sampling) is given as input to MSC Nastran SOL 144, which outputs the load factor, the pitch rate required to trim the airplane and the corresponding Hinge Moments. In Step 2, the control surface deflections and AoA are used as inputs to train the ANN to output all the components of Hinge Moments, the load factor and the pitch rate. In Step 3, the optimization is initialized. The control surface deflections and AoA are used as the optimization design variables with lower and upper bounds ([-30º, 30º]) as constraints. The ANN trained in Step 2 outputs not only all the components of Hinge Moments required to compute the objective function (Sum of Absolute Values of Hinge Moments), but also the load factor and pitch rate to enforce the trim constraints. The optimization displayed in Fig. 7.9 was conducted, and the results are shown in Fig. 7.10 and Table 7.3.

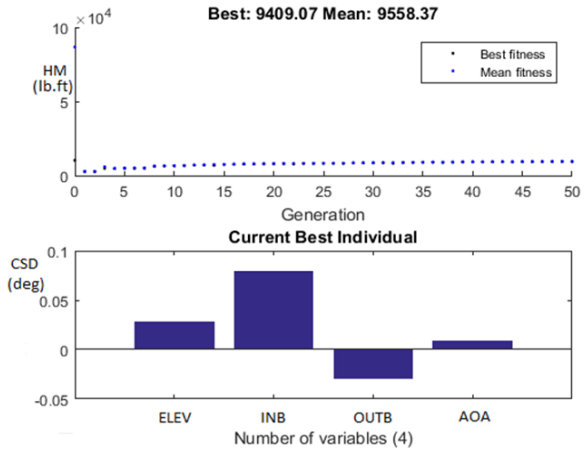

Figure 7.10: Optimization results using Load factor and Pitch rate constraints for FSW case.

Table 7.3: Optimization results using Load factor and Pitch rate constraints for FSW case.

DEFLECTIONS (deg.)	ANN+GA	Validation from MSC Nastran
AoA	0.54	0.54
Elevator	1.65	1.65
Inboard	4.56	4.56
Outboard	-1.69	-1.69
Load Factor (g)	2.50	2.50
Pitch rate	0	0
Sum of Absolute Values of Hinge Moments (lb. ft)	9566.00	9567.20

From Fig. 7.10, we see that the Sum of Absolute Values of Hinge Moments is converging as the number of iterations increases, meaning that an optimum is approached. Table 7.3 shows the optimum control surface deflections and their corresponding Sum of Absolute Values of Hinge Moments. The Sum of Absolute Values of Hinge Moments from ANN+GA is equal to the Sum of Absolute Values of Hinge Moments from MSC Nastran, and it converged. The load factor from ANN+GA is equal to the load factor from MSC Nastran and equal to 2.5-G. The pitch rate from ANN+GA is equal to the pitch rate from MSC Nastran and equal to 0.0. As a result, the optimum control surface deflections trim the airplane and are validated by MSC Nastran, showing that adding the trim constraints to the optimization process makes it robust.

Because the trim constraints were not applied in the HWB optimization from the previous work [11], it was decided to test the feasibility of the optimum control surface deflections obtained for the HWB full-span and half-span models using the improved formulation.

7.2 Hybrid Wing Body Case

To test the feasibility of the previous solutions, the optimum control surface deflections for both the HWB half- and full-span models (Table 7.4) [11] are given as input to MSC Nastran SOL 144, while leaving load factor and AoA as free variables.

Table 7.4: Optimum control deflections for both half- and full-span models of the HWB.

DEFLECTIONS (deg.)	Half-Span Model	Full-Span Model
AoA	8.03	8.00
Elevator	2.29	7.24
Inboard 1	-11.45	3.70
Inboard 2	30	15.78
Outboard 1	29.08	-18.37
Outboard 2	30.02	-20.36
Outboard 3	30.03	-16.66
Outboard 4	25.47	-17.06
Rudder	-10.98	-7.53
Sum of Absolute Values of Hinge Moments (lb. in)	1.20e+06	2.93e+06

For the above control surface deflections, the resulting load factors from MSC Nastran to trim the airplane are 2.44 and 0.46 for the half- and full-span models, respectively. The samples used to train ANN were generated for a 2.5-G case, and the optimum solution was supposed to trim the airplane for the same load case, but this was not the case. The half-span model is almost trimmed (2.44-G), and the full-span model is not (0.46-G). This shows that the trim constraints must be added in the optimization process even if the ANN is well trained.

The new optimization process including the trim constraints described on Fig. 7.9 was then applied to the HWB full-span model, and the results are displayed in Fig. 7.11 and Table 7.5.

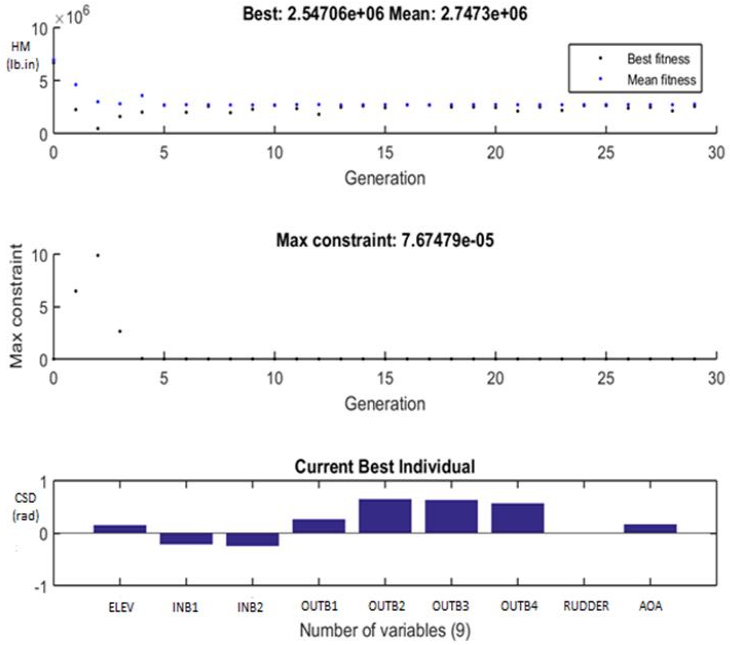

Figure 7.11: Optimization results using Load factor and Pitch rate constraints for the HWB Full model case.

Table 7.5: Optimization results using Load factor and Pitch rate constraints for the HWB Full model case.

DEFLECTIONS (deg.)	ANN+GA	Validation from Nastran
AoA	8.44	8.44
Elevator	8.34	8.34
Inboard 1	-11.99	-11.99
Inboard 2	-13.83	-13.83
Outboard 1	14.92	14.92
Outboard 2	37.16	37.16
Outboard 3	36.18	36.18
Outboard 4	32.65	32.65
Rudder	0.19	0.19
Load Factor	2.50	2.50
PITCH rate	0	0
Sum of Absolute Values of Hinge Moments (lb. in)	2.69e+06	2.69e+06

Table 7.5 shows the optimum control surface deflections and their corresponding Sum of Absolute Values of Hinge Moments. The Sum of Absolute Values of Hinge

Moments from ANN+GA is equal to the Sum of Absolute Values of Hinge Moments from MSC Nastran and is converged. The load factor from ANN+GA is equal to the load factor from MSC Nastran and equal to 2.5. The pitch rate from ANN+GA is also equal to the pitch rate from MSC Nastran and equal to 0.0. As a result, the optimum control surface deflections trim the airplane and are validated by MSC Nastran, showing that adding the trim constraints to the optimization process makes it robust.

7.3 FSW Case Optimization using Stability Derivatives

While working on the FSW case as described in Sect. 7.1, a new alternative method utilizing stability derivatives was developed to optimize Sum of Absolute Values of Hinge Moments. For a set of trim (steady) flight conditions (altitude, airspeed…), the control derivatives can be expected to be constant. Also, the aeroelastic performance of the aircraft is linearized about a given trim condition. The aeroelastic analysis MSC Nastran SOL 144 was run for different AELINK cards and the same flight conditions, and the following control derivatives (Table 7.6) were found to be constant for this airplane.

Table 7.6: Constant values of control derivatives.

Variables	AoA	n_z	ELEV	INB	OUTB	INBL	OUTBL
$\frac{\partial C_L}{\partial variable}$	-6.4629	0.0038	-0.5430	0.3074	0.3251	-0.3074	-0.3251
$\frac{\partial C_m}{\partial variable}$	-3.6671	0.0027	0.3860	0.2993	0.2447	-0.2993	-0.2447

The control derivatives being constant implies that they can be used to set up the trim equations that can be used in the optimization process to check whether a set of control surface deflections trim the airplane. There are two trim constraint equations:

1) Forces in the vertical direction:

$$\sum \boldsymbol{F_z} = \boldsymbol{n_z} * \text{Weight} \left(\frac{\partial C_L}{\partial \alpha}\alpha + \frac{\partial C_L}{\partial ELEV}ELEV + \frac{\partial C_L}{\partial INB}INB + \frac{\partial C_L}{\partial OUTB}OUTB + \frac{\partial C_L}{\partial INBL}INBL + \frac{\partial C_L}{\partial OUTBL}\text{OUTBL} + \frac{\partial C_L}{\partial n_z}n_z - \frac{L_o}{q\,S} + accelerations\ derivatives\right) = 0 \quad (27)$$

2) Moment about the y-axis:

$$\frac{\partial C_m}{\partial \alpha}\alpha + \frac{\partial C_m}{\partial ELEV} ELEV + \frac{\partial C_m}{\partial INB} INB + \frac{\partial C_m}{\partial OUTB} OUTB + \frac{\partial C_m}{\partial INBL} INBL + \frac{\partial C_m}{\partial OUTBL} OUTBL + \frac{\partial C_m}{\partial n_z} n_z - \frac{M_o}{q\,Sc} + accelerations\ derivatives = 0 \quad (28)$$

Here, the accelerations (pitch, roll.) derivatives are all equal to 0. nz is the load factor and equals 2.5; the dynamic pressure q equals 1200 psf, the wing area S equals 400 sq. ft and the chord c equals 10 ft; α is the angle of attack; $L_o = n_z * weight$ and M_o are the lift and moment that are not dependent on AoA and control surface deflections.

Like the control derivatives, the hinge moment derivatives were also constant (Table 7.7).

Table 7.7: Constant values of control derivatives.

Variables	AoA	n_z	ELEV	INB	OUTB	INBL	OUTBL
$\frac{\partial ELEV}{\partial Variable}$	-7.2641	0.1249	227.7316	6.4863	8.7517	-6.4863	-8.7517
$\frac{\partial INB}{\partial Variable}$	12.5821	-0.0089	-0.3043	-22.3137	-3.5665	0.2024	0.2323
$\frac{\partial OUTB}{\partial Variable}$	11.0639	-0.0065	0.6696	-3.3739	-21.6621	0.1066	0.1609
$\frac{\partial INBL}{\partial Variable}$	-12.5821	0.0089	0.3043	0.2024	0.2323	-22.3137	-3.5665
$\frac{\partial OUTBL}{\partial Variable}$	-11.0639	0.0065	-0.6696	0.1066	0.1609	-3.3739	-21.6621

The constant hinge moment derivatives can be used to compute the Hinge Moments, just as the lift coefficients are used to compute the lift, or the moment coefficients are used to compute the moment. Knowing the hinge moment derivatives, the Hinge Moments are computed as:

$$HM_x = q(\frac{\partial HM_x}{\partial \alpha}\alpha + \frac{\partial HM_x}{\partial ELEV} ELEV + \frac{\partial HM_x}{\partial INB} INB + \frac{\partial HM_x}{\partial OUTB} OUTB + \frac{\partial HM_x}{\partial INBL} INBL + \frac{\partial HM_x}{\partial OUTBL} OUTBL + \frac{\partial HM_x}{\partial z} n_z) \quad (29)$$

where x is the Control Surface Deflection

The updated optimization problem using this information is summarized in Fig. 7.12.

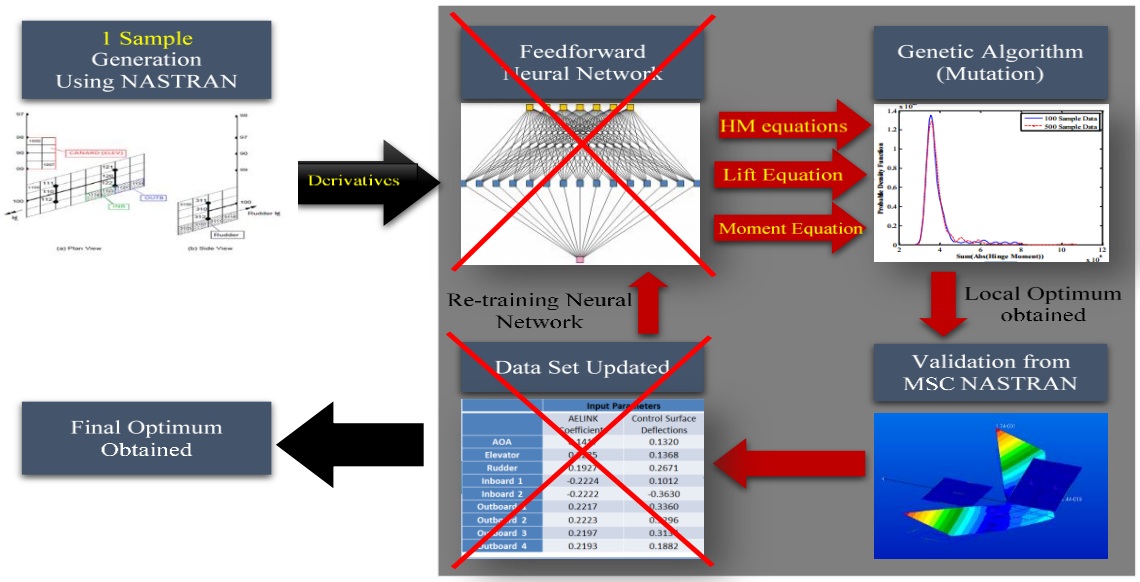

Figure 7.12: Optimization approach using stability derivatives.

The first step of the new process is to run MSC Nastran SOL 144 once to extract the control derivatives and hinge moment derivatives. Those are used in Step 2 to set up the Hinge Moments and trim equations to conduct the optimization. This takes away the need to train an ANN. Using the new process, the optimization was conducted and the results are displayed in Fig. 7.13 and Table 7.8 below.

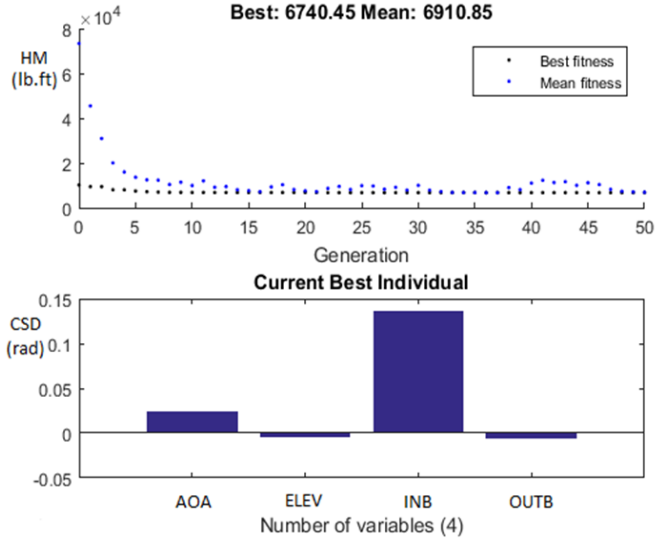

Figure 7.13: FSW case optimization results using stability derivatives.

Table 7.8: FSW case optimization results using stability derivatives.

DEFLECTIONS (deg.)	Optimization with stability derivatives	Validation from Nastran	ANN+GA
AoA	1.40	1.40	0.54
Elevator	-0.30	-0.30	1.65
Inboard	7.85	7.85	4.56
Outboard	-0.31	-0.31	-1.69
Load Factor	2.50	2.50	2.50
PITCH rate	0	0	0
Sum of Absolute Values of Hinge Moments (lb. in)	6740.45	6740.45	9566.00

Fig. 7.13 shows that the Sum of Absolute Values of Hinge Moments is converging as the number of iterations increases, meaning that an optimum is approached. Table 7.8 shows the optimum control surface deflections and their corresponding Sum of Absolute Values of Hinge Moments. The Sum of Absolute Values of Hinge Moments from the optimization with the stability derivatives is equal to the Sum of Absolute Values of Hinge Moments from MSC Nastran. The load factor from the optimization with the stability derivatives is equal to the load factor from MSC Nastran and equal to 2.5. The pitch rate is also equal to the pitch rate from MSC Nastran and equal to 0.0. The third column of Table 7.8 shows the earlier results obtained from ANN+GA optimization and presented in Table 7.3. The optimum Sum of Absolute Values of Hinge Moments from the optimization with the stability derivatives is much better than that from ANN+GA.

Another means of comparison between both optimization processes is the computational time. Tables 7.9 and 7.10 show the computational time for both methods.

Table 7.9: Computational time with optimization process using ANN+GA.

Operations	Computational Time
Generate samples	More than 24h
Train Neural Network, determine the best number of neurons	10h
Optimization, validation, retrain Neural Network at each iteration	5-10 min /iteration 8h-16
Total	**2-3 days**

Table 7.10: Computational time with optimization process using stability derivatives.

Operations	Computational Time
Generate samples	1 min
Train Neural Network, determine the best number of neurons	1 min
Optimization, validation, retrain Neural Network at each iteration	5-10 min
Total	**15 min**

From the Tables above, it can be seen that using the stability derivatives to optimize Sum of Absolute Values of Hinge Moments reduces the computational time from 2-3 days to less than 15 min.

7.4 HWB Case Optimization using Stability Derivatives

Next, the updated optimization process with the stability derivatives was applied to the HWB. The control derivatives as well as the hinge moment derivatives were again found to be constant for the HWB, implying that the optimization process described in Fig. 7.12 can also be applied here. The results are presented in Fig. 7.14 and Table 7.11.

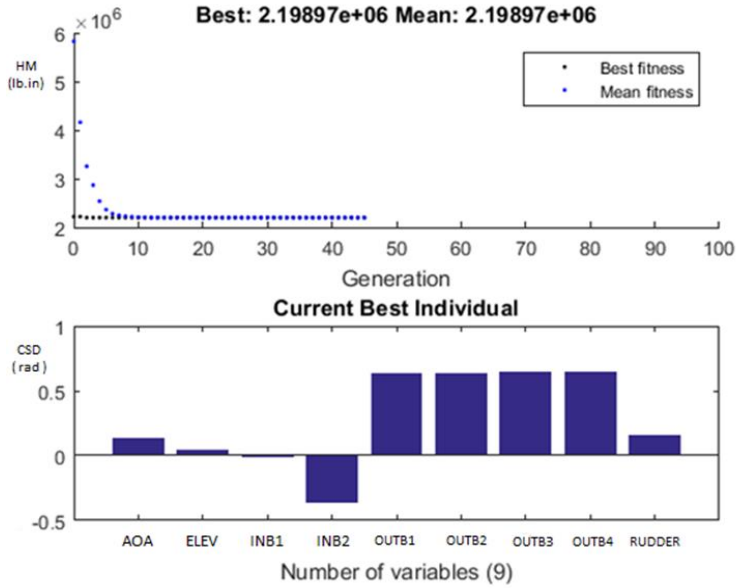

Figure 7.14: HWB case optimization results using stability derivatives.

Table 7.11: HWB case optimization results using stability derivatives.

DEFLECTIONS (deg.)	Optimization with stability derivatives	Validation from Nastran
AoA	7.95	7.95
Elevator	2.32	2.32
Inboard 1	-0.54	-0.54
Inboard 2	-20.76	-20.76
Outboard 1	36.33	36.33
Outboard 2	36.55	36.55
Outboard 3	37.42	37.42
Outboard 4	36.79	36.79
Rudder	9.41	9.41
Load Factor	2.50	2.49
PITCH rate	0	0
Sum of Absolute Values of Hinge Moments (lb. in)	2.20e+06	2.19e+06

From Fig. 7.14 the Sum of Absolute Values of Hinge Moments is converging as the number of iterations increases, meaning that an optimum is approached. Table 7.11 shows the optimum control surface deflections and their corresponding Sum of Absolute Values of Hinge Moments. The Sum of Absolute Values of Hinge Moments from the optimization with the stability derivatives is equal to the Sum of Absolute Values of Hinge Moments from MSC Nastran, and it converged. The load factor from the optimization with the stability derivatives is equal to the load factor from MSC Nastran and equal to 2.5. The pitch rate is also equal to the pitch rate from MSC Nastran and equal to 0.0. The computational time was reduced from 4-5 days using ANN+GA to less than an hour using the stability derivatives.

7.5 HWB Case: Optimization with Stability derivatives vs. ANN+GA vs. SOL 200

A different process was developed by Jesse R. Quinlan at NASA Langley Research Center to conduct the same optimization on the HWB full-span model [50]. In this work, a conventional gradient-based optimization method was used to provide a means of

comparison for the ANN+GA optimization. To this end, the multidisciplinary optimizer MSC Nastran SOL 200 was used. Table 7.12 shows a comparison of the results between optimization with the ANN+GA, the stability derivatives and Quinlan's results using SOL 200.

Table 7.12: HWB: Optimization with Stability derivatives vs. Neural Network vs. SOL 200.

DEFLECTIONS (deg.)	ANN+GA	Stability derivatives	SOL 200
AoA	8.44	7.95	7.73
Elevator	8.34	2.32	9.17
Inboard 1	-11.99	-0.54	-34.26
Inboard 2	-13.83	-20.76	-17.73
Outboard 1	14.92	36.33	13.25
Outboard 2	37.16	36.55	10.34
Outboard 3	36.18	37.42	28.78
Outboard 4	32.65	36.79	-23.71
Rudder	0.19	9.41	-25.39
Sum of Absolute Values of Hinge Moments (lb. in)	2.69e+06	2.20e+06	2.67e+06
Computational Time	5 days	1 hour	5 days

The above table shows that the method using stability derivatives gives a better minimized Sum of Absolute Values of Hinge Moments and computational time than SOL 200, the latter having similar results with ANN+GA.

All the work performed so far is related to a specific flight condition (speed, altitude, load factor...). To take advantage of both the computational efficiency of the method using stability derivatives and the ability of an Artificial Neural Network to estimate functions with a large number of inputs, the next section presents a hybrid scheme combining both methods for the control allocation optimization of the HWB at variable flight conditions in pitching motion.

7.6 Hinge Moment Optimization for HWB for a Pull-up Maneuver at Various Flight Conditions

To determine the optimal Sum of Absolute Values of Hinge Moment and control surface deflections for various flight conditions (speed, altitude, load factor...) the stability derivatives method and ANN+GA method were combined together. This will allow for the real-time, efficient deployment of the control surfaces to trim the airplane with the minimum power consumption in the event of a sudden change in the flight conditions due to turbulence or a decision of the pilot to fly at a different altitude, speed or load factor. The steps involved in this scheme are described in Fig. 7.15.

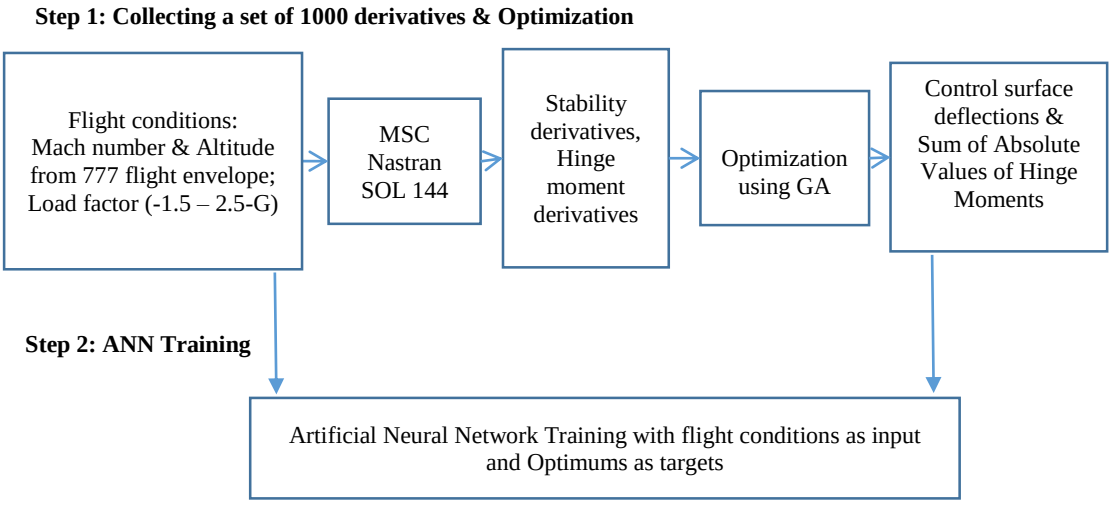

Figure 7.15: Hinge moment optimization for variable flight conditions in a Pull-up maneuver.

The first step of the scheme in Fig. 7.15 is to construct a database of flight conditions (Mach, altitude, load factor) using Latin Hypercube Sampling. The load factor was generated within [-1.0, 2.5]. The Mach number and altitude were generated within the flight envelope of a long-range mission of the Boeing 777-200 LR shown in Fig. 7.16, which was adopted for the HWB for this study. The bold line in the middle is the operational Mach number at various altitudes, and the two lines on either side represent the lower and upper limits at which the control effectiveness is to be calculated [46].

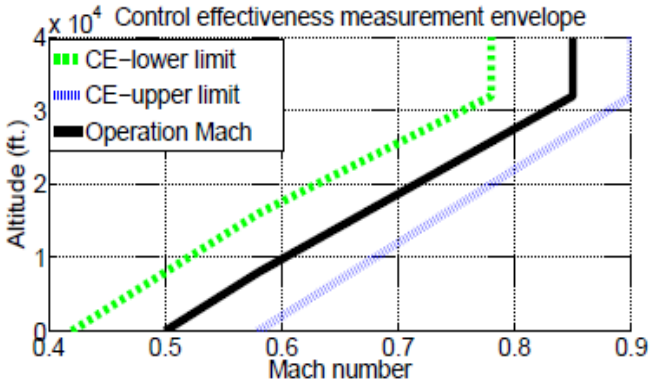

Figure 7.16: Flight envelope of long range mission of 777-200 LR.

The database was first populated with 1000 sets of flight conditions within the range described above using Latin Hypercube Sampling. Then, MSC Nastran SOL 144 was run to extract the set of control derivatives and hinge moment derivatives for each of the flight conditions. Those are used to set up the Hinge Moments and trim equations to conduct the optimization of the aircraft for each set of flight conditions using the Genetic Algorithm. This results in 1000 sets of optimized control deflections and Sum of Absolute Values of Hinge Moments corresponding to the 1000 sets of flight conditions. In the second step of the scheme, two ANNs were trained using the sets of flight conditions as input and the optimized control deflections and Sum of Absolute Values of Hinge Moments as targets. The trained ANNs can thus serve as response surfaces to estimate the real-time set of optimum control surface schedules while minimizing the power consumption at different sets of flight conditions for a pull-up maneuver.

A first ANN with flight conditions (Mach, altitude, load factor) as inputs and Sum of Absolute Values of Hinge Moments as the target was trained. Different number of neurons between 50 -75 were tried with 95% samples for training and 5% for testing the network. Figure 7.17 below shows the training results of the best network with 60 neurons.

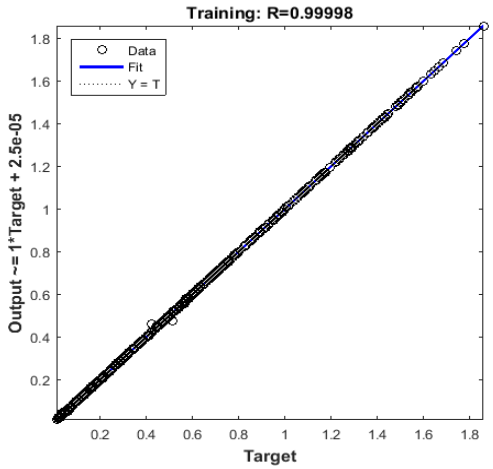

Figure 7.17: Training regression plot for 60 neurons for Sum of Absolute Values of Hinge Moments Training.

A second ANN with flight conditions (Mach, altitude, load factor) as inputs and control surface deflections as the targets was also trained. Different numbers of neurons between 50-200 were tried with 95% samples for training and 5% for testing the network. Figure 7.18 below shows the training results of the best network with 150 neurons. The reason for the higher number of neurons with respect to the first ANN is that the size of the network had to increase to account for the increased number of targets: 8 control surface deflections and Angle of Attack (AoA) for the second ANN as opposed to only the Sum of Absolute Values of Hinge Moments for the first ANN.

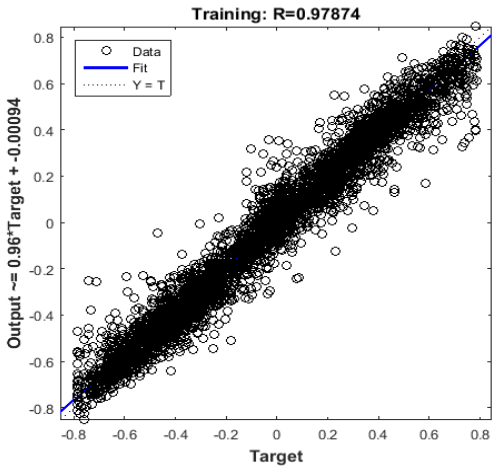

Figure 7.18: Training regression plot for 150 neurons for control surface deflections training.

To test the Neural Networks trained, random sets of flight conditions within the range of training were given as input to the ANNs, which output the corresponding control surface deflections and Sum of Absolute Values of Hinge Moments. These estimated results were validated through comparison with the optimized results from the GA. The results are presented in Table 7.13 for Mach = 0.65, Altitude = 18300 ft. and Load factor = 1.66.

Table 7.13: Estimated results from the trained Neural Networks and validation with Genetic Algorithm for Mach = 0.65, Altitude = 18300 and Load factor = 1.66.

DEFLECTIONS (deg.)	Estimation from ANN	Validation from GA	Relative Errors (%)
AoA	21.77	26.93	19.15
Elevator	2.86	4.01	28.57
Inboard 1	12.61	16.62	24.14
Inboard 2	-30.37	-37.24	18.45
Outboard 1	14.32	16.62	13.80
Outboard 2	24.64	36.10	31.75
Outboard 3	23.50	15.47	51.85
Outboard 4	9.74	12.61	22.72
Rudder	15.47	19.48	20.59
Sum of Absolute Values of Hinge Moments (lb. in)	9.92E+05	9.93E+05	0.10

Table 7.13 above shows the same trend of values between the estimated results from the trained Neural Networks and the validated ones from the GA. However, the errors between those values are relatively high for the control surface deflections (20-30%) and low for the Sum of Absolute Values of Hinge Moments (0.1%). This shows that ANN training for Sum of Absolute Values of Hinge Moments is almost perfect as evidenced by the training regression plot on Fig. 7.17 above. This is not the case for the ANN training for control surface deflections as displayed on Fig. 7.18. The ANNs training could, however, be further improved using the techniques listed in the Sect. 3.3.4.3 (Improve Neural Network Training and Avoid Overfitting).

7.7 Control Surface Deflection Schedule Optimization for HWB for a Steady Roll Maneuver using Stability Derivatives

For a steady roll maneuver, the aeroelastic analysis MSC Nastran SOL 144 was run for different AELINK cards and the same flight conditions (Mach number, altitude, roll rate) and the hinge moment and control derivatives were again found to be constant. The hinge moment derivatives being constant implies that they can be used to compute the hinge moments using the following equation:

$$HM_x = q(\frac{\partial HM_x}{\partial p}p + \frac{\partial HM_x}{\partial ELEV}ELEV + \sum\frac{\partial HM_x}{\partial INB}INB_{1,2} + \sum\frac{\partial HM_x}{\partial OUTB}OUTB_{1,2,3,4} + \frac{\partial HM_x}{\partial RUDDER}RUDDER + \frac{\partial HM_x}{\partial ELEVL}ELEVL + \sum\frac{\partial HM_x}{\partial INBL}INBL_{1,2} + \sum\frac{\partial HM_x}{\partial OUTBL}OUTBL_{1,2,3,4} + \frac{\partial HM_x}{\partial RUDDERL}RUDDERL) \quad (30)$$

where x is the control surface deflection, p is the roll rate, ELEV is the right elevator and ELEVL is the left elevator

For the same flight conditions (Mach=0.5, dynamic pressure= 1.769 psi and roll rate = 0.26 rad/s = 15 deg/s), two different sets of AELINK cards were used to test the accuracy of the results provided by Eq. 30. The first set of AELINK cards was chosen so that the right control surfaces can deflect independently of the left ones. The aircraft was then trimmed, and Table 7.14 below shows the control surface deflections and corresponding hinge moments. From Table 7.14, it can be seen that the deflections of the right control surfaces are different from those of the left control surfaces. The hinge moments corresponding to the control surfaces and calculated from MSC Nastran were also displayed. Those hinge moments were also computed using Eq. 30. Table 7.15 displays the hinge moments computed from both MSC Nastran and Eq. 30, and it shows that the hinge moments from the equation are equal to the hinge moments from MSC Nastran.

Table 7.14: Control surface deflections and corresponding hinge moments for the first set of AELINK cards.

Right Wing Control Surfaces	Deflections (deg)	Hinge Moment (E+06 lb.in)	**Left** Wing Control Surfaces	Deflections (deg)	Hinge Moment (E+06 lb.in)
Elevator	-13.94	1.025	Elevator	17.43	-0.800
Inboard 1	7.15	-0.349	Inboard 1	-21.07	0.143
Inboard 2	29.89	-0.677	Inboard 2	16.45	-0.264
Outboard 1	21.85	-0.180	Outboard 1	-49.32	0.226
Outboard 2	-12.99	0.003	Outboard 2	-53.15	0.169
Outboard 3	46.64	-0.172	Outboard 3	32.08	-0.138
Outboard 4	-26.16	0.079	Outboard 4	-32.85	0.071
Rudder	-2.29	0.012	Rudder	3.07	-0.023
Sum of Absolute Values of Hinge Moments = 4.331E+06 lb. in					

Table 7.15: Hinge moments from MSC Nastran vs. Hinge moments from Eq. 30.

Right Wing Control Surfaces	Hinge Moment from Nastran (E+06 lb.in)	Hinge Moment from Equation 30 (E+06 lb.in)	**Left** Wing Control Surfaces	Hinge Moment from Nastran (E+06 lb.in)	Hinge Moment from Equation 30 (E+06 lb.in)
Elevator	1.025	1.025	Elevator	-0.800	-0.800
Inboard 1	-0.349	-0.349	Inboard 1	0.143	0.143
Inboard 2	-0.677	-0.677	Inboard 2	-0.264	-0.264
Outboard 1	-0.180	-0.180	Outboard 1	0.226	0.226
Outboard 2	0.003	0.003	Outboard 2	0.169	0.169
Outboard 3	-0.172	-0.172	Outboard 3	-0.138	-0.138
Outboard 4	0.079	0.079	Outboard 4	0.071	0.071
Rudder	0.012	0.012	Rudder	-0.023	-0.023
Sum of Absolute Values of Hinge Moments = 4.331E+06 lb. in					

The second set of AELINK cards was chosen so that the aircraft can achieve an antisymmetric, steady roll maneuver. The hinge moment derivatives for this set were equal to those of the first set, because the flight conditions remained the same. Tables 7.16 and 7.17 show the control surface deflections to trim the aircraft and the corresponding hinge moments computed from both MSC Nastran and Eq. 30.

Table 7.16 shows that the control surface deflections of the right and left wing are antisymmetric to one another. Table 7.17 shows that the hinge moments from Eq. 30 are equal to the hinge moments from MSC Nastran, proving the accuracy of Eq. 30 to estimate the hinge moment values.

Table 7.16: Control surface deflections and corresponding hinge moments for the second set of AELINK cards.

Right Wing Control Surfaces	Deflections (deg)	Hinge Moment (E+06 lb.in)	**Left** Wing Control Surfaces	Deflections (deg)	Hinge Moment (E+06 lb.in)
Elevator	-20.93	1.528	Elevator	20.93	-0.909
Inboard 1	10.69	-0.368	Inboard 1	- 10.69	0.213
Inboard 2	24.69	-0.651	Inboard 2	- 24.69	0.369
Outboard 1	53.10	-0.324	Outboard 1	- 53.10	0.223
Outboard 2	-58.85	0.186	Outboard 2	58.85	-0.255
Outboard 3	48.16	-0.168	Outboard 3	- 48.16	0.111
Outboard 4	28.39	-0.077	Outboard 4	- 28.39	0.070
Rudder	2.99	-0.046	Rudder	- 2.99	0.046
Sum of Absolute Values of Hinge Moments = 5.545E+06 lb. in					

Table 7.17: Hinge moments from MSC Nastran vs. Hinge moments from Eq. 30.

Right Wing Control Surfaces	Hinge Moment from Nastran (lb.in)	Hinge Moment from Equation (lb.in)	**Left** Wing Control Surfaces	Hinge Moment from Nastran (lb.in)	Hinge Moment from Equation (lb.in)
Elevator	1.528	1.528	Elevator	-0.909	-0.909
Inboard 1	-0.368	-0.368	Inboard 1	0.213	0.213
Inboard 2	-0.651	-0.651	Inboard 2	0.369	0.369
Outboard 1	-0.324	-0.324	Outboard 1	0.223	0.223
Outboard 2	0.186	0.186	Outboard 2	-0.255	-0.255
Outboard 3	-0.168	-0.168	Outboard 3	0.111	0.111
Outboard 4	-0.077	-0.077	Outboard 4	0.070	0.070
Rudder	-0.046	-0.046	Rudder	0.046	0.046
Sum of Absolute Values of Hinge Moments = 5.545E+06 lb. in					

For simplicity, the aircraft will be constrained to achieve an antisymmetric roll maneuver.

The control derivatives being constant implies that they can be used to set up the lateral-directional trim equations that can be used in the optimization process to constrain the

control surface deflections to trim the aircraft. The lateral-directional trim equations are the following:

Side Force:

$$C_Y = C_{Y_\beta}\beta + \sum C_{Y_x}x + C_{Y_p}\frac{p\,b}{2\,V} + \ accelerations\ terms \quad (31)$$

Rolling Moment:

$$C_l = C_{l_\beta}\beta + \sum C_{l_x}x + C_{l_p}\frac{p\,b}{2\,V} + \ accelerations\ terms \quad (32)$$

Yawing Moment:

$$C_n = C_{n_\beta} + \sum C_{n_x}x + C_{n_p}\frac{p\,b}{2\,V} + \ accelerations\ terms \quad (33)$$

Here the accelerations (roll, yaw.) terms are all equal to 0 because of the steady maneuver. β is the sideslip angle; $\frac{p\,b}{2\,V}$ is the roll rate and x represents the control surface deflections.

To determine the control surface deflections to minimize the Sum of Absolute Values of Hinge Moments the optimization described below was carried out by a GA:

Objective: min ($\sum|HM_x|$)

Subject to the lateral-directional trim equations described above,

And $LB \leq x \leq UB$

Where x are the control surface deflections, LB=-30◦ and UB=30◦

The optimization was conducted for the same flight conditions: Mach=0.5, dynamic pressure= 1.769 psi and roll rate = 0.26 rad/s = 15 deg/s, and the results are presented in Fig. 7.19 and Table 7.18 below.

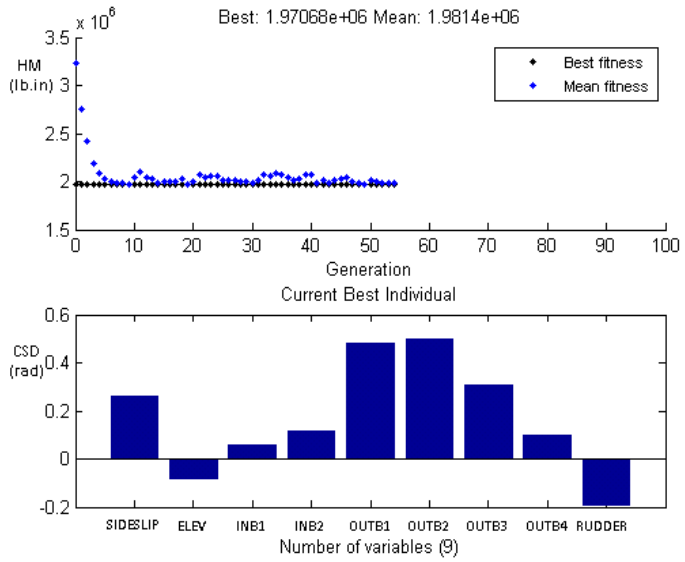

Figure 7.19: HWB optimization in a steady roll maneuver.

Table 7.18: HWB optimization in a steady roll maneuver.

DEFLECTIONS (deg.)	Optimization with stability derivatives	Validation from Nastran
Sideslip	14.95	14.95
Elevator	-5.05	-5.05
Inboard 1	3.25	3.25
Inboard 2	6.81	6.81
Outboard 1	27.62	27.62
Outboard 2	28.52	28.52
Outboard 3	17.52	17.52
Outboard 4	5.61	5.61
Rudder	-11.43	-11.43
Roll rate	15.00	15.00
Sum of Absolute Values of Hinge Moments (lb. in)	1.97E+06	1.97E+06

Fig. 7.19 shows that the Sum of Absolute Values of Hinge Moments is converging as the number of iterations increases, meaning that an optimum is approached. Table 7.18 shows the optimum control surface deflections and their corresponding Sum of Absolute Values of Hinge Moments. The Sum of Absolute Values of Hinge Moments from the optimization with the stability derivatives is equal to the Sum of Absolute Values of Hinge

Moments from MSC Nastran. The roll rate, sideslip angle and control surface deflections from the optimization are also equal to the roll rate, sideslip angle and control surface deflections from MSC Nastran. As a result the optimum solution from the optimization is a valid and feasible solution.

7.8 Hinge Moment Optimization for Various Flight Conditions in a Roll Maneuver

The approach applied for a pull-up maneuver as displayed in Fig. 7.15 was extended to a roll maneuver case to determine the optimal Sum of Absolute Values of Hinge Moment and control surface deflections for various flight conditions (speed, altitude, *roll rate*...). The steps involved are displayed in Fig. 7.20 below.

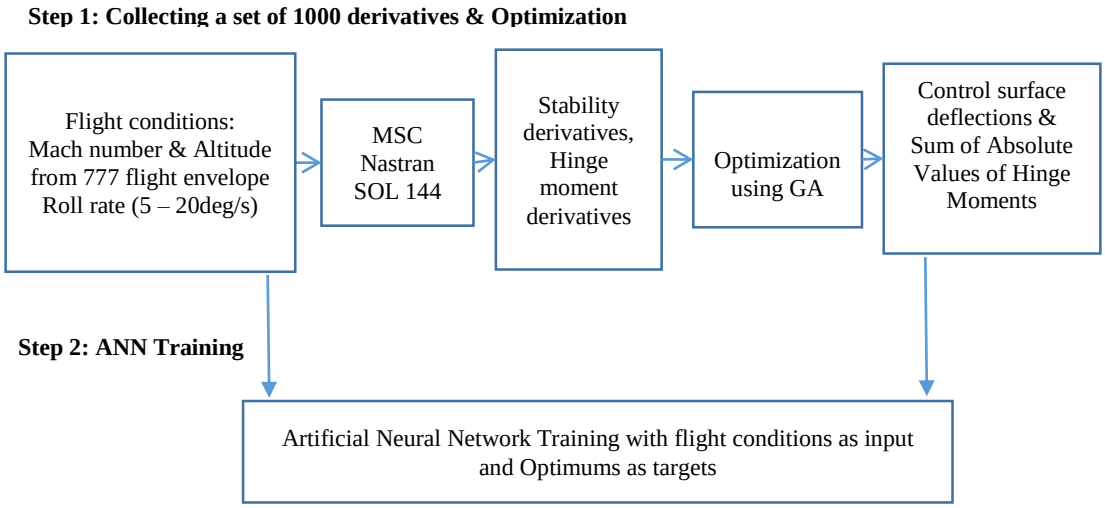

Figure 7.20: Hinge moment optimization for variable flight conditions in a roll maneuver.

A first ANN with flight conditions (Mach, altitude, roll rate) as inputs and Sum of Absolute Values of Hinge Moments as the target was trained. Different number of neurons were tried, and Fig. 7.21 below shows the training results of the network with 75 neurons.

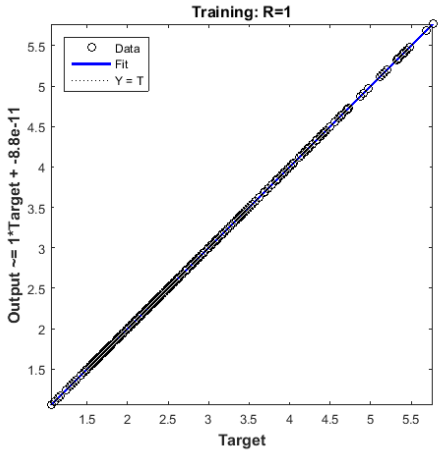

Figure 7.21: Training regression plot for 75 neurons for Sum of Absolute Values of Hinge Moments Training in a roll maneuver.

A second ANN with flight conditions (Mach, altitude, roll rate) as inputs and control surface deflections as the targets was also trained. Different number of neurons were tried, and Fig. 7.22 below shows the training results of the network with 300 neurons. The reason for the higher number of neurons with respect to the first ANN is that the size of the network had to increase to account for the increased number of targets: 8 control surface deflections + Sideslip for the second ANN as opposed to only the Sum of Absolute Values of Hinge Moments for the first ANN.

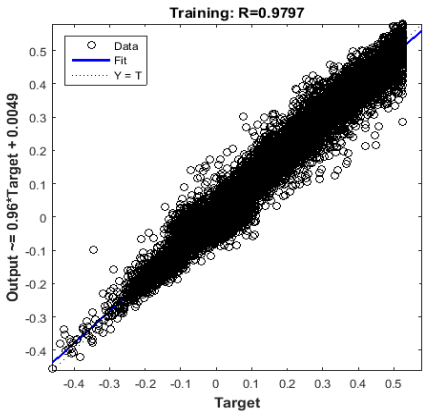

Figure 7.22: Training regression plot for 300 neurons for control surface deflections training in a roll maneuver

To test the Neural Networks trained, a few sets of flight conditions were given as input to the ANNs, which output the corresponding control surface deflections and Sum of

Absolute Values of Hinge Moments. These estimated results were validated through comparison with the optimized results from the GA. The results are presented in Table 7.19 for Mach = 0.78, Altitude = 30200 ft. and Roll Rate = 20 deg/s.

Table 7.19: Estimated results from the trained Neural Networks and validation with Genetic Algorithm for Mach = 0.78, Altitude = 30200 ft. and Roll Rate = 20 deg/s.

DEFLECTIONS (deg.)	Estimation from ANN	Validation from GA	Relative Errors (%)
Sideslip	5.87	6.03	2.65
Elevator	-4.74	-5.24	9.54
Inboard 1	18.87	17.35	8.06
Inboard 2	27.22	27.78	2.02
Outboard 1	29.65	30.00	1.17
Outboard 2	12.90	12.71	1.49
Outboard 3	10.67	9.49	12.43
Outboard 4	-4.59	-5.18	11.39
Rudder	-1.51	-1.46	3.31
Sum of Absolute Values of Hinge Moments (lb. in)	4.06E+06	4.04E+06	0.49

Table 7.19 shows the same trend of values between the estimated results from the trained Neural Networks and the validated ones from the GA. However, the errors between those values are relatively low for both the control surface deflections and for the Sum of Absolute Values of Hinge Moments, with better results for Sum of Absolute Values of Hinge Moments. This shows that ANN training for Sum of Absolute Values of Hinge Moments is almost perfect as evidenced by the training regression plot on Fig. 7.21.

For Mach = 0.78, Altitude = 30200 ft. and Roll Rate = 20 deg/s, MSC Nastran and GA were used to find the optimum control surface deflections and the minimum Sum of Absolute Values of Hinge Moments of 4.04E+06 lb.in. The trained ANNs were used to approximate the control surface deflections. The estimated control surface deflections from the ANN are close to the real ones from MSC Nastran and GA with 3-10% error. However, the estimated and true Sum of Absolute Values of Hinge Moments are almost equal, with 0.49 % error, which shows the efficiency of the scheme presented. The ANNs training could, however, be further improved using the techniques listed in the Sect. 3.3.4.3 (Improve Neural Network Training and Avoid Overfitting).

8 Discussions

The prior Artificial Intelligence procedure applied to the HWB case [11] for control allocation optimization in a 2.5-G pull-up maneuver was used to optimize a FSW test aircraft case. Our first attempt for optimization using an Artificial Neural Network resulted in failure. It failed for two primary reasons. First, the ANN that was trained using the control surface deflections as inputs and the Sum of Absolute Values of Hinge Moments as output resulted in negative values for Sum of Absolute Values of Hinge Moments during the optimization! It was then decided to train the ANN between the control surface deflections and all the components of Hinge Moments before summing their absolute values. This overcame the problem of the negative Sum of Absolute Values of Hinge Moments previously obtained during the optimization. However, the values of the control surface deflections leading to an optimal value for the Sum of Absolute Values of Hinge Moments were not able to trim the airplane for the specified 2.5-G. This implied that trim constraints were not being applied in the optimization process. Two different ANNs were next trained to enforce both the load factor and the pitch rate constraints in the optimization. This step lead to successful optimization.

The optimum control surface deflections from the HWB also were only able to trim the airplane for 2.5-G after the trim constraints were added explicitly in the optimization. This suggests that even if the ANN is well trained and the data used for training the ANN satisfied the constraints on the optimization variables (control surface deflections trimming the airplane), those constraints must still be enforced in the optimization process to satisfy the requirements of the problem. This is a major finding and achievement of this research.

A completely different process using stability derivatives was next developed to optimize the Sum of Absolute Values of Hinge Moments. This gave better results than the optimization methods using ANN+GA and SOL 200 in MSC Nastran, and the computational time was reduced from 3 days to less than 15 min for the FSW and from 5 days to less than one hour for the HWB. This is another major achievement of this work. The use of stability derivatives implies a linear control effectiveness model. This is

probably valid if the control surface deflections at trim are not too large. The magnitude of the control deflections depends on the aircraft characteristics. There may be a control surface deflection limit where the effectiveness begins to enter a nonlinear region such as control reversal (for example, due to aeroelastic deformations, shock effects, flow separations), large rate amplitudes, or nonlinear aerodynamics. Thrust vectoring may also be a situation in which effectiveness in modulating forces and moments may not be accurately represented through simple control derivatives. In general, aircraft flight dynamics can also involve nonlinear derivatives. These nonlinear derivatives may be so small at nominal conditions or during gentle maneuvers, that they are negligible. However, they might become more important at off-nominal conditions or near the edge of the flight envelope. There are some control derivatives that are inherently nonlinear such as those involving the drag coefficient (C_d). For trimmed motion, C_d is important, and linear C_d derivatives are not adequate to deal with nonlinear drag characteristics. C_m derivatives also tend to be slightly nonlinear due to the nature of the downwash interacting with the horizontal tails. Some wind tunnel data actually shows that there is some nonlinearity in C_m vs. AoA. So the assumption of linear C_m derivatives may need to be validated against wind tunnel data or high-fidelity CFD.

To take advantage of both the computational efficiency of the method using stability derivatives and the ability of Neural Network to estimate functions with large number of inputs, a hybrid scheme combining both methods was developed to provide a real-time estimate of the optimum control deflection schedules to trim the airplane and minimize the actuation power for changing flight conditions in a pull-up maneuver. The results show the same trend of values between the estimated results from the trained Neural Networks and the validated ones from the GA, which demonstrates the efficiency of the scheme.

An antisymmetric, steady roll maneuver was performed on the HWB. The process using the stability derivatives was also employed for the control surface deflections schedule optimization for a steady roll maneuver with Mach=0.5, Dynamic Pressure= 1.769 psi and Roll Rate = 15 deg/s. The optimized Sum of absolute Values of Hinge Moments and control surface deflections obtained from GA were equal to those obtained from MSC Nastran for the same flight conditions. The hybrid scheme combining both GA

and ANN was also applied for a steady roll maneuver, and the estimated results from the ANNs were in agreement with the true values from MSC Nastran and the GA. This demonstrates that the scheme can be used for a wide variety of aircraft configurations and other maneuvers for real-time, efficient deployment of the control surfaces to trim the airplane with the minimum power consumption in the event of a change in the flight conditions.

Future work will include detailed studies of the range of applicability of the stability derivatives method, conducting asymmetric maneuvers (engine out and dynamic overswing) and adding actuator dynamics in terms of damping, stiffness and power consumption into the analysis to have a realistic fully aeroservoelastic MSC Nastran model of the HWB platform.

References

[1] Shankman, S., *3 Biggest Challenges Facing the Global Aviation Industry*, https://skift.com/2014/10/14/3-biggest-challenges-facing-the-global-aviation-industry/, 2014.

[2] Liebeck, R.H., "Design of the BWB Subsonic Transport", AIAA-2002-0002, *40th AIAA Aerospace Sciences Meeting and Exhibit*, January 14-17, 2002, Reno, NV.

[3] Gern, F.H., Vicroy, D.D., Mulani, S.B., Chhabra, R., Kapania, R.K., Schetz, J.A., Brown, D., and Princen, N.H., "Artificial Intelligence Based Control Power Optimization on Tailless Aircraft," *ARMD Seedling Fund Phase I Final Report, NASA Technical Memorandum*, NASA/TM-2014-218671, November, 2014.

[4] Cameron, D. and Princen, N., "Control Allocation Challenges and Requirements for the Blended Wing Body," *AIAA Paper* 2000-4539, AIAA Guidance, Navigation, and Control Conference and Exhibit, August 14-17, 2000, Denver, CO.

[5] Rose, R., Nicolas O.P., and Glynis Vo. "*Flight Measurements of the Elevator and Aileron Hinge-Moment Derivatives of the Fairey Delta 2 Aircraft up to a Mach Number of 1"6 and Comparisons with Wind-tunnel Results*", Reports and Memoranda No. 3485, July 1965.

[6] Shahrokhi, A., Jahangirian, A., An efficient aerodynamics optimization method using a genetic algorithm and a surrogate model, *16th Australasian Fluid Mechanics Conference*, 2nd -7thDecember, 2007.

[7] Xu Y M, Li and Rong X., Composite structural optimization by genetic algorithm and neural network response surface modeling. *Chinese Journal of Aeronautics* 2005; 18(4):310-316.

[8] Rajkumar, T., and Bardina J., Prediction of aerodynamic coefficients using neural networks for sparse data, *FLAIRS Conference*, 2002.

[9] Askali, M., Azouaoui, A., Nouh, S., & Belkasmi, M., On the computing of the minimum distance of linear block codes by heuristic methods. *arXiv preprint arXiv:1303.4375*, 2013.

[10] Chhabra, R., Mulani, S., Kapania, R., and Schetz, J. "Control Power Optimization using Artificial Intelligence for Hybrid Wing Body Aircraft", *AIAA Aviation,* 16th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, June 22-26, 2015, Dallas, TX.

[11] Chhabra, R., *Control Power Optimization using Artificial Intelligence for Hybrid Wing Body Aircraft*, Master's thesis, Virginia Polytechnic Institute and State University, Blacksburg, VA, 2015.

[12] Rodden, W.P. and Johnson, E.H. (Editors), MSC. Nastran Version 68, *Aeroelastic Analysis User's Guide*, MSC Software.

[13] Agarwal, T., *Artificial Neural Networks (ANN) and Different Types*, https://www.elprocus.com/artificial-neural-networks-ann-and-their-types/.

[14] Boeree, C.G., *General Psychology*, http://webspace.ship.edu/cgboer/theneuron.html.

[15] Rojas, R., "*Neural Networks: A Systematic Introduction,"* Springer, Berlin. 1996

[16] Zupan, J. Introduction to artificial neural network (ANN) methods: what they are and how to use them. *Acta Chimica Slovenica*, *41*, 327-327, 1994.

[17] Rojas, R., Unsupervised learning and Clustering Algorithms. In *Neural Networks* (pp. 99-121). Springer Berlin Heidelberg, 1996.

[18] Graupe, D. *Principles of Artificial Neural Networks*. Vol. 7. World Scientific, 2013.

[19] Wang, B. *Chemosensors: Principles, Strategies, and Applications (*pp. 86-87), 2011.

[20] Gershenson, C., "*Artificial Neural Networks for Beginners*." arXiv preprint cs/0308031, 2003.

[21] *Neural Network Toolbox User's Guide*, available at https://www.mathworks.com/help/nnet/gs/product-description.html, 2015, The MathWorks, Inc.

[22] *Neural Network Toolbox User's Guide*, available at https://www.mathworks.com/help/nnet/ug/workflow-for-neural-network-design.html, 2015, The MathWorks, Inc.

[23] *Neural Network Toolbox User's Guide*, available at https://www.mathworks.com/products/neural-network/features.html, 2015, The MathWorks, Inc.

[24] *Neural Network Toolbox User's Guide*, available at https://www.mathworks.com/help/nnet/ug/neuron-model.html, 2015, The MathWorks, Inc.
[25] Sarle, W. S., *Neural Network FAQ, part 3 of 7: Generalization*, 1997.
[26] Efron, B. and Tibshirani, R. J., *An Introduction to the Bootstrap*, 1993, Chapman & Hall, London.
[27] Beale, M.H., Hagan, M.T. and Demuth, H.B., *Neural Network Toolbox User's Guide*, available at https://www.mathworks.com/help/nnet/ug/train-and-apply-multilayer-neural-networks.html, 2015, The MathWorks, Inc.
[28] *Neural Network Toolbox User's Guide*, available at https://www.mathworks.com/help/nnet/ug/limitations-and-cautions.html, 2015, The MathWorks, Inc.
[29] Deepa, S.N., "*Introduction to Neural Networks Using Matlab 6.0*", 2006.
[30] *AI-junkie*, available at http://www.ai-junkie.com/ga/intro/gat1.html.
[31] Mitchell, M., "*An Introduction to Genetic Algorithms"*, 1999.
[32] Winter, G., "*Genetic Algorithms in Engineering and Computer Science"*, 1995
[33] Rawlins, J.G., "*Foundations of Genetic Algorithms"*, 1991.
[34] ABrandao, "Genetic Algorithms in PHP code", available at http://www.abrandao.com/2015/01/simple-php-genetic-algorithm/, 2015.
[35] https://www.mathworks.com/discovery/genetic-algorithm.html, 2015, The MathWorks, Inc.
[36] Dimitriadis, G., Aircraft Design, available at http://www.ltas-cm3.ulg.ac.be/AERO0023-1/ConceptionAeroAeroelasticite.pdf.
[37] Bisplinghoff, R.L. and Ashley, H., Half man, RL, Aeroelasticity, 1955.
[38] "*Introduction to the Aerodynamics of Flight*", available at http://history.nasa.gov/SP-367/chapt9.htm#f131.
[39] MSC Nastran 2008, Static Aeroelastic Analysis, MSC Software Corporation, 2 MacArthur Place, Santa Ana, CA 92707, USA, 2008.
[40] Rodden, W.P. and Johnson, E.H. (Editors), MSC. Nastran Version 68, Aeroelastic Analysis User's Guide, MSC Software.

[41] Caughey, D., Introduction to Aircraft Stability and Control, available at https://courses.cit.cornell.edu/mae5070/Caughey_2011_04.pdf, 2011.

[42] Nielsen, Norman, J. and Goodwin, F. K., *Preliminary Method for Estimating Hinge Moments of All-Movable Controls*. No. NEAR-TR-268. Nielsen Engineering And Research Inc Mountain View CA, 1982.

[43] Keramat, M. and Kielbasa, R., "Latin Hypercube Sampling Monte Carlo Estimation of Average Quality Index for Integrated Circuits", Analog Integrated Circuits and Signal Processing, Vol. 14, No. 1/2, pp. 131-142, 1997.

[44] MSC Nastran 2008, Quick Reference Guide, MSC Software Corporation, 2 MacArthur Place, Santa Ana, CA 92707, USA, 2008.

[45] Han, Z.-H. and Zhang, K.-H., Surrogate-Based Optimization, Real-World Applications of Genetic Algorithms, Dr. Olympia Roeva (Ed.), ISBN: 978-953-51-0146-8, InTech, Available from: http://www.intechopen.com/books/real-world-applications-of-genetic-algorithms/surrogate-based-optimization, 2012.

[46] Mallik, W., Kapania, R. K., and Schetz, J. A, Aeroelastic Applications of a Variable-Geometry Raked Wingtip. *Journal of Aircraft*, 1-13, 2016.

[47] Adegbindin, M., Love, N., Kapania, R. K., and Schetz, J. A. "Control Power Optimization Using Artificial Intelligence For Forward Swept Wing And Hybrid Wing Body Aircraft." *17th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*. p. 3361. 2016.

[48] Fahlman, S.E. and Lebiere, C. (1990), "The Cascade Correlation Learning Architecture," NIPS2, 524-532, available at https://papers.nips.cc/paper/207-the-cascade-correlation-learning-architecture.pdf.

[49] Orr, M. J. "Introduction to radial basis function networks." 1996.

[50] Quinlan, J. R., & Gern, F. H. (2016). Optimization of an Advanced Hybrid Wing Body Concept using HCDstruct Version 1.2. *analysis*, *5*, 6.

[51] Yousif, S. T., and Najem, R. M., "Optimum cost design of reinforced concrete continuous beams using Genetic Algorithms." *International Journal of Applied Science and Engineering Research* 2.1 (2013): 79-92.