OPEN LOOP LONGITUDINAL ENVELOPE PROTECTION FOR AIRCRAFT IN ICING ENCOUNTERS

BY

KISHWAR NAZ HOSSAIN

B.S.M.E, Lafayette College, 2000

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Aeronautical and Astronautical Engineering
in the Graduate College of the
University of Illinois at Urbana Champaign, 2003

Urbana, Illinois

Abstract

In-flight icing has been recognized as a safety threat to aircraft operations since the 1930's. Although significant progress has been made over the past 70 years in improving flight safety in icing conditions, recent icing accidents and incidents have been reported. Thus further developments are necessary in order to improve aircraft icing safety. The goal of this research was to improve aircraft safety through enhancing the envelope protection capabilities of an aircraft in icing conditions. To accomplish this goal, an open loop envelope protection algorithm was developed to ensure the safe operation of an iced aircraft during the manual mode of flight. The Iced Aircraft Envelope Protection system (IAEP), developed as a part of the Smart Icing Systems (SIS) research project at the University of Illinois, was based on data from wind tunnel tests, flight tests and iced aircraft simulations obtained from a six-degree-of-freedom computational flight dynamics model. The system consisted of estimative and predictive methods for approximating, and avoiding the envelope boundaries. Simulation results demonstrated that IAEP was capable of successfully avoiding incidents and accidents during flight in icing conditions.

Acknowledgements

I would like to acknowledge the contributions of all the people involved in the Smart Icing Systems Project. I would especially like to thank my advisor Dr. Michael Bragg for his guidance, support and patience. It was a privilege to be able to work with him and I really appreciate all his advice and ideas toward completing this project. I would also like to acknowledge the NASA-Glen grant NAG 3-2135 for supporting the Smart Icing Systems research.

Dr. Sam Lee and Ed Whalen deserve special thanks for helping me in different aspects of the project. I would also like to thank Tom Ratvasky at NASA Glen for providing me with data that was crucial to this project.

I would also like to thank my parents and my brother for their continued support in my endeavors. They have always been there for me and provided me with encouragement and love whenever I needed it most. I am grateful to them for that.

Finally I would like to thank Pesh for his support and love. I couldn't have done it without him.

Table of Contents

Table of Figures vi

1 INTRODUCTION 1

1.1 Motivation 1
1.2 Performance Degradation Due to Icing 4
1.3 Envelope Protection 8
1.3.1 The Operating Flight Envelope 8
1.3.2 Current Envelope Protection Systems 8
1.3.3 A Prediction Based Envelope Protection Scheme 9
1.4 Smart Icing Systems 11

2 METHOD AND APPROACH 15

2.1 Icing Characterization 15
2.2 Iced Aircraft Envelope Protection Methodology 19
2.3 Critical Parameter and its Limit Boundary 20
2.4 Dynamic Trim 24
2.5 Open-Loop IAEP 26
2.5.1 The Open-Loop IAEP in Matlab 28
2.5.2 ODE Solver 29
2.6 Linearized Equations of Motion 29
2.7 Modification of the Flight Dynamics Code 31
2.7.1 Non-linear Aerodynamic Model 31
2.7.2 Implementing the IAEP in the FDC 32
2.8 Figures 33

3 RESULTS AND DISCUSSION 41

3.1 Validation 42
3.1.1 Estimative IAEP 42
3.1.2 Predictive IAEP 43
3.2 Integration Time for IAEP Predictive Solutions 45
3.3 The Maximum Control Deflection 48
3.4 Effectiveness of the IAEP Scheme 49
3.5 Figures 52

4 SUMMARY, CONCLUSIONS AND RECOMMENDATIONS 66

4.1 Summary 66
4.2 Conclusions 67
4.3 Recommendations 68

APPENDIX 70

REFERENCES 97

Table of Figures

Fig. 1.1. Operating Flight Envelope of a Typical Fighter 14

Fig. 2.1. Coefficient of Lift for Different Icing Conditions 33

Fig. 2.2. Coefficient of Pitching Moment as a Function of the Icing Condition and Angle of Attack 33

Fig. 2.3. Coefficient of Pitching Moment as a Function of Elevator Deflection and the Icing Conditions 34

Fig. 2.4. Comparison of Maximum Lift Coefficient with the Change in Required Angle of Attack for Fixed Lift Coefficient 34

Fig. 2.5. Variation of Maximum Lift Coefficient with Drag Rise 35

Fig. 2.6. Comparison of Zero Lift Drag Coefficient with η_{ice} 35

Fig. 2.7. Variation in Maximum Lift with Change in Lift at Fixed Angle of Attack 36

Fig. 2.8. Angle of Attack Response Due to Step Elevator Inputs from FDC Simulations 36

Fig. 2.9. Time History of Rate of Change of the Angle of Attack 37

Fig. 2.10. Time History of Rate of Change of the Pitch Rate 37

Fig. 2.11. Angle of Attack Response Computed using Linearized Equations of Motion 38

Fig. 2.12. Incorporation of Non-linear Aerodynamic Model in Simulink 38

Fig. 2.13. Open-Loop Configuration of Simulink Structure Apilot with IAEP Modications 39

Fig. 2.14. IAEP Modified Aircraft Dynamics Structure 39

Fig. 2.15. Modified Block AEROMOD 40

Fig. 3.1. Angle of Attack Response for Validation at $\eta = 0.05$ 52

Fig. 3.2. Elevator Command for Validation at $\eta = 0.05$ 53

Fig. 3.3. Altitude Time History for Validation at $\eta = 0.05$ 53

Fig. 3.4. Comparison of Estimated Stall Angles with Simulation Values 54

Fig. 3.5. Validation of Open Loop IAEP Prediction with FDC Result 54

Fig. 3.6. Validation of Open Loop IAEP Prediction with Flight Test Data 55

Fig. 3.7. Validation of Open Loop IAEP Prediction with Flight Test Data 55

Fig. 3.8. IAEP Performance for Different Integration Times for Ramped Elevator Input. 56

Fig. 3.9. IAEP Performance with Different Integration Times for Step Elevator Input 56

Fig. 3.10. Angle of Attack Response from Issuing Step Command Equal to Elevator Limit at η=0.01 57

Fig. 3.11. Step Elevator Input Equal to the Limit Value at η=0.01 57

Fig. 3.12. Angle of Attack Response from Issuing Step Command Equal to Elevator Limit at η=0.05 58

Fig. 3.13. Step Elevator Input Equal to the Limit Value at η=0.05 58

Fig. 3.14. Angle of attack Response from Issuing Step Command Equal to Elevator Limit at η=0.1 59

Fig. 3.15. Step Elevator Input Equal to the Limit Value at η=0.1 59

Fig. 3.16. Comparison of Simulations with and without IAEP for a 10^{o} Elevator Input at η=0.01 60

Fig. 3.17. IAEP Elevator Command for an Initial Step Elevator at η=0.01 60

Fig. 3.18. Comparison at η=0.05 61

Fig. 3.19. IAEP Elevator Command at η=0.05 61

Fig. 3.20. Comparison at η=0.1 62

Fig. 3.21. IAEP Elevator Command at η=0.1 62

Fig. 3.22. Comparison of simulations with and without IAEP with Initial Ramp Elevator Command at η=0.01 63

Fig. 3.23. IAEP Elevator Command at η=0.01 63

Fig. 3.24. Comparison at η=0.05 64

Fig. 3.25. IAEP Elevator Command at η=0.05 64

Fig. 3.26. Comparison at η=0.1 65

Fig. 3.27. IAEP Elevator Command at η=0.1 65

1 INTRODUCTION

1.1 Motivation

In-flight icing has been recognized as a safety threat to aircraft operations since the 1930's. Although extensive research efforts have been undertaken since then to prevent icing related catastrophes, recent accidents, such as the American Eagle roll upset near Roselawn, Indiana in October 1994 and the Com Air accident in January 1997 clearly show that icing continues to be a serious safety concern.[1] In addition, recent accident and incident reports analyzed for the Aviation Safety Program (AvSP) showed that 13% of all weather-related accidents were due to airframe icing.[2] Thus it is evident that further research is necessary in order to drastically reduce icing related incidents and accidents.

Icing accidents are caused mainly due to the operation of an aircraft outside its safe flight envelope. Ice accretion, which causes changes in the performance, stability and control of an aircraft, reduces the safe flight envelope through higher stall speeds, lower stall angles, etc. Recent flight tests conducted using the NASA Icing Research Aircraft, N607NA, showed that the idle power stall speed for the aircraft with zero flaps reduced from 70 kias for a baseline "no ice" case, to 77 kias for the simulated "all iced" case. This change in the stall speed corresponded to a stall angle of attack reduction of 3^{o}.[2]

The stall warning systems that are required by Code of Federal Regulations (CFR) Part 25 are intended to provide flight crews with adequate warning of proximity to the stall angle of attack; however, the current systems often do not provide adequate warning when the airplane is operating in icing conditions when the stall angle of attack is markedly reduced.[3] Thus, the crew is unprepared for the unpleasant or even fatal consequences of maneuvers that would be considered safe in a clear air environment.

The ineffectiveness of current envelope protection systems to cue pilots of the ice accretion induced reduction of the safe flight envelope has led to many incidents and accidents. In the NTSB report on the Com Air accident of 1997 there are comments suggesting that the lack of information regarding the safe flight limits lead to the operation of the aircraft beyond its aerodynamic boundaries resulting in an accident which caused 29 deaths,

> "contributing to the accident were the flight crew's decision to operate in icing conditions near the lower margin of the operating airspeed envelope

(with flaps retracted) and Com Air's failure to establish and adequately disseminate unambiguous minimum airspeed values for flap configurations and for flight in icing conditions."[3]

Further more, the Safety Board concluded that the stall warning system installed in the accident airplane did not provide an adequate warning to the pilots because ice contamination was present on the airplane's airfoils and the system was not designed to account for aerodynamic degradation or adjust its warning to compensate for the reduced stall warning margin caused by the ice.[3]

The ATR 72 accident of 1994 can also be attributed to operation of the aircraft beyond its safe flight envelope. The National Transportation Safety Board determined that the probable causes of the accident were the loss of control, due to a sudden and unexpected aileron hinge moment reversal, that occurred after a ridge of ice accreted beyond the deice boots.[5] The aileron hinge moment reversal was caused by the separated flow over the iced wing of the aircraft at an angle of attack of 5^{o}.[4] Since ATR indicated that the aileron hinge moment reversals were linked to aerodynamic stall,[5] it can be concluded that, although the aural warning and stick shaker of the aircraft was set to start at 11.2^{o}, the wings of the ATR 72 stalled at a lower angle of attack and thus there was no warning issued to the pilot.

In view of the Com Air, ATR and other major aircraft icing incidents and accidents, where the inadequacy of the envelope protection systems resulted in catastrophe, an initiative was taken by NASA and the University of Illinois as part of the Smart Icing

Systems project to further study the effects of icing on the aircraft flight envelope and develop the logic for envelope protection systems for safer operations in icing conditions. This document provides a detailed description of the analysis done and the system developed to enhance the performance of envelope protection systems in icing conditions.

1.2 Performance Degradation Due to Icing

Numerous experiments have been conducted over the past 70 years to gather information about icing related performance degradation. These experiments ranged from flight tests in natural icing conditions to wind tunnel tests on airfoils with simulated ice shapes. Results from these experiments showed that icing causes significant changes in aerodynamic parameters such as the lift, drag and moment. As early as 1940, Johnson[6] reported that wind tunnel tests showed reductions of the maximum lift coefficient of up to 32% with flaps retracted and 28% with flaps extended. Johnson also presented results showing reduction in the stall angle of attack and stated that; "Because of the fact that the normal stalling angle is reduced from 13^o to 9^o the pilot is confused because he has no means for knowing when he is approaching the stall." Thus, from the inception of icing tests it was known that icing may drastically reduce the effective flight envelope of an aircraft and thus efforts have continued to identify trends in icing related performance degradations.

In 1976, following a number of serious icing related accidents in the 60's and 70's, a joint Swedish-Soviet task force was formulated to study the physical, meteorological and

aerodynamical aspects of the icing problem.[7] In 1977, in their first report, the group documented experimental results from wind tunnel tests using simulated ice shapes of different thicknesses and shapes on wing sections.[8] These results showed reductions of about 25% in the C_{Lmax} and a 12% increase in the stall speed. However, it was found that the shape of the ice accretion was much more crucial than its size. From the flight test data, it was found that there were reductions of 31% in the C_{Lmax} and a 3^o decrease in the stall angle due to ice deposit shapes mounted on the leading edge of the wing. In a second report published in 1979, the group reported that there may be losses in elevator effectiveness of up to 35% due to even slight ice roughness on the leading edge due to premature flow separation in the lower surface which may cause large and sudden changes in the hinge moment and the elevator stick force.[9]

In 1985 Mikkelson et al.[10] conducted flight tests using the Twin Otter aircraft in natural icing to document ice shapes, measure section drag of an iced wing and measure the overall performance loss of the aircraft caused by ice. These flight tests showed that even with the wings deiced, the drag was 26 % over the baseline clean value. At an angle of attack of 6^o, the lift coefficient was about 11 percent lower than the uniced baseline for a flight flown in mixed rime and glaze conditions for 22 min, with LWC of 0.58 g/m^3. In 1986, Ranaudo et al.[11] attempted to identify relationships between measured icing cloud properties and the effect these properties had on aerodynamic coefficients and stability derivatives. Results from these tests showed that the shape of aircraft ice accretions on both lifting and non-lifting surfaces was the most important factor influencing performance. Ranaudo et al. reported degradations in lift coefficient of 7%-8% at an

angle of attack of 6^{o} and up to 16.1% degradation in elevator effectiveness derivatives with flaps extended and 11.6% with flaps retracted. However, since most of the data for these tests were taken in the linear aerodynamic range, no correlations could be made between the degradation in force and moment coefficients or stability and control derivatives at low angles of attack with changes in the stall characteristics of the aircraft.

In 1994 the Tailplane Icing Program was formulated to further the understanding of iced tailplane aeroperformance and aircraft aerodynamics.[12,13] Both wind tunnel and flight tests were performed as a part of this program. The wind tunnel tests were performed using models of the horizontal tail section of the Twin Otter. The test aircraft used was a modified Twin Otter. From the wind tunnel tests it was found that even with inter-cycle icing the stall angle of attack of the tail-plane was reduced by 2.3^{o} and for the LEWICE and S&C shapes this discrepancy rose to 9.5^{o}. One of the major observations from the subsequent flight tests was that there was loss of elevator authority with increasing ice severity. It was reported that for one of the maneuvers the baseline trim elevator deflection was 6.7^{o} while for the S&C shape it was –1.3. Thus it may be useful to utilize elevator and aileron trim values as ice detectors.

Lee et al.[14,15] investigated the effects of ice accretion shape, size and location on the NACA 0012, NACA 23012m and NLF 0414F airfoils. These experiments showed that the both ice shape geometries and size (height) had significant effects on the extent of the performance degradation of the airfoils namely reductions in the stall angle of attack and

C_{lmax}. They also concluded that the most severe performance degradations were observed when the ice shape was located near the adverse pressure recovery region of the airfoils.

From the review of icing research presented above it is evident that considerable effort has been expended on identifying the effects of icing on aerodynamic performance. One of the most important conclusions that can be drawn from these efforts is that ice accretion may cause severe penalties in terms of the flight envelope of an aircraft. This fact is imbedded in the observations of increased stall speed, and decreased C_{Lmax} and stall angle values for iced airfoils and test flights in icing conditions. However, very little information exists in the literature to quantify the reduction in the flight envelope in terms of icing conditions. Also, no method has been developed thus far to inform the pilot of icing related premature stall in advance so that preventive measures can be taken in time to avoid accidents and incidents.

Hence, research was conducted as a part of the Smart Icing Systems (SIS) research project to develop an iced aircraft envelope protection system to avoid icing related incidents and accidents. The envelope protection system thus developed utilizes available sensor information and icing characterization information obtained from other aspects of the SIS. It was modeled on current envelope protection systems in terms of the aerodynamic parameters to be limited and the limiting methods used. A brief overview of the SIS and current envelope protection systems was thus included in the following sections.

1.3 Envelope Protection

1.3.1 The Operating Flight Envelope

The flight envelope of an aircraft typically maps the combinations of altitude and velocity that the aircraft has been designed to withstand.[16] Other restrictions are added to the definition of the flight envelope depending on the physical, aerodynamic and structural limits of the aircraft. Figure 1.1 shows the operating envelope for a typical fighter.

1.3.2 Current Envelope Protection Systems

Currently most envelope protection systems provide the pilot and autopilot preset limits for parameters such as the speed, angle of attack, bank angle etc. For example the stall speed warning for the Boeing 777 is activated at 148 kts with no flaps and 118 kts when the flap is deflected at 20^{o}. The force feedback increases at a rate of 15 lbs per degree increase in the angle of attack beyond the onset of the stick shaker. The lateral envelope protection system of the 777 issues a visual cue at a bank angle of 30^{o} and an aural warning at a bank angle of 35^{o}.[17]

The Airbus A320, A330 and A340 envelope protection systems limit the pitch attitude for the aircraft between $+30^{o}$ and -15^{o}. For lateral protection the bank angle is limited to 67^{o} at moderate angles of attack and 45^{o} at high angles of attack.[17]

If there is ice accretion on the aircraft, preset limits that correspond to the performance boundaries of a clean aircraft are not applicable. As a result there is an increased chance

of incidents and accidents if the pilot or autopilot decides to operate the aircraft close to the clean flight envelope. Due to numerous icing related incidents and accidents the envelope protection systems aboard some aircraft were modified to account for the performance degradation due to ice accretion. For example in the ATR 72 the Stall Protection System (SPS) operates in conjunction with the IPS to reduce the angle of attack limit from 18.1^{o} to a predetermined value of 11.2^{o} in icing conditions.[18] However, this value is not modified based on the actual ice accretion although the performance degradation may cause the aircraft to stall at a much lower angle of attack. For example in the ATR accident of 1994 the roll anomaly occurred at an angle of attack of 5^{o}.[18]

1.3.3 A Prediction Based Envelope Protection Scheme

A novel envelope protection method was developed by Horn et al[19] at the Georgia Institute of Technology, to enhance the ability of safely flying a rotorcraft to the edges of its true operational envelope within reasonable pilot workload constraints.[19] Horn et al developed a system to provide enhanced cueing of the envelope limits to the pilot by utilizing the concept of dynamic trim,

> "a dynamic flight condition that the pilot is likely to sustain over several seconds in order to maneuver the aircraft,"[19]

along with neural networks to predict if the aircraft would exceed its flight envelope during the course of a maneuver. In this method a multi-layered feed-forward neural network was applied to approximate the values of the envelope parameters using values of measured aircraft states and control deflections as inputs. Thus, instead of implementing envelope limits using instantaneous sensor data, a system was developed to

predict the future values of critical parameters, at the beginning of a maneuver based on the aircraft state and the control inputs. The aircraft state was characterized using state parameters such as velocity, angle of attack, pitch rate etc., and control parameters such as elevator and aileron deflections. The critical parameters and equations of motion were then expressed as functions of the state and control parameters as shown in equations (1.1) and (1.2). At the onset of a maneuver, the equations of motion were solved for a future dynamic trim state where the rate of change of the aerodynamic angles and the rotational velocities is zero as defined by equations (1.3) and (1.4).

$$\dot{\bar{x}} = g(\bar{x}, \bar{u}) \tag{1.1}$$

$$\bar{y}_p = \bar{y}_p(\bar{x}, \bar{u}) \tag{1.2}$$

$$\dot{\alpha} = \dot{q} = 0 \tag{1.3}$$

$$\dot{p} = \dot{r} = \dot{\beta} = 0 \tag{1.4}$$

The estimated state was then used to calculate the value of the critical parameters at dynamic trim. The limits of the critical parameters were found as functions of control inputs and state parameters using simulations.

The neural net training was carried out using aircraft state data covering the entire flight envelope. The data were obtained from simulations using modified trim routines that generated trim data in quasi-steady maneuvers. The output from the neural network, which consisted of the quasi steady state value of the envelope parameter was compared

to the limit value of the envelope parameter to calculate the "critical control margin" used to determine the amount of force-feedback to be relayed to the pilot.

1.4 Smart Icing Systems

The goal of the Smart Icing Systems project was to equip commuter class and other small aircraft with a system to successfully avoid icing accidents. In order to do this the concept of the Ice Management System (IMS) was developed.[1,20] The functions of the proposed IMS included measuring in flight the effect of ice on performance and control, managing the IPS, performing envelope protection, and adapting the flight controls. The inputs to the system consisted of the icing sensors, the ice protection system, the flight crew, the aircraft flight dynamics and other aircraft state information. The research initiative was divided into four groups, aerodynamics and flight mechanics, sensors and control, human factors and flight simulation.

The aerodynamics and flight mechanics group of the SIS developed an icing effects model which quantified the effect of ice accretion on the aircraft using an icing severity parameter.[21,22] The group also modified a six degree of freedom computational flight dynamics code (FDC)[23] to include sensor noise, non linear aerodynamics models, hinge moment models and atmospheric disturbance models.[1] The primary motivation for developing this capability was derived from the need to perform simulation studies and aid in the development and testing of different modules of the SIS.[24] The research reported in this document was completed as one of the tasks of the aerodynamics and

flight mechanics group. The simulation results reported in this document were obtained using the modified version of FDC as mentioned above.

The focus of the sensors group was on characterizing the effect of ice accretion. Neural networks were trained using simulation data in order to identify and characterize the effects of icing on the performance and control of an aircraft from sensor information. Detailed description of the characterization effort can be found in a conference paper published by Melody et al. in 2001.[25] The sensors and controls group also developed an autopilot with modified control laws for improved performance in icing conditions.[26] The autopilot was also modified to incorporate flight envelope information and fly the aircraft within its aerodynamic bounds.[27]

The role of the Human Factors Group in SIS was to present the IMS-generated icing-related information to pilots in a meaningful and effective manner in order to support them in making decisions and selecting/executing actions under time pressure and uncertainty.[1]

The icing flight simulator was developed as a platform for the demonstration of SIS capabilities in a real-time piloted setting. The simulator acted as a systems integrator for testing the entire gamut of functions of the SIS.[28]

As mentioned earlier the research reported in this document was conducted as a part of the SIS project. The iced aircraft envelope protection system is a module, which receives

information from and imparts information to other modules in the system. The system developed here is especially reliant on the nonlinear aircraft model developed by the aerodynamics and flight mechanics group and the icing characterization developed by the sensors and controls group.

This thesis documents the research conducted to develop envelope protection methods for an iced aircraft. The research included analysis of wind tunnel data to obtain information about the flight envelope of an iced aircraft, develop algorithms to represent the flight envelope as a function of icing severity and develop the capability to predict envelope violation.

Due to a lack of information pertaining to the flight envelope of an iced aircraft a rather simple method was developed to estimate the flight envelope of an iced aircraft. This estimative algorithm was based on two-dimensional wind tunnel test data from experiments conducted at the University of Illinois. A discussion of the data used and the algorithm developed is given in chapter 2.

A predictive method was developed to provide SIS the ability to issue stall warnings to the pilot in time to avoid accidents. Although the dynamic trim based system described above provided many of the desirable features needed for the envisioned iced aircraft envelope protection system simulation results showed that the idea of dynamic trim was not applicable for the aircraft model being used for icing research. Thus an alternative system, based on online solutions of the equations of motion was developed for the

predictive aspect of the envelope protection system. Details of this system were given in chapter 2.

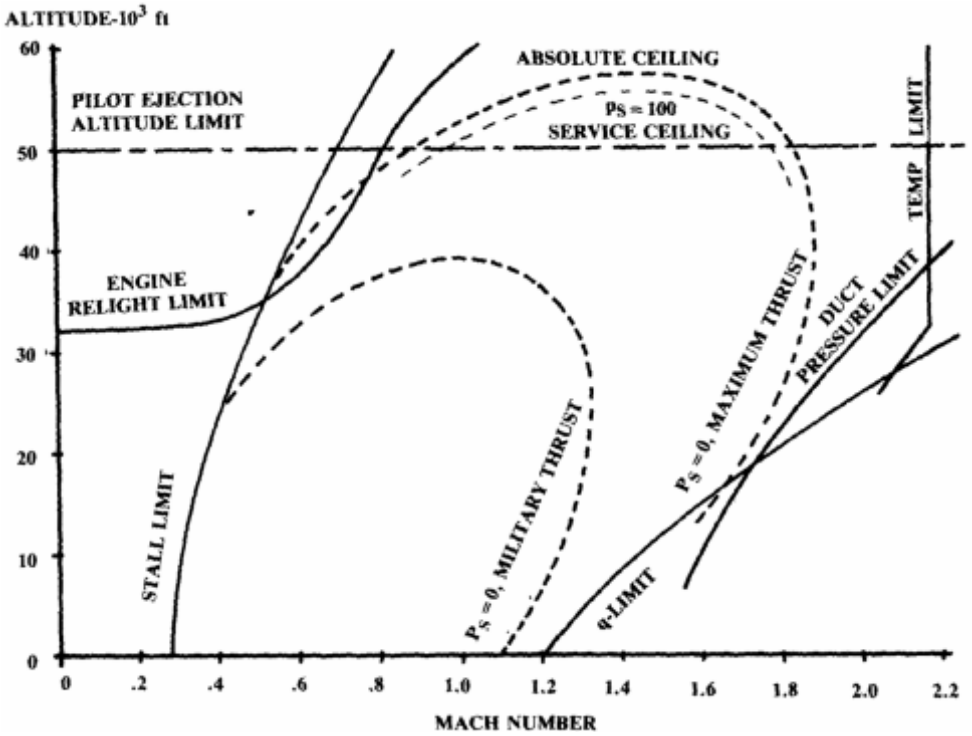

Fig. 1.1. Operating Flight Envelope of a Typical Fighter[16]

2 METHOD AND APPROACH

The development of an Iced Aircraft Envelope Protection (IAEP) system involved research in many areas, including the modification of existing icing aerodynamic models to include the non-linear effects of ice accretion near stall, the identification of the critical envelope parameters, and running simulations to test the system. A detailed account of these modifications and developments were presented in this chapter.

2.1 Icing Characterization

In order to formulate an iced aircraft envelope protection scheme, it was imperative to understand fully the effect of icing on the performance, stability and control parameters of the aircraft, especially near the edges of the flight envelope. Since there weren't any existing models for an iced aircraft near the non-linear aerodynamic region, it was

necessary to formulate one from the limited data available. The non-linear aerodynamic model, thus developed, was built on the principle of the simple model of icing effects as created by Bragg *et al.*[21]:

$$C_{(A)iced} = (1 + \eta_{ice} k'_{C_A}) C_{(A)} \tag{2.1}$$

η_{ice} as seen in Eq. (2.1) is the icing severity parameter. It depends only on icing conditions and its formulation was based on the drag rise of the NACA 0012 airfoil in icing conditions. η_{ice} was defined as the ratio of the drag rise of a 3 ft chord NACA 0012 airfoil at a velocity of 175 knots to the drag rise on the same airfoil in continuous maximum conditions. However, since the drag rise was a function of the freezing fraction, the accumulation parameter and the collection efficiency, η_{ice} was expressed as follows:

$$\eta_{ice} = f(n, A_c, E) \tag{2.2}$$

A more detailed account of the development of η_{ice} can be found in Bragg *et al.*[21]

k'_{C_A} in Eq. (2.1) represents the aircraft specific changes in the stability and control parameter due to icing:

$$k'_{C_A} = \frac{\eta}{\eta_{ice}} k_{C_A} \tag{2.3}$$

where, η is calculated in the same way as η_{ice} except the chord and the velocity correspond to the aircraft and conditions being studied.

The above model was applied to the Twin Otter using non-linear aerodynamic data obtained from wind tunnel tests performed using a 6.5% model of the Twin Otter in the low-speed wind tunnel at Witchita State University and Burill Applied Research's Low Amplitude Multiple Purpose facility at Neuburg a.d. Donau, Germany.[2] These tests were run using the clean configuration and two different simulated ice configurations: 1) Tail iced and 2) Tail and wing iced. The simulated ice configurations corresponded approximately to a wing η value of 0.08 and a tail η eta value of 0.20.

Since data were available for the limited icing conditions mentioned above, a scaling method was adapted in order to model icing effects at varying icing severities. Thus the $k_{C_{(A)}}$ was scaled as follows:

$$k_{C_{(A)}} C_{(A)clean} = -\frac{\Delta C_{(A)}}{\eta_{ref}} \tag{2.4}$$

In Eq. (2.4) $\Delta C_{(A)}$ is the iced value of a stability and control coefficient minus its clean, uniced value and η_{ref} is the icing parameter representative of the simulated icing condition for the wind tunnel tests.

Although the original icing effects model proposed by Bragg *et al*.[21] as represented by Eq. (2.1) gave the value of the stability and control parameter $C_{(A)}$, for the non-linear

aerodynamic model, the icing effects on the aircraft force and moment coefficients were calculated. This was done because more data were available for these parameters, and also to facilitate the implementation of the icing model in the flight simulator. Another aspect of this model was that the effects of aircraft configuration namely the angle of attack and elevator deflection were included explicitly in the calculations.

The basic form of the non-linear aerodynamic model for the arbitrary force or moment coefficient, C_A was as follows:

$$C_{(A)iced} = C_{(A)clean} + k_{C_{(A)},\alpha}\eta_{wing}C_{(A)clean} + k_{C_{(A)},\delta e}\eta_{tail}C_{(A)clean} \quad (2.5)$$

The η's used in Eq. (2.5) are the icing parameters representative of the pertinent aircraft components being represented by their k coefficients.

Simplifications were used in order to model the lift and drag coefficients. It was found from the wind tunnel data that the lift and drag were not affected significantly by tail ice and so the iced lift and drag coefficients were modeled as follows:

$$C_{L,iced} = C_{L,clean} + k_{C_L,\alpha}C_{L,clean}\eta_{wing} \quad (2.6)$$

$$C_{D,iced} = C_{D,clean} + k_{C_D,\alpha}C_{D,clean}\eta_{wing} \quad (2.7)$$

Figure 2.1 shows plots of the lift coefficient as a function of the angle of attack for the clean aircraft and 2 different icing conditions. As seen from the figure, the coefficients calculated using the non-linear aerodynamic model follow expected trends. The lift decreases as the icing severity increases and the stall becomes sharper with higher η values.

The pitching moment coefficient was modeled as follows:

$$C_{M,iced} = C_{M,clean} + k_{M,\alpha} C_{M,clean} \eta_{wing} + k_{M,\delta e} C_{M,clean} \eta_{tail} \qquad (2.8)$$

Figures 2.2 and 2.3 show plots of the pitching moment as a function of the angle of attack and the elevator deflection respectively, for different icing conditions. As seen from these plots the stability decreased with the icing severity. However, icing seemed to have a larger effect on C_{m_α} than on $C_{m_{\delta_e}}$. This may be attributed to the fact that there is a much larger change in the lift produced by the wing, than in the downforce produced by the tail, which in turn affects the pitching moment. It was also observed that the pitching moment became more nonlinear with increases in the icing severity.

2.2 Iced Aircraft Envelope Protection Methodology

The idea of the Iced Aircraft Envelope Protection (IAEP) system is to provide information to be used to limit the control deflections of an aircraft so that it remains within its flight envelope. A prediction scheme was thus designed to inform the pilot if the aircraft was anticipated to exceed the limits of the operating envelope.

The critical parameters are defined in this study as those that are constrained by aerodynamic boundaries. The excursion of these parameters beyond their limit values may result in loss of aircraft control. During the formulation of the IAEP, a vector consisting of the critical parameters was defined as the envelope vector, $\overline{y}_p$.[19] The limits of the envelope vector, i.e. the maximum and minimum allowable values of the critical parameters at the given icing condition, were defined as functions of the icing parameter η[29]:

$$\overline{y}_p = \overline{y}_p(\overline{x}, \overline{u}, \eta) \tag{2.9}$$

$$\overline{y}_{p\lim} = \overline{y}_{p\lim}(\eta) \tag{2.10}$$

Then the envelope protection problem was simplified to constraining control inputs at each time instant so that the critical parameters remained within their bounds:

$$\overline{y}^{l}_{p\lim} \le \overline{y}_p \le \overline{y}^{u}_{p\lim} \tag{2.11}$$

where, $\overline{y}^{l}_{p\lim}$ and $\overline{y}^{u}_{p\lim}$ were the lower and upper limits on the critical parameter. These equations were key to the development of the IAEP since they described the dependence of the envelope limits on the icing parameter and expressed their bounds.

2.3 Critical Parameter and its Limit Boundary

From a review of icing incidents and accidents it was found that aerodynamically the iced aircraft needed protection from wing stall, horizontal tail stall, roll upset and loss of longitudinal and lateral control. For example, the ATR 72 accident of 1994 near Roselawn, Indiana was caused by "a sudden and unexpected aileron hinge moment reversal," which resulted from the loss of roll control above a specific, but low, angle of attack.[5] Similarly the BA-3101 Jetstream accident of 1989 was attributed to "loss of control at low altitude" which may have been caused when the horizontal stabilizer stalled.[30]

Eventually IAEP is expected to provide protection from all the phenomena mentioned above. However, in this thesis the development of a system for the prevention of wing stall only is addressed. In order to prevent stall the aircraft angle of attack must be maintained at a value lower than the stall, angle limit. Thus, the aircraft angle of attack was chosen as the critical parameter in order to formulate the longitudinal envelope protection scheme.

Having identified the critical parameter, it was necessary to define its boundaries as a function of data expected to be available for an SIS equipped aircraft. In order to develop this capability, data spanning the entire envelope for different icing conditions was needed. However, at this time only very limited data were available on iced aircraft limits from icing flight tests. Hence, for this research a very basic method was developed to calculate the limits of the angle of attack using data obtained from wind tunnel tests performed at the University of Illinois.[31,32,33] The analysis was intended to identify trends in the performance degradation of an airfoil caused by simulated ice-shapes.

An example of the dependence of stall on icing parameters is shown in Figure 2.4. This is a plot of the increment in angle of attack of an iced airfoil, as compared to the clean airfoil, for a lift coefficient of 0.35.

$$\Delta\alpha = \alpha_{iced(C_l=0.35)} - \alpha_{clean(C_l=0.35)} \tag{2.12}$$

The increment in the angle of attack varied almost linearly with the maximum lift coefficient. However, it must be noted that $\Delta\alpha$ ranges from -0.05 to 0.05. In flight, identification of such small changes in angle of attack would be very difficult. Hence, although the trend is promising this relationship may not be very helpful in limit prediction.

Figure 2.5 demonstrates the relationship between the maximum lift coefficients with the change in drag due to icing. The data shown in this plot was obtained from 2-D wind tunnel experiments performed on the NLF0414F airfoil in the University of Illinois.[14] The drag corresponded to a lift coefficient of 0.2.

$$\Delta C_d = C_{d,iced(C_l=0.2)} - C_{d,clean(C_l=0.2)} \tag{2.13}$$

This figure indicates that there is a significant drag rise due to ice accretion even when the airfoil is producing low lift. The almost linear relationship between this drag rise and the maximum lift coefficient shows potential for use in stall prediction. Previous flight test data were also analyzed to determine the dependence of aerodynamic parameters on icing. It should be noted that for an airfoil there is no drag due to non-lifting surfaces and that the lift-drag relationship is strong. However, when the aerodynamics of a full aircraft is considered this may not be true. In order to investigate this, data from flight tests[10,11] were analyzed. Figure 2.6 is a plot of C_{D0} as a function of η_{ice}. The zero-lift drag coefficient was interpolated from Twin Otter icing flight test data published by Mikkelson et al.[10] and Ranaudo et al.[11] This plot shows an almost linear correlation

between the icing parameter η_{ice} and the zero-lift drag coefficient. Unfortunately no $C_{L,max}$ data were available, but the trend with η_{ice} suggests a relationship with $C_{L,max}$ may exist as well.

The key was to develop the capability to determine the stall limits based on data available at low angle of attack. The most promising trend was found in the comparison of ΔC_l at constant angle of attack with iced airfoil C_{lmax} data. Figure 2.7 illustrates this trend at an angle of attack of 4^o. In this figure, C_{lmax} was plotted against ΔC_l. ΔC_l was the difference between the lift generated by an airfoil with simulated ice and the clean airfoil at the specified angle of attack.

$$\Delta C_{l(\alpha=4^o)} = C_{l,iced(\alpha=4^o)} - C_{l,clean(\alpha=4^o)} \tag{2.14}$$

It is evident from the figure that there is an almost linear relationship between C_{lmax} and ΔC_l. This linearity existed for angles of attack ranging from 0^o to 11^o. Hence, if the stall angle could be approximated as a function of C_{lmax}, it would be possible to use the value of ΔC_l to calculate the stall limit at a given angle of attack. Figure 2.7 thus illustrates the possibility of successfully predicting stall in icing conditions from the reduction in lift at low angles of attack.

It was assumed that the trends found in the 2-D parameters were applicable to the 3-D ones. Hence, a method was developed to estimate stall in icing conditions for a full

aircraft. A linear relationship was assumed between the stall angle of attack and the C_{Lmax}.

$$\alpha_{\text{limit}} = \frac{C_{L\max} + C_{L_0}}{C_{L_\alpha}} \tag{2.15}$$

Then, it was possible to define the limit value of the angle of attack as shown in Eq. (2.16).

$$\alpha_{\text{limit}} = f(\Delta C_L) \tag{2.16}$$

Therefore, this model could be used to approximate the limit boundary of the angle of attack when the aircraft is trimmed as low as 0^o. Simulations have shown that the algorithm was successful in estimating the stall angle from a trimmed state at different icing conditions.

2.4 Dynamic Trim

An attempt was made to apply the idea of dynamic trim, described in Section 1.3.3, to the predicting envelope protection scheme. It was assumed that for the longitudinal system the following conditions held during dynamic trim.

$$\dot{\alpha} = \dot{q} = 0 \tag{2.17}$$

Five minute FDC simulations were run in order to determine the time taken for the Twin Otter to reach a state of dynamic trim. The simulations were run using an initial velocity of 155 kts. The initial altitude was set at 7545 ft. The results from these simulations are shown in Fig. 2.8. This figure shows the angle of attack response to several step elevator inputs. The steps ranged from 1^o to 6^o. As seen from the figure, the angle of attack response was oscillatory and the transient values were higher than the steady state ones. In order to determine the time at which the aircraft reached dynamic trim, the rate of change of the angle of attack and the pitch rate were plotted as shown in Figs. 2.9 & 2.10. These plots demonstrated that the Twin Otter took about 200 seconds on average to reach dynamic trim. This time was much higher than the values quoted by Horn et al. for the V22, which were around 2-3 seconds.[19] In order to verify the FDC results, analysis was carried out using linearized equations of motion. The solutions from these are shown in Fig. 2.11. Figure 2.11 shows the solution for the angle of attack response versus time. These solutions corresponded to step elevator inputs from 1^o to 6^o. This plot showed similar results to the plots from Fig 2.8. That is, the response was oscillatory, the transients seemed to have higher values than the steady state and that the oscillations continued till about 200 seconds. Amplitude differences between FDC results and linearized calculations may be attributed to approximations such as $C_{L_{\dot{\alpha}}} = 0$, $C_{m_{\dot{\alpha}}} = 0$, and to the fact that the effect of thrust on the force and moment coefficients were ignored.

It should also be noted that from the FDC simulations it was found that the state of the aircraft at dynamic trim conditions was very similar to the state reached at a regular trim condition where even the rates of change of the slow states as given in Section 1.3.3 go to

0. It was thus concluded that the Twin Otter may be too lightly damped to have a dynamic trim state and that the dynamic trim method was not applicable to the Twin Otter.

2.5 Open-Loop IAEP

The purpose of the open loop IAEP was to warn the pilot of an impending stall and potential loss of control in sufficient time to correct the situation. Thus it was necessary to develop a method to predict limit violations in the future using the available sensor information. Although the formulation proposed by Horn et al.[19] utilized the same concept, the idea of dynamic trim as described by Horn et al. was not applicable to the icing aircraft as discussed in Section 2.4. Thus the neural net-based method developed for the rotorcraft could not be implemented in the current research. Instead an alternative scheme was used. In the method developed here the equations of motion were integrated forward in time starting at the current aircraft state to predict the value of the critical parameter in the future. The future values of the critical parameter were then checked for limit violation.

The equations of motion were functions of the state and control vectors and also the icing parameter as shown in eq. (2.18).

$$\dot{\overline{x}} = g(\overline{x}, \overline{u}, \eta) \tag{2.18}$$

The state vector as given included the inertial velocity components, the components of the inertial rotational velocity and the Euler angles.

$$\bar{x} = [u \quad v \quad w \quad p \quad q \quad r \quad \theta \quad \phi \quad \psi] \tag{2.19}$$

The control vector included the deflections of the control surfaces and the power.

$$\bar{u} = [\delta_e \quad \delta_a \quad \delta_r \quad \delta_{power}] \tag{2.20}$$

The forces and moments were calculated using a nonlinear model, which incorporated the effect of the ice accretion quantitatively through η. The contribution of the icing parameter η thus occurred through the calculation of the forces and moments, which included the effect of the ice accretion, modeled using η. The nonlinear model was based on data obtained from wind tunnel experiments using a scaled model of the Twin Otter aircraft as described in Section 31.[34]

The equations of motion of a rigid body are modeled through the nonlinear ordinary differential equations shown in equations (2.21) through (2.31). These equations were derived from applying the flat earth approximation where the effect of the rotation and the curvature of the earth were neglected. Details of the derivation can be found in Etkin.[35]

Force equations:

$$\dot{u} = \frac{1}{m}(X - mg\sin\theta - m(qw - rv)) \tag{2.21}$$

$$\dot{v} = \frac{1}{m}(Y + mg\cos\theta\sin\phi - m(ru - pw)) \tag{2.22}$$

$$\dot{w} = \frac{1}{m}(Z + mg\cos\theta\cos\phi - m(pv - qu)) \tag{2.23}$$

Moment equations:

$$I_x\dot{p} - I_{zx}\dot{r} = L + (I_y - I_z)qr + I_{zx}pq \tag{2.24}$$

$$I_y\dot{q} = M + I_{zx}(r^2 - p^2) + (I_z - I_x)rp \tag{2.25}$$

$$I_z\dot{r} - I_{zx}\dot{p} = N + I_{zx}qr) + (I_x - I_y)pq \tag{2.26}$$

Kinematics:

$$\dot{\phi} = p + q\sin\phi\tan\theta + r\cos\phi\tan\theta \tag{2.27}$$

$$\dot{\theta} = q\cos\phi - r\sin\phi \tag{2.28}$$

$$\dot{\psi} = (q\sin\phi + r\cos\phi)\sec\theta \tag{2.29}$$

Position equations:

$$L_{VB} = \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \sin\phi\sin\theta\sin\psi\cos\phi\cos\psi & \sin\phi\cos\theta \\ \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi & \cos\phi\cos\theta \end{bmatrix} \tag{2.30}$$

$$\begin{bmatrix} \dot{x}_E \\ \dot{y}_E \\ \dot{z}_E \end{bmatrix} = L_{VB}\begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{2.31}$$

The resultant aerodynamic forces X, Y and Z in equations (2.21) through (2.23) were functions of the state vector, and control vector, and the icing parameter.

2.5.1 The Open-Loop IAEP in Matlab

The predictive open-loop IAEP algorithm was implemented using Matlab. The algorithm consisted of a main level batch file "solveom.m" and three subroutines, "ic.m," "solver.m," and "DE.m." "ic.m" was the script used to extract the initial conditions to be

used to solve the oridinary differential equations shown above in Eqs. (2.21) - (2.31). "solver.m" was used to arrange the initial conditions in the required format while "DE.m" was used to solve the equations of motion. The script for these codes was included in Appendix A.

2.5.2 ODE Solver

The ordinary differential equation solver used in the IAEP algorithm was a built-in matlab ODE solver, "ode45." "ode45" is based on an explicit Runge-Kutta (4,5) formula, the Dormand-Prince pair. It is a *one-step* solver - in computing $y(t_n)$, it needs only the solution at the immediately preceding time, $y(t_{n-1})$.[36] This solver is a medium accuracy solver used for non-stiff problems.

2.6 Linearized Equations of Motion

In order to complete the IAEP formulation it was necessary to device a method to calculate the elevator deflection that corresponded to stall, i.e. the value of the elevator deflection that would cause the aircraft to stall. To accomplish this at first an iterative method was used. That is, once stall was detected, the full equations of motion were solved for different values of the elevator until the maximum safe deflection value was found. This method although effective was time consuming and hence the Linearized equations of motion were used to accelerate the process. In this method the transfer function, $\frac{\alpha(s)}{\delta_e(s)}$, derived from taking the Laplace Transform of the longitudinal small perturbation equations of motion[37], was used.

$$\frac{\alpha(s)}{\delta_e(s)} = \frac{\begin{vmatrix} s - X_u - X_{T_u} & X_{\delta_e} & g\cos\theta \\ -Z_u & Z_{\delta_e} & \{-(Z_q + U)s + g\sin\theta\} \\ -(M_u + M_{T_u}) & M_{\delta_e} & (s^2 - M_q s) \end{vmatrix}}{\begin{vmatrix} s - X_u - X_{T_u} & -X_\alpha & g\cos\theta \\ -Z_u & \{s(U - Z_{\dot{\alpha}}) - Z_\alpha\} & \{-(Z_q + U)s + g\sin\theta\} \\ -(M_u + M_{T_u}) & -\{M_{\dot{\alpha}} s + M_\alpha + M_{T_\alpha}\} & s^2 - M_q s \end{vmatrix}} = \frac{N_\alpha}{D} \qquad (2.32)^{37}$$

The transfer function given in Eq. (2.32) is the ratio of the Laplace transform of the motion variable $\alpha(s)$ to the Laplace transform of the elevator input $\delta_e(s)$.[37] However, for the IAEP problem, where an estimate of the angle of attack limit and the angle of attack at given point in time were known, the unknown parameter was the elevator input which would cause the angle of attack of the aircraft to jump from its current value to the limit value. Hence, in order to determine the elevator deflection, it was assumed that there was a step change in the angle of attack and a reciprocal of Eq. (2.32) was taken.

$$\delta_e(s) = \frac{D}{sN_\alpha} \qquad (2.33)$$

The inverse Laplace transform of Eq. (2.33) was then found using de Hoog et al's quotient difference method with accelerated convergence for the continued fraction expansion.[38] The script files used to calculate the elevator deflections were included in the Appendix.

2.7 Modification of the Flight Dynamics Code

The Flight Dynamics Code (FDC)[23] for Matlab and Simulink was used to simulate different flight scenarios in icing conditions. These simulations were used to investigate the effects of icing on aircraft performance, identify the key requirements of an envelope protection scheme and to test the iced aircraft envelope protection system. Since the simulations were an integral part of the research, this section gives detailed descriptions of the modifications made to the FDC in order to create realistic iced aircraft simulations. Details of the FDC code and modifications made to it for the simulation for the SIS project can be found in Rauw[23] and Pokhariyal,[24] respectively.

2.7.1 Non-linear Aerodynamic Model

As mentioned in Sec. 2.1 the functions modeling the non-linear aerodynamics of the aircraft were dependent on the angle of attack, elevator deflection and the icing condition. Since FDC created time dependent solutions, the forces and moments of the iced aircraft had to be updated at every time step. In order to do this, the clean aerodynamic data were incorporated in a block in Simulink (apilotnl/Beaver Dynamics/AC/ae/aero) to compute the clean coefficients that are needed to calculate the iced coefficients as shown in Eq. (2.5). Modifications were then made to the Matlab file "icesev.m"[24] to calculate the aircraft force and moment coefficients using the non-linear aerodynamics model. Figure 2.12 represents the incorporation of the clean non-linear aerodynamics into the Simulink block, while the modified icesev.m script is included in the Appendix.

2.7.2 Implementing the IAEP in the FDC

Several additions were made to the original Simulink FDC structure in order to incorporate the IAEP in to it. Most of the IAEP functions were completed in the Matlab file "icesev.m." The files, "solverlin.m" and "solvernonlin.m" are called by the "icesev.m" file in order to determine whether the angle of attack limit is exceeded during a simulation and to calculate the safe limit of the control deflection. The "solverlin.m" file is used if a linear IAEP solution is desired while the "solvernonlin.m" is used if a nonlinear solution is needed.

The Simulink block "apilot" was modified in order to enforce a hard limit on the elevator deflections as shown in Fig. 2.13 All elevator inputs are passed through the function "IAEP1.m" in order to determine whether the safe limit is exceeded. If the limit is violated, the limit value of the elevator is set as the elevator value thus instituting a hard limit on the elevator. It must be noted here that the hard limits were used in simulations only to evaluate the system and that a decision is yet to be made on whether hard or soft limits are going to be used in the final IAEP.

Modifications were also made to subsystems of the "apilot" block. These subsystems are shown in Figs. 2.14 and 2.15. The modifications were labeled as IAEP MOD. Most of these modifications were necessary in order to provide the required input to the "icesev.m" for estimating possible limit violations and the safe limits of the control deflection.

2.8 Figures

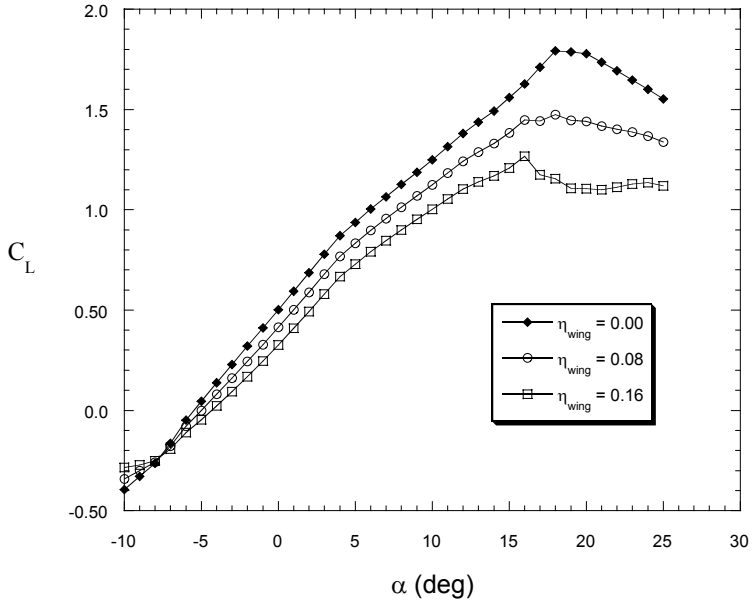

Fig. 2.1. Coefficient of Lift for Different Icing Conditions

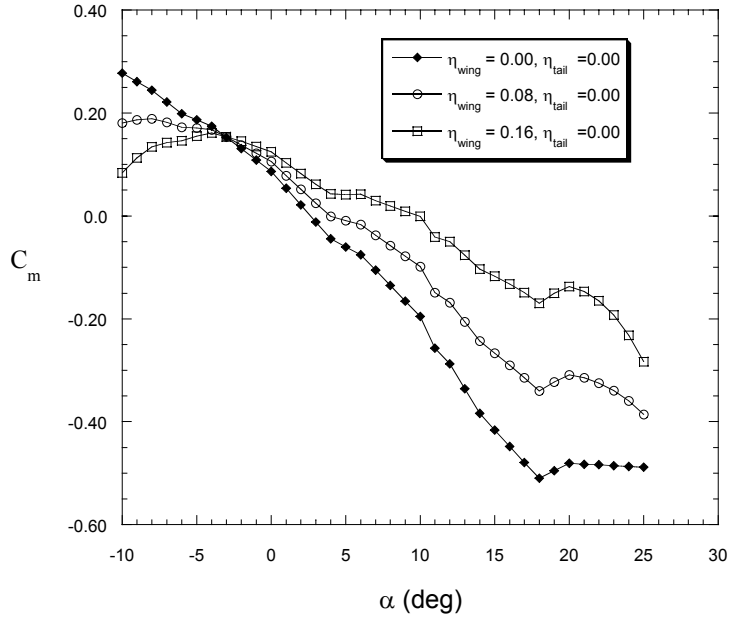

Fig. 2.2. Coefficient of Pitching Moment as a Function of the Icing Condition and Angle of Attack

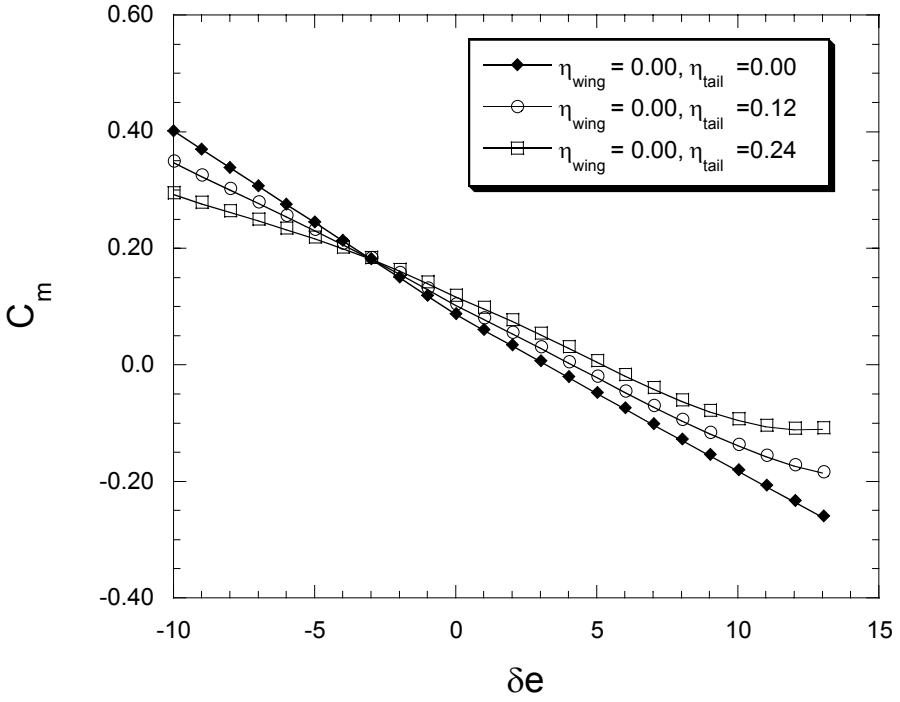

Fig. 2.3. Coefficient of Pitching Moment as a Function of Elevator Deflection and the Icing Conditions

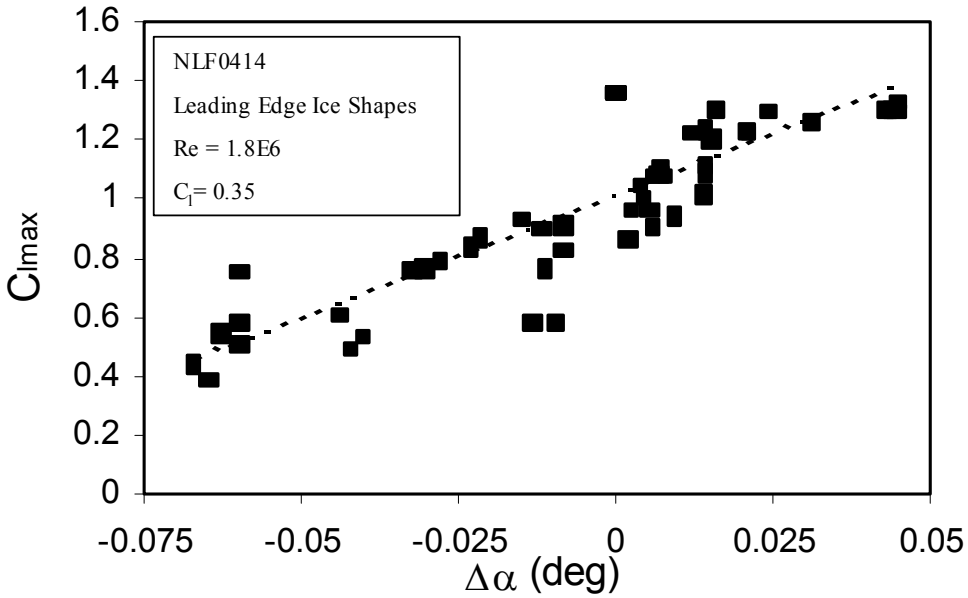

Fig. 2.4. Comparison of Maximum Lift Coefficient with the Change in Required Angle of Attack for Fixed Lift Coefficient

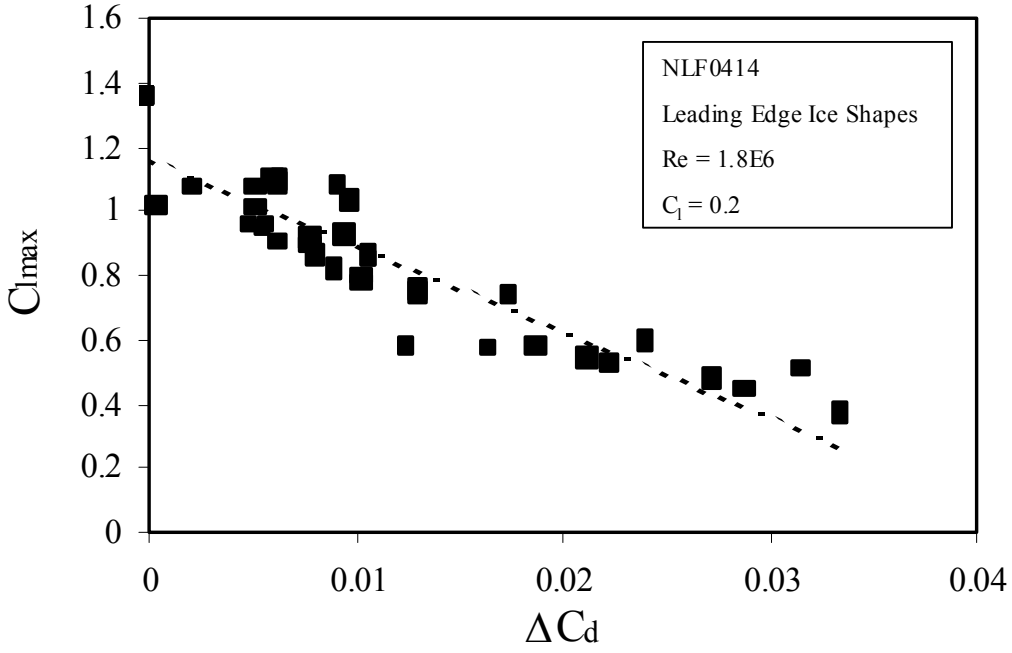

Fig. 2.5. Variation of Maximum Lift Coefficient with Drag Rise

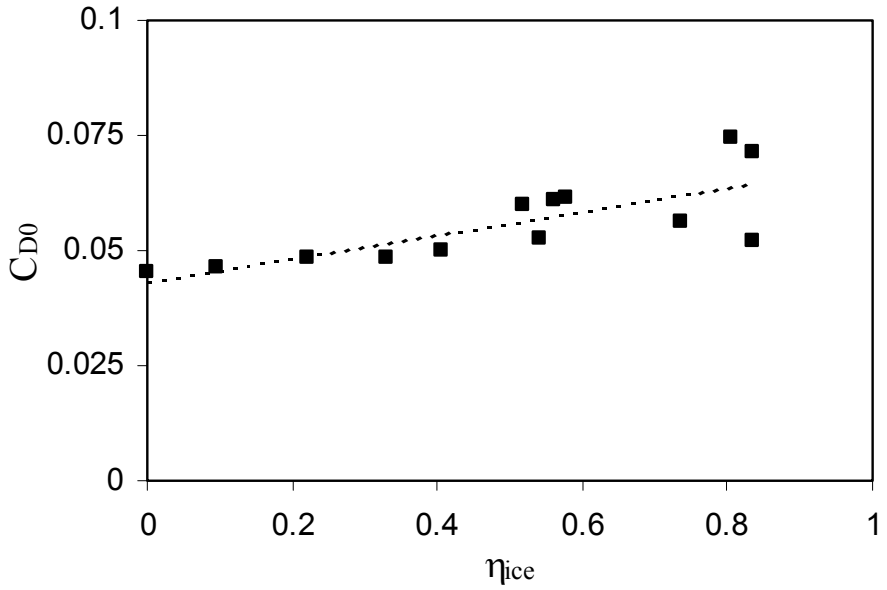

Fig. 2.6. Comparison of Zero Lift Drag Coefficient with η_{ice}

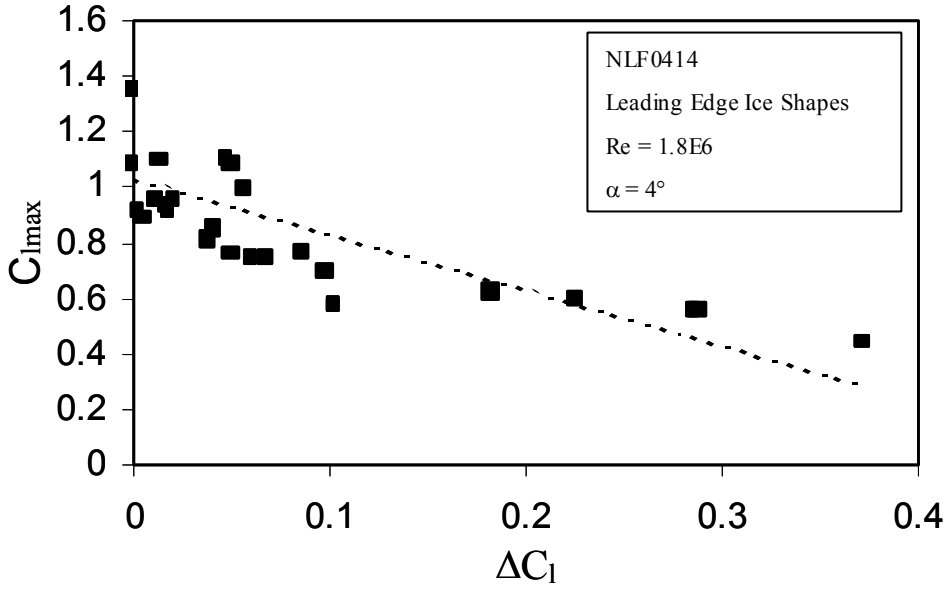

Fig. 2.7. Variation in Maximum Lift with Change in Lift at Fixed Angle of Attack

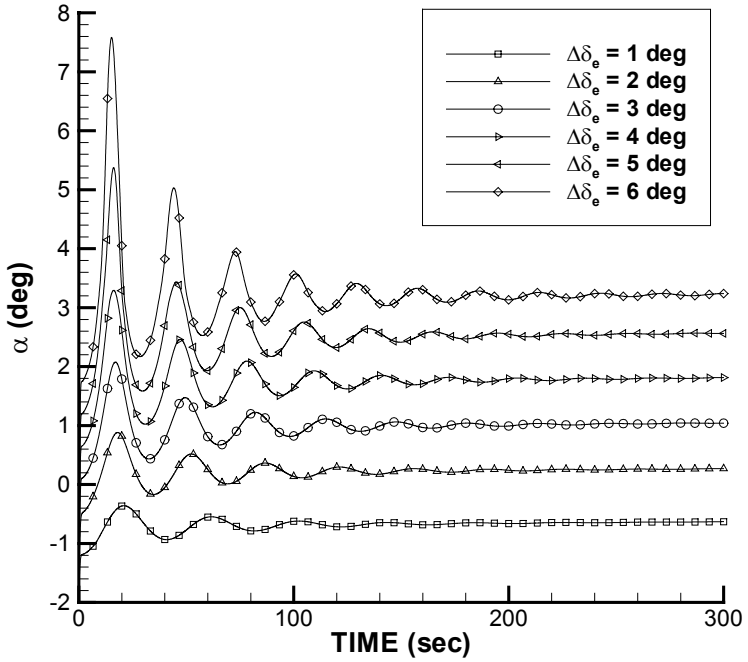

Fig. 2.8. Angle of Attack Response Due to Step Elevator Inputs from FDC Simulations

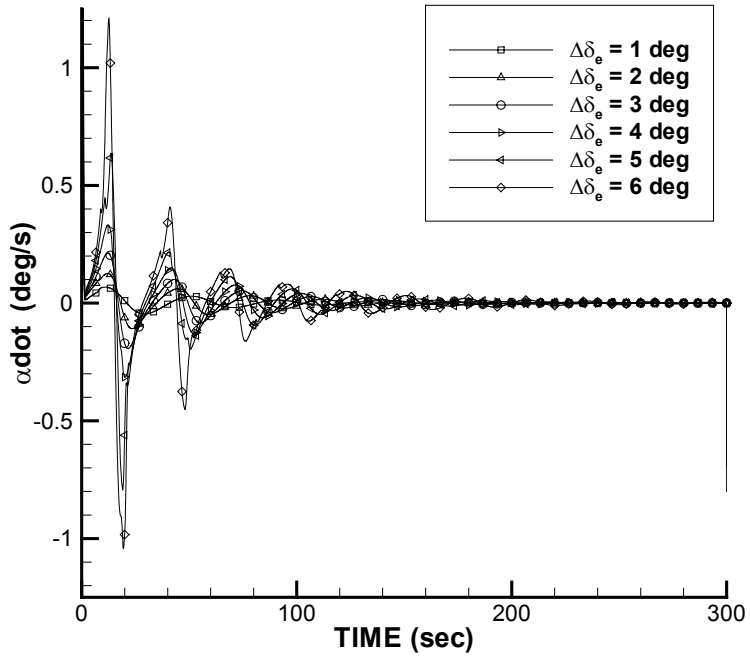

Fig. 2.9. Time History of Rate of Change of the Angle of Attack

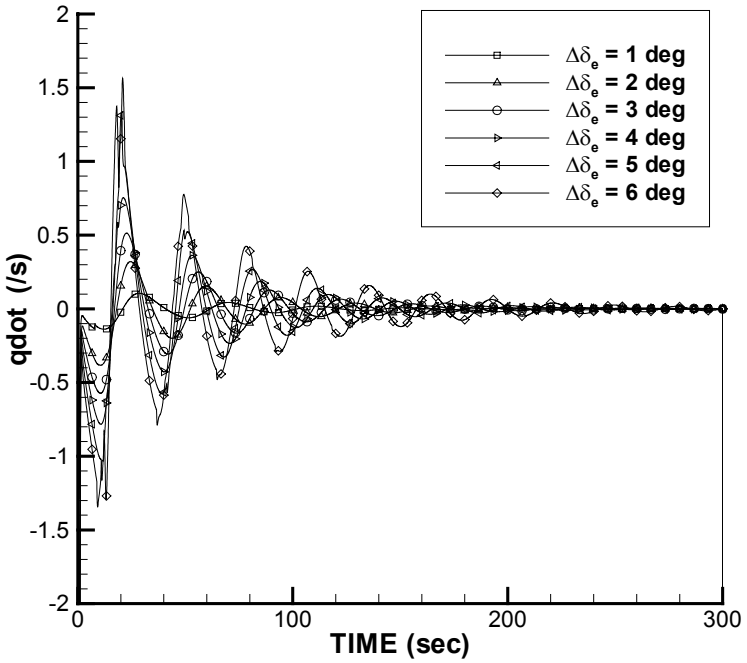

Fig. 2.10. Time History of Rate of Change of the Pitch Rate

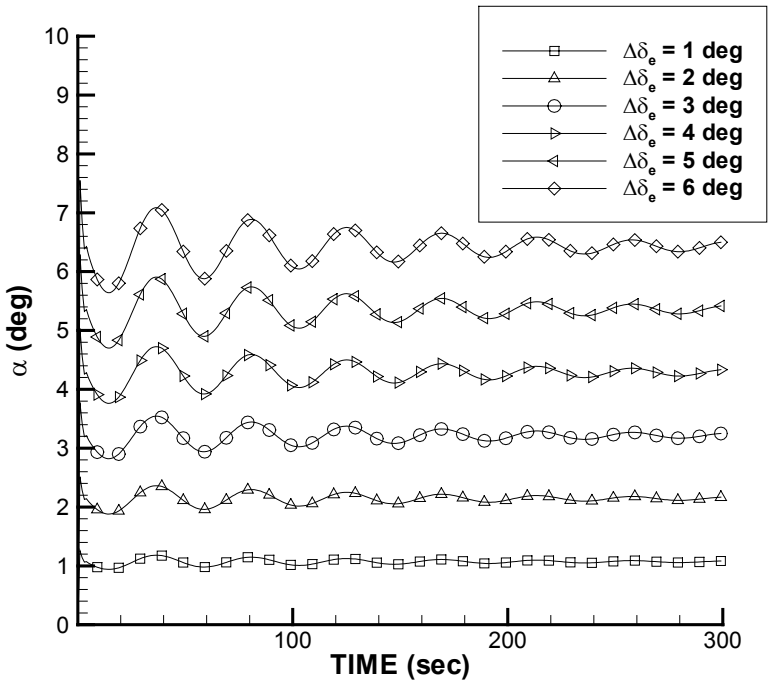

Fig. 2.11. Angle of Attack Response Computed using Linearized Equations of Motion

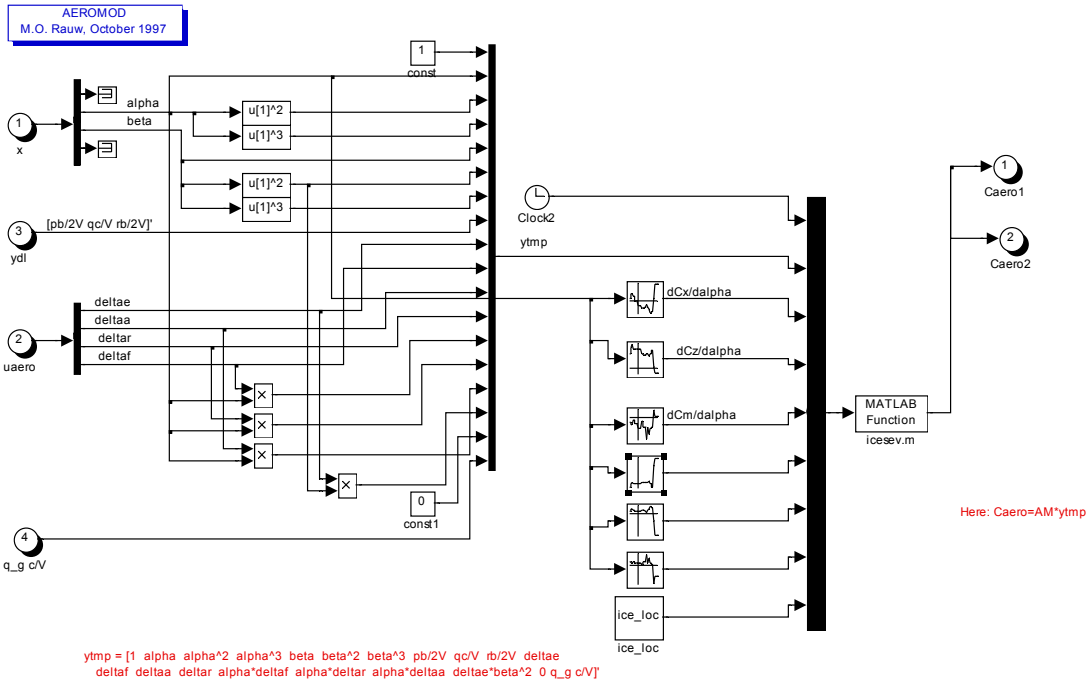

Fig. 2.12. Incorporation of Non-linear Aerodynamic Model in Simulink

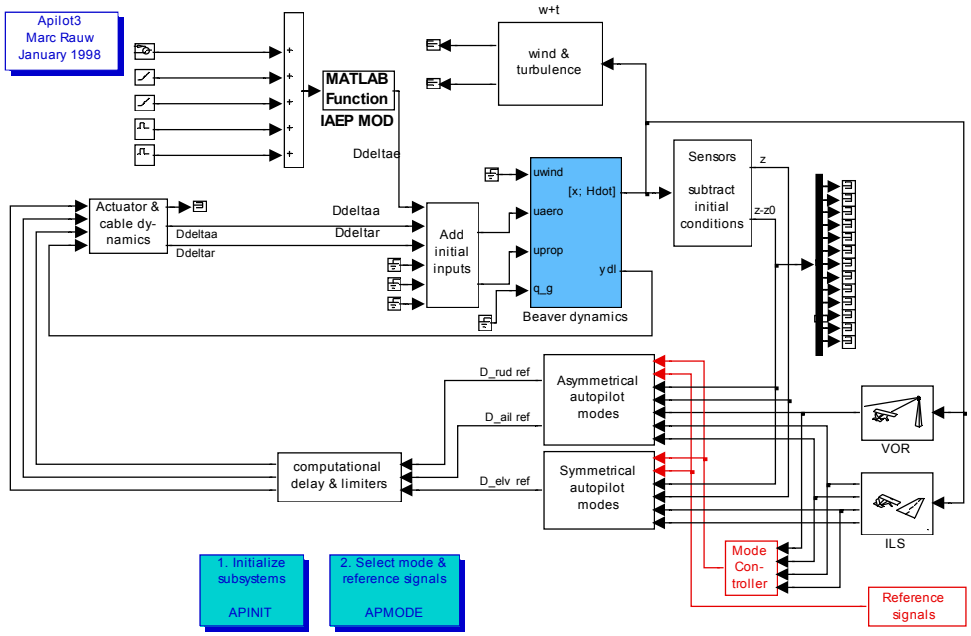

Fig. 2.13. Open-Loop Configuration of Simulink Structure Apilot with IAEP Modications

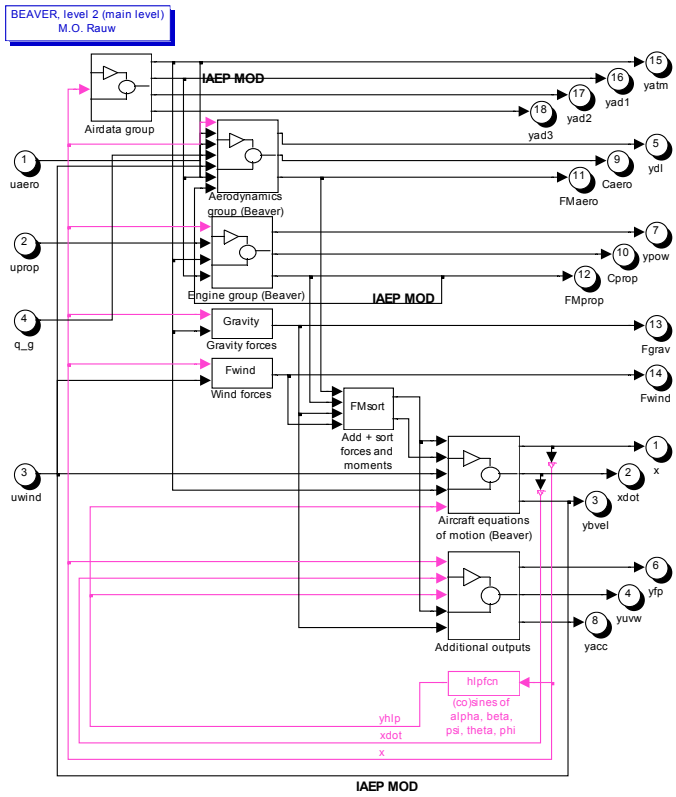

Fig. 2.14. IAEP Modified Aircraft Dynamics Structure

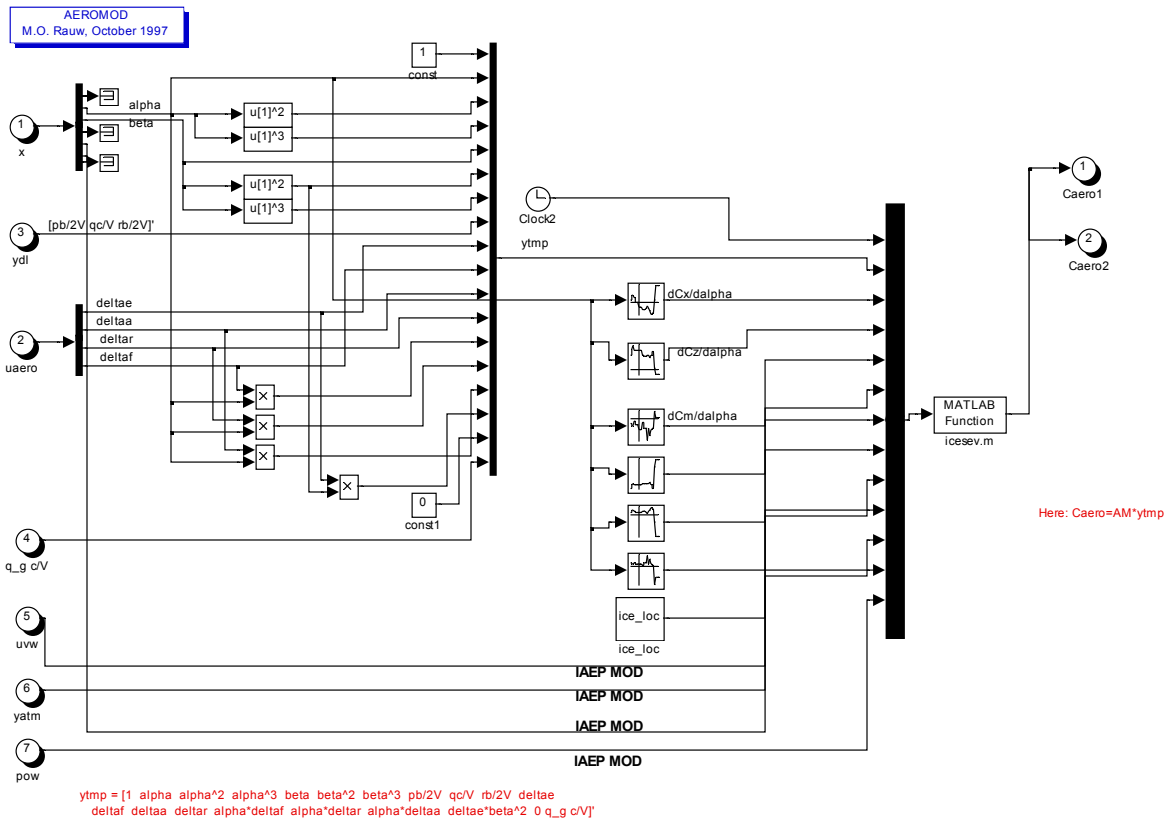

Fig. 2.15. Modified Block AEROMOD

3 RESULTS AND DISCUSSION

This chapter presents the results obtained using FDC simulations, run mainly to validate the estimative and predictive Iced Aircraft Envelope Protection (IAEP) schemes. All the simulation results presented in this chapter were gathered using Twin Otter aerodynamic models as discussed in Section 2.7.1. The validation data were presented toward the beginning of the chapter followed by results from a study used to determine the integration time for the predictive IAEP. Then, the method used to determine the safe boundaries of the control deflections was presented. Discussions pertaining to simulation results gathered using the IAEP at different icing scenarios were also included. Thus, the main purpose of this chapter was to present the results from simulations run to determine the potential of the IAEP.

3.1 Validation

3.1.1 Estimative IAEP

The estimative IAEP discussed in Section 2.3 was validated using data from FDC simulations at different icing conditions. These icing conditions corresponded to wing only icing with constant η values of 0.01, 0.05, 0.08, 0.1 and 0.2. Simulations were also run at clean conditions in order to obtain the clean flight data needed by the estimative algorithm. The simulations were run using initial velocities of 97 kts and 136 kts. The manifold pressure was set at 20" Hg and the engine RPM was at a 1000. The simulations were started at an altitude of 1640 ft. The icing parameter was left constant as mentioned above and ramped elevator inputs were issued in order to stall the aircraft. A sample simulation is shown in Figs. 3.1. This particular simulation was run at an η value of 0.05. Figures 3.1-3.3 show the angle of attack response, the elevator command issued, and the time history of the altitude respectively. As seen from Fig. 3.2 the elevator was initially constant, and then decreased in decrements of 1^{o}. As a result of the decrease in the elevator, the aircraft pitched up until it stalled at an angle of attack of 9.99^{o}. The effect of the stall was seen even more clearly in Fig. 3.3 where the altitude drops sharply at the time corresponding to stall.

For the validation, the lift data from the iced and the clean simulations were used to calculate the change in lift due to icing. These changes were then used to estimate the maximum lift coefficient of the iced aircraft and consequently the stall angle of attack as shown in Eqs. (2.14) - (2.16). The script of the codes “extractdcl.m,” “datafrom.m,” and

"stall.m" used to accomplish this were included in the Appendix. The validation results were presented in a graphical format in Fig. 3.4 This figure shows the stall angles of attack as predicted by the estimative IAEP and the stall angles of attack as extracted from FDC simulations using the non-linear model for different icing conditions. As seen from the plot there was some scatter at each icing condition. This was attributed to the fact that the estimates were calculated at different angles of attack. Since the linear relationship between the stall angle of attack and ΔC_L varied for the different angles of attack, it was found that the error range was $\pm 1^o$.

3.1.2 Predictive IAEP

The solution code written using the equations presented in Section 2.5 was validated against FDC simulations and flight test data as shown in Figs. 3.5-3.7.

Figure 3.5 shows the aircraft angle of attack versus time for a 30 second FDC simulation. The Twin Otter simulation was started at an initial velocity of 136 knots and altitude of 7545 ft. The η was set at a constant value of 0.1. A step elevator input of 3^o was issued at 5 seconds. As seen from the plot, due to the elevator input, the α goes beyond its stall limit of 10.5^o.

The open loop IAEP prediction shown with symbols on Fig. 3.5 was initiated at 12 seconds into the flight. The IAEP solution was run for 5 seconds into the future. As seen from the plot, the IAEP solution corresponds quite well with the FDC result especially for the first 3 seconds. The slight divergence observed toward the 4^{th} second was

attributed to the fact that the control parameter, $\bar{u}$ as given by eq. (2.20) was kept constant during the solution. Hence, unlike the FDC simulation where the power changes due to the change in altitude, in the IAEP solution the power term remained constant. However, the difference between the FDC and IAEP solutions were within $1/2^{o}$ and it was found that the IAEP solutions always over predicted the FDC solutions. Therefore, the IAEP provides a safe estimate of the future values of the critical parameter.

The IAEP solutions were also validated against Twin Otter flight test data. The flight data used in this validation was obtained from flight tests completed in February 2002 as a part of the Smart Icing Systems research project. Figure 3.6 is a plot of the θ response to the δ_e inputs plotted in Fig. 3.7 for flight test number 020213f, a morning flight in clean conditions.[39] In the time frame being studied here, a 0.25g doublet was initiated at a time of 4 seconds.

For the validation, the IAEP was run starting at 4 seconds, 5.2 seconds and 8 seconds as indicated by the arrows in Fig. 3.6. The symbols in Fig. 3.6 correspond to the IAEP solutions. As seen in this figure, the IAEP solution initiated at 4 seconds performed well until 4.8 seconds but beyond that, the IAEP predicted a continuous increase in the pitch while there was a drastic pitch down in the flight test data. This was due to the fact that, a step elevator was issued at 4.8 seconds in the flight test. This step was not modeled in the IAEP as illustrated in Fig. 3.6 where at 4.8 seconds there was a step increase in the elevator for the flight test but the IAEP elevator value remained constant at -3.8^{o}.

Similarly, the IAEP solution initiated at 5.2 seconds compares well to the flight test data till the 6^{th} second when another step elevator was issued in the flight test while the IAEP elevator value remained at a constant value of 2.8^{o}.

The last IAEP solution being discussed here was initiated at 8 seconds. As seen from Fig. 3.7 the flight elevator remained fairly constant during this time. As a result the IAEP solution matched almost exactly the pitch response of the Twin Otter as seen in Fig. 3.6. However, toward the 11^{th} second the IAEP began to diverge. This was attributed to the fact that at this point, there was a slight increase in the power by the pilot during flight, which was not modeled in the IAEP.

The Twin Otter flight test data and the IAEP solutions compared well when the control deflections remained constant. This suggested that the IAEP solutions could be used to estimate the future time at which the Twin Otter would reach its stall angle of attack.

3.2 Integration Time for IAEP Predictive Solutions

Once a method was formulated to predict the state of the critical parameter, some analysis was done to determine the lead-time needed for the aircraft dynamics to adjust to corrective control inputs and avoid stall. FDC results were obtained for situations where the iced aircraft with η values of 0.1, 0.15 or 0.3, was initially trimmed at an altitude of 1640 ft and an elevator input was issued. The initial velocities for the simulations ranged from 97 knots to 116.6 knots. The elevator inputs included instantaneous step increases

of 3^o and 6^o, and gradual ramped increases of 0.06^o/sec, 0.08^o/sec and 0.2^o/sec. The IAEP was run with integration times of 1 second, 3 seconds, 5 seconds and 10 seconds.

Figure 3.8 is a plot of a situation where a ramped elevator input of 0.08^o/sec was issued at 0 seconds. The angle of attack increased at a rate of 0.2^o/sec until it reached its limit of 6.8^o at 12.1 seconds. The initial trim velocity was 97.2 knots. The icing parameter η was constant at a value of 0.3. Figure 3.8 shows the solutions from five different simulations. The solid curve without any symbols represents the angle of attack response from the FDC simulation without any IAEP. While the curves with the symbols illustrate the responses from simulations using IAEP with integration times of 1 second, 3 seconds, 5 seconds and 10 seconds. The 1, 3, 5 and 10 second IAEP corresponds to results obtained from simulations where the angle of attack response was predicted 1, 3, 5 and 10 seconds in the future, respectively, for each time step in order to determine whether the stall angle of attack was exceeded within the corresponding time span. If the critical parameter limit were exceeded within the solution time, the elevator was decreased by half a degree and left constant at that value when the IAEP was initiated. The initial points of the IAEP solutions were indicated on Fig. 3.8 by the arrows. From this plot it is seen that although an integration time of 1, 3, 5 and 10 seconds were used, the modified elevator was not implemented exactly 1, 3, 5 and 10 seconds before the stall limit is reached in the simulation without IAEP. This was because step inputs were used in the IAEP to predict the future values of the angle of attack. As a result, the peak angles of attack in the predictions were reached faster than in the simulation without the IAEP.

As seen from Fig. 3.8, when the modified elevator was issued, the angle of attack peaked at a value lower than the stall angle of attack. The same phenomenon occurred for all the integration times that were studied. Thus, even with a lead-time of only 1 second, the aircraft dynamics adjusted to the modified control deflection to avoid stall. However, the increase in the elevator input studied in Fig. 3.8 was a rather mild one and hence go-arounds were simulated using step elevator inputs to verify that a 1 second prediction time was sufficient to avoid stall.

Figure 3.9 is a summary plot of the results obtained from one of the go-around scenarios mentioned above. In the original FDC simulation for this case, the initial velocity was 116.6 knots. The icing parameter η was at a constant value of 0.05. The go-around was initiated with a step elevator input of 3^o at 0 seconds. As a result the angle of attack increased and exceeded the stall limit of 10^o at a time of 8.8 seconds. However, the angle of attack continued to increase instead of experiencing a sharp descent as would be expected after stall. This may be due to the fact that the momentum of the aircraft is so high at that point that it takes time for the dynamics to be overpowered by the loss in lift due to stall. It was also found that the angle of attack response was oscillatory after the application of the modified elevator. This may be attributed to the fact that the aircraft is very lightly damped and abrupt changes in the control inputs result in oscillatory responses.

Included in the plot were the modified responses obtained from applying the open loop IAEP to the situation described above. As mentioned above, 4 different integration times

were used for the IAEP. In Fig. 3.9 the curve labeled IAEP 1 sec corresponds to the response of the aircraft when stall was predicted with a 1 second lead-time and the elevator input was modified to a new value to avoid exceeding α_{limit}. Also shown in Fig. 3.9 are the modified responses from the 3 second, 5 seconds and 10 seconds lead-time cases.

From the results presented in Figs. 3.5-3.9, it was evident that the open loop IAEP was successful in avoiding stall. The fact that the IAEP was successful in avoiding stall in the severe icing conditions and in spite of the high angular rates induced by a step elevator input, indicated that it would be an effective method to use for envelope protection of an iced aircraft.

Another important conclusion that can be drawn from this study is that the integration time to be used for the IAEP does not seem to be dependent on the aircraft dynamics or the proximity of the aircraft angle of attack to its limit value. The warning lead-time necessary to avoid stall is expected to depend primarily on pilot reaction time and other human factors issues. Thus determining the appropriate integration time, or the time IAEP looks forward in time to predict potential stall, is a topic for a future human factors study.

3.3 The Maximum Control Deflection

The linearized equations of motion, discussed in Section 2.6, were used to calculate the elevator deflections required to avoid the stall. Figures 3.10 - 3.15 illustrate the

effectiveness of this method in estimating the safe control limits. The simulations shown in these figures were started from trimmed conditions with an initial velocity of 116 knots and an altitude of 3280 ft. The engine RPM was kept constant at 2200 while the icing parameter values used were 0.01, 0.05 and 0.1. The dashed lines represented the stall angle limit while the solid line with symbols were the angle of attack responses from step inputs of the maximum allowable control deflections as estimated from the solution of the linearized equations of motion. As seen from these figures the angle of attack increased to within a degree of the stall angle limit but did not reach it for any of the three icing conditions. This may be attributed to the fact that at high angles of attack the aerodynamic relationships become highly nonlinear and hence the linearized equations do not represent the behavior of the aircraft exactly. However, several simulations were run at different icing conditions to evaluate whether the estimated control deflections allowed the angle of attack to exceed the stall limit and also to discern whether the estimates were too conservative. Figures 3.10, 3.12. and 3.14. show only three of these simulations but the results were similar in all the simulations where the angle of attack response never exceeded the stall limit. These results indicate that the elevator limits obtained using the solutions of the linearized equations of motion could be used in an IAEP scheme for enforcing soft or hard limits on the control deflection.

3.4 Effectiveness of the IAEP Scheme

Using the results from simulations shown in the previous sections, the IAEP scheme was outlined as follows:

1. The longitudinal envelope is calculated in terms of the angle of attack limit using the method outlined in Section 2.3.
2. Elevator deflection limits are calculated, such that $\alpha \simeq \alpha_{\text{limit}}$, at every time step from the advent of flight. The results shown here use the method outlined in Section 2.6 to predict $\delta_{e_{\text{limit}}}$.
3. Pilot inputs a change in the elevator deflection causing the aircraft to leave its trim condition.
4. If $\left|\delta_{e_{\text{input}}}\right| < \left|\delta_{e_{\text{limit}}}\right|$ no action is taken.
5. If $\left|\delta_{e_{\text{input}}}\right| > \left|\delta_{e_{\text{limit}}}\right|$ then,
    - The elevator deflection is restricted to $\delta_{e_{\text{limit}}}$
    - $\delta_e = \delta_{e_{\text{limit}}}$ was used in simulations

This IAEP scheme was implemented in FDC and simulations were run using both step elevator changes and ramped elevator changes as "pilot" inputs. The results presented in this section were obtained from FDC simulations without any IAEP, and with simulations obtained using the IAEP scheme outlined above.

The first set of simulations, the results from which are shown in Figs. 3.16 - 3.21, were run using an initial trim velocity of 116.6 knots, starting at an altitude of 10,000 ft with a constant engine RPM of 2200. The simulations were run at η values of 0.01, 0.05 and 0.1. In the simulations, a step elevator input of –10 ^{o} was issued at a time of 2 seconds.

In the figures, the lines with the square symbols show the angle of attack response and elevator commands from simulations run without any IAEP, the lines with triangle symbols represent results from simulations with the IAEP. As seen from Figs. 3.16, 3.18 and 3.20 the angle of attack responses for the simulations with IAEP were below the stall limits indicated by the dashed lines. As seen from Fig. 3.17, 3.19 and 3.21 in the simulations without the IAEP operation, the aircraft stalls due to the 10^o step elevator input. However, with the IAEP in operation the elevator was limited to a safe value of -5.65^o for an η of 0.01, -5.3^o for and η of 0.05 and -4^o for an η of 0.1. As a result, stall was avoided in all three icing conditions.

Simulations were also run using an initial ramp elevator input of slope 1^o/sec to demonstrate the effectiveness of the linear and nonlinear IAEP. The initial conditions for these simulations were identical to the ones for the simulations discussed above. The ramp was initiated at a time of 2 seconds. Figures 3.22 - 3.27, show the results from these simulations. As seen from these figures, the IAEP performed well for η values of 0.01 and 0.05. However, at an η value of 0.1, the estimated elevator limit did not maintain the angle of attack within its limit value. As seen from Fig. 3.26 the maximum angle of attack reached for the linear predictive case was 9.1^o, although the predicted stall angle of attack was 8.0^o. This phenomenon may be attributed to the fact that at a high η value of 0.1, the aerodynamic model becomes fairly nonlinear. However, the limit estimates used to restrict the elevator are obtained from solutions of linearized equations of motion using the static values of the performance parameters. Since the limit calculations of the elevator are completed while angle of attack is still fairly low, they are

not precise. Thus further study is needed in order improve the system to avoid such limit violations.

The simulations discussed above showed that the IAEP performed well for most of the simulations. For severe icing cases, restricting the elevator to the estimated elevator limit value may not provide full protection against limit violations. As a result, it may be necessary to use predictions obtained from the solution of the nonlinear equations of motion in severe icing.

3.5 Figures

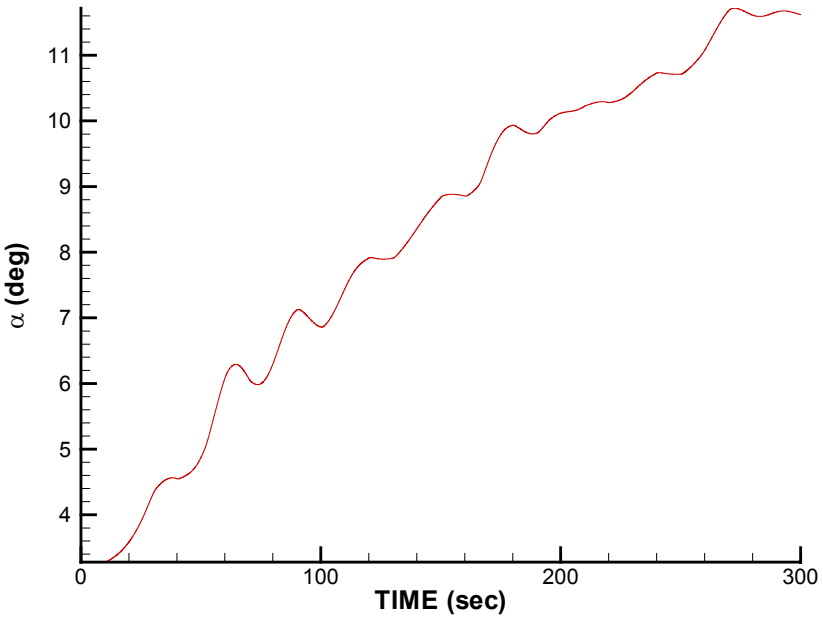

Fig. 3.1. Angle of Attack Response for Validation at $\eta = 0.05$

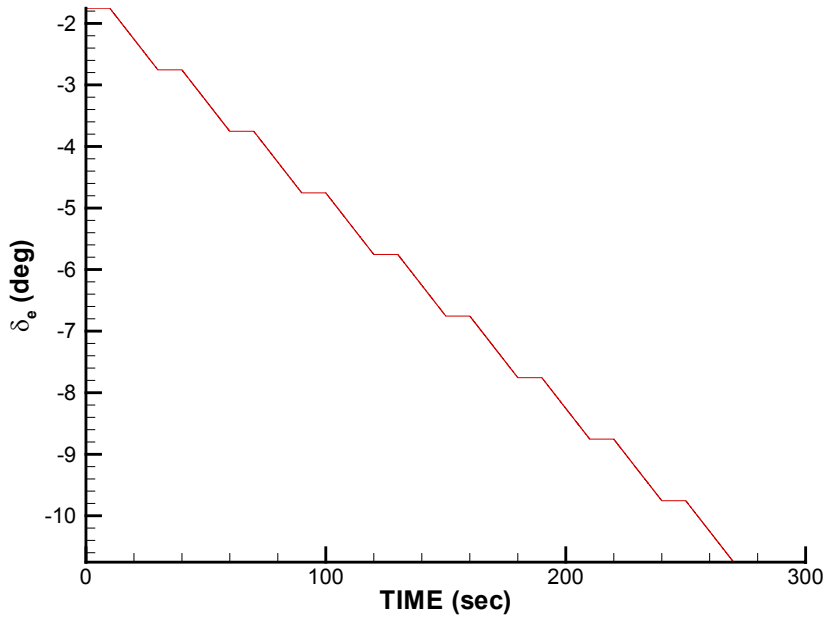

Fig. 3.2. Elevator Command for Validation at $\eta = 0.05$

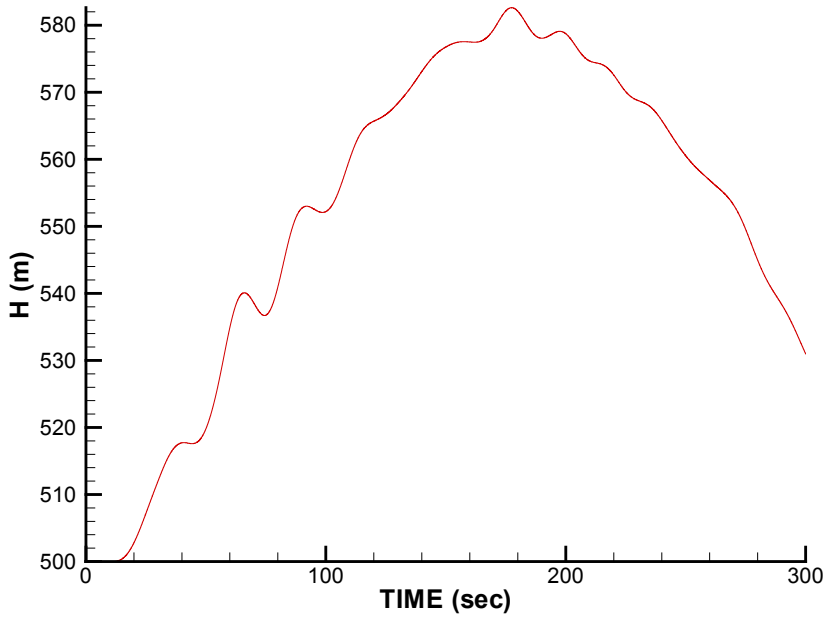

Fig. 3.3. Altitude Time History for Validation at $\eta = 0.05$

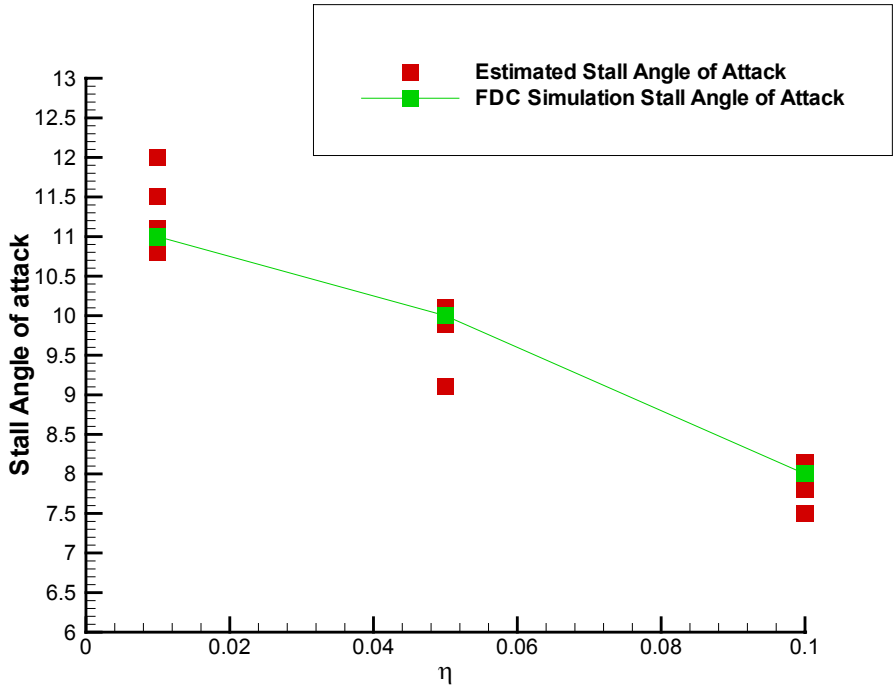

Fig. 3.4. Comparison of Estimated Stall Angles with Simulation Values

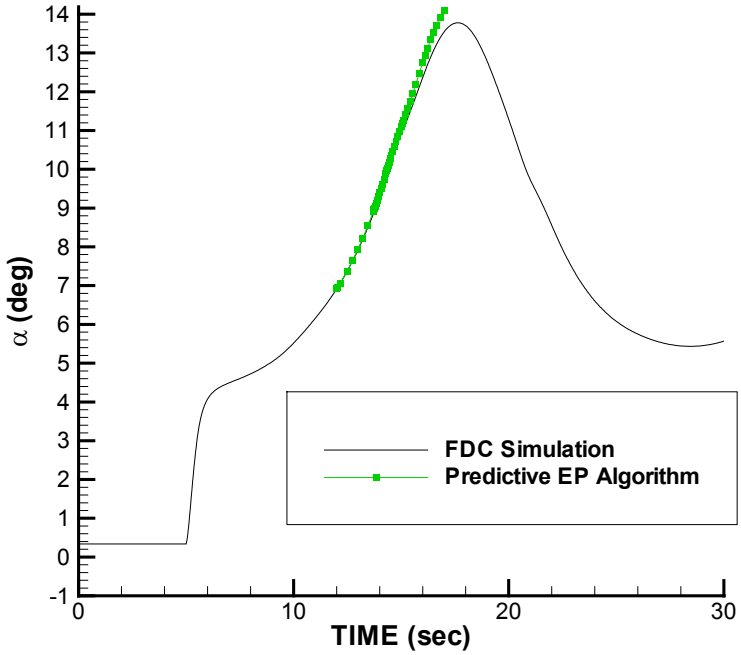

Fig. 3.5. Validation of Open Loop IAEP Prediction with FDC Result

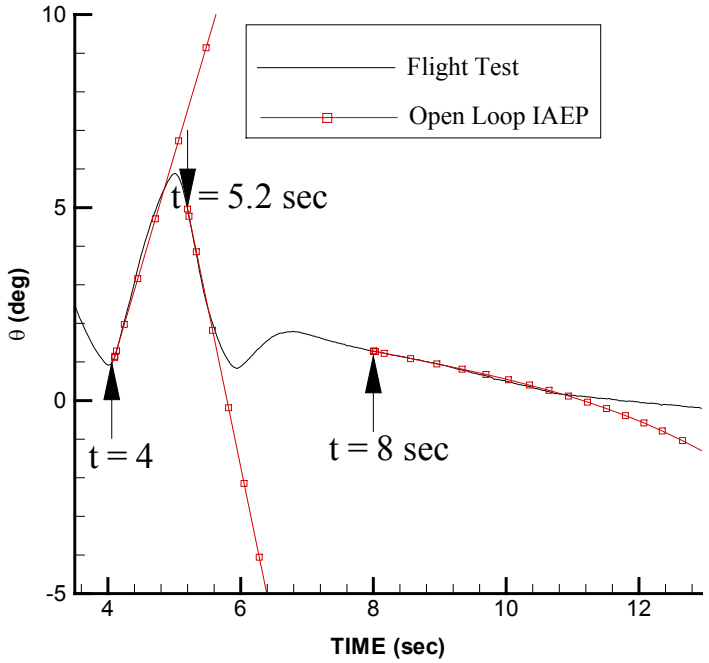

Fig. 3.6. Validation of Open Loop IAEP Prediction with Flight Test Data[39]

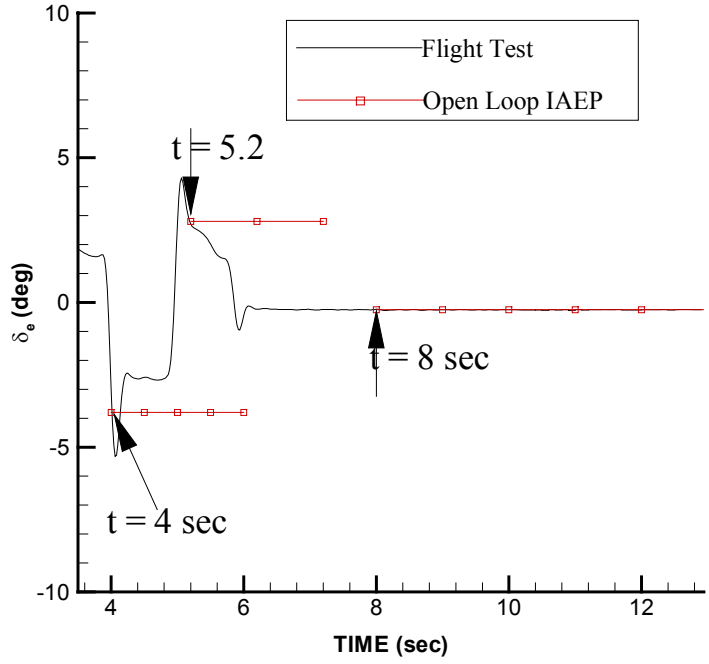

Fig. 3.7. Validation of Open Loop IAEP Prediction with Flight Test Data[39]

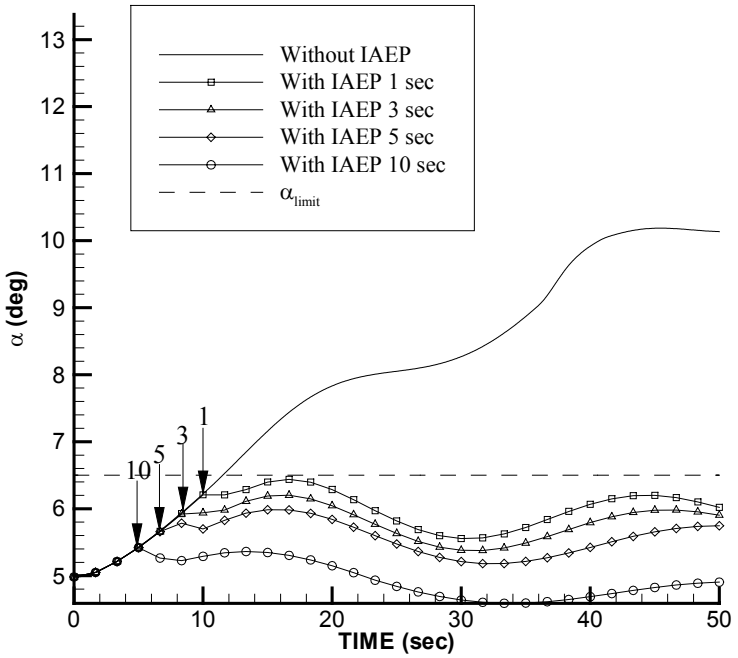

Fig. 3.8. IAEP Performance for Different Integration Times for Ramped Elevator Input.

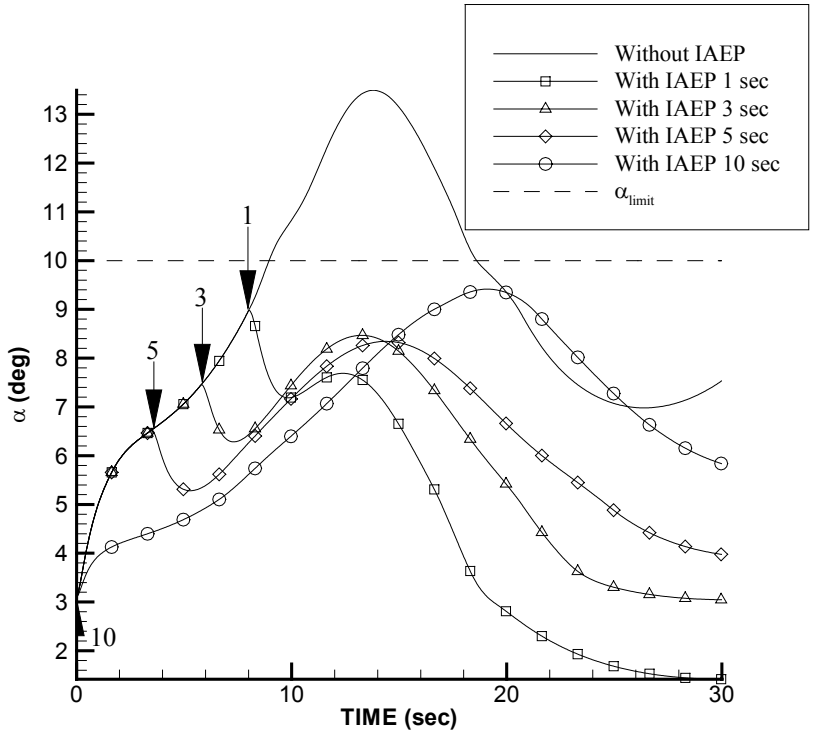

Fig. 3.9. IAEP Performance with Different Integration Times for Step Elevator Input

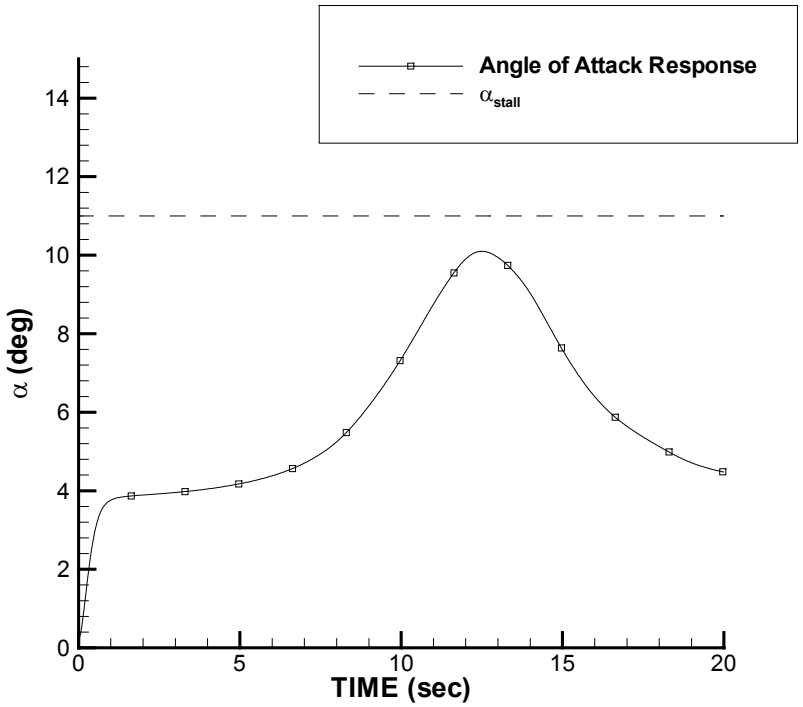

Fig. 3.10. Angle of Attack Response from Issuing Step Command Equal to Elevator Limit at η=0.01

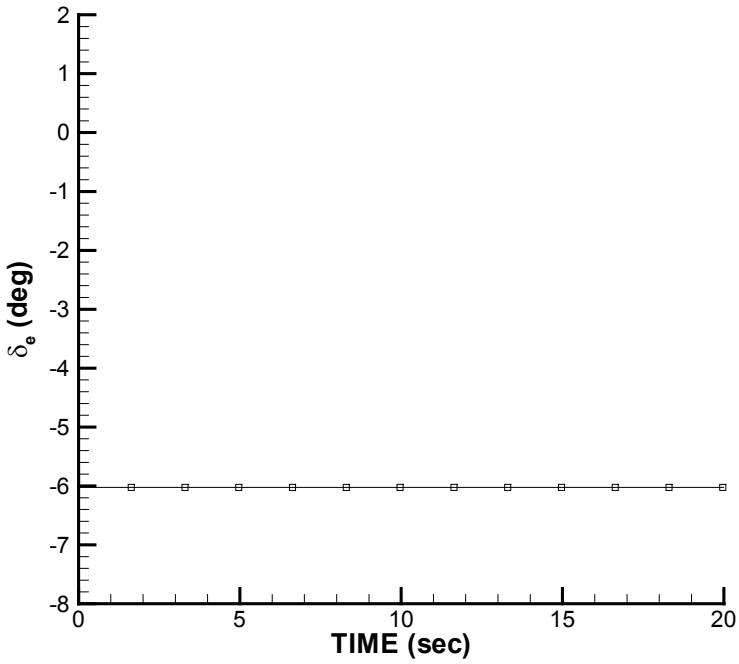

Fig. 3.11. Step Elevator Input Equal to the Limit Value at η=0.01

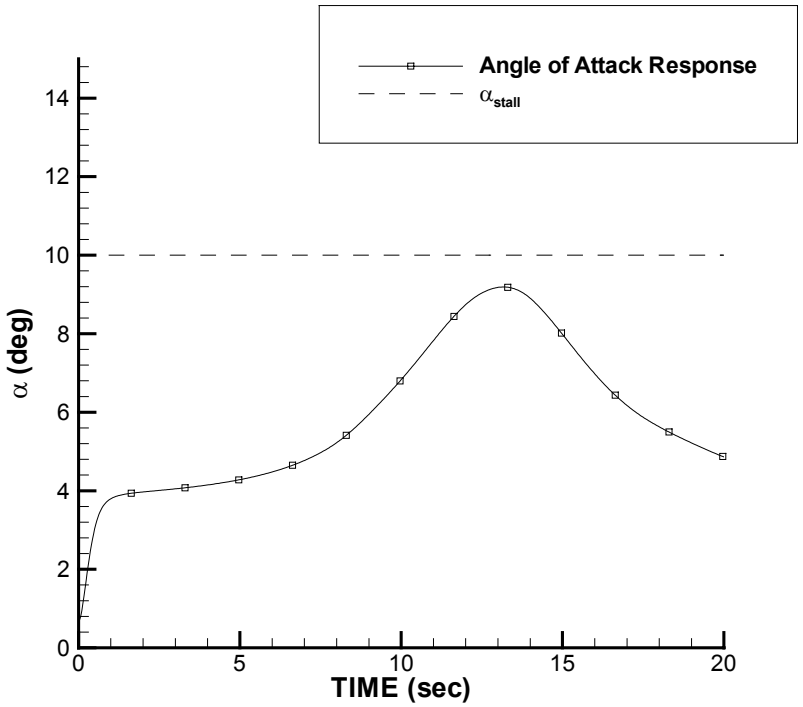

Fig. 3.12. Angle of Attack Response from Issuing Step Command Equal to Elevator Limit at η=0.05

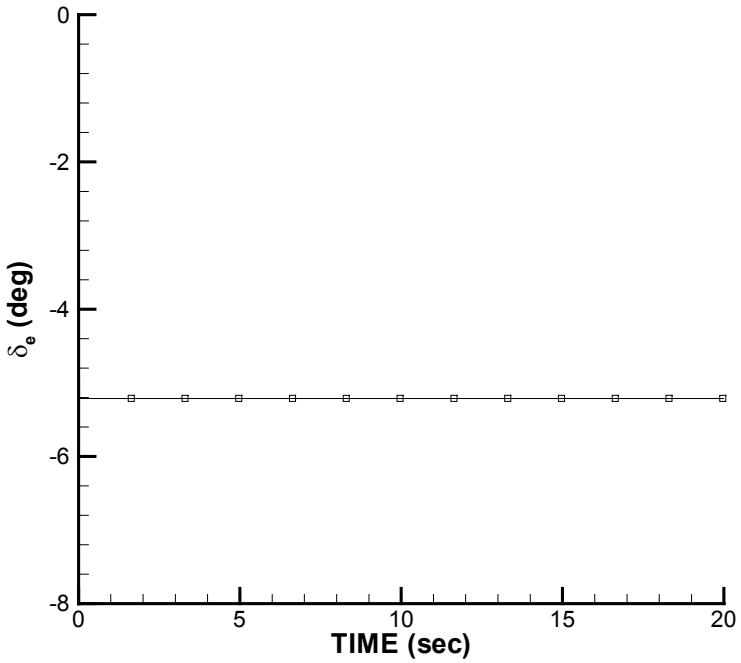

Fig. 3.13. Step Elevator Input Equal to the Limit Value at η=0.05

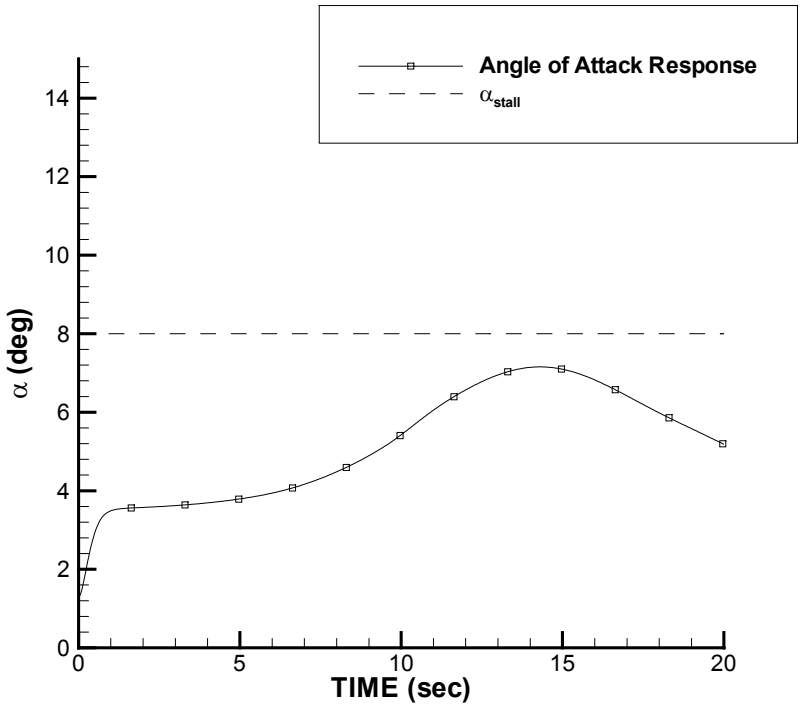

Fig. 3.14. Angle of attack Response from Issuing Step Command Equal to Elevator Limit at η=0.1

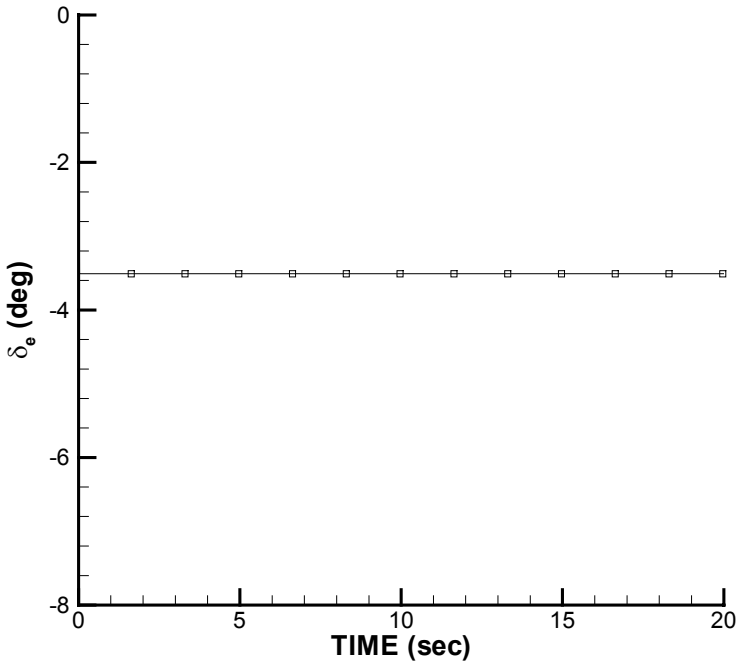

Fig. 3.15. Step Elevator Input Equal to the Limit Value at η=0.1

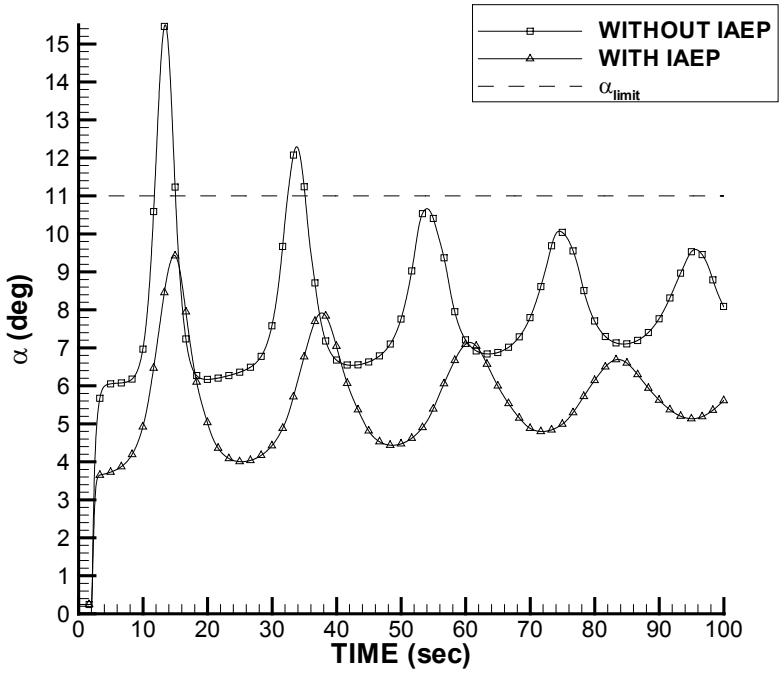

Fig. 3.16. Comparison of Simulations with and without IAEP for a 10^{o} Elevator Input at η=0.01

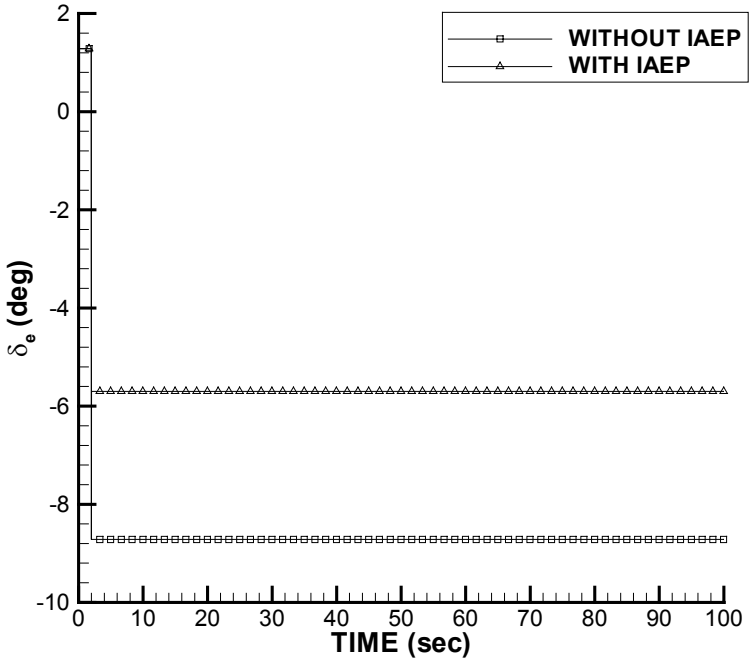

Fig. 3.17. IAEP Elevator Command for an Initial Step Elevator at η=0.01

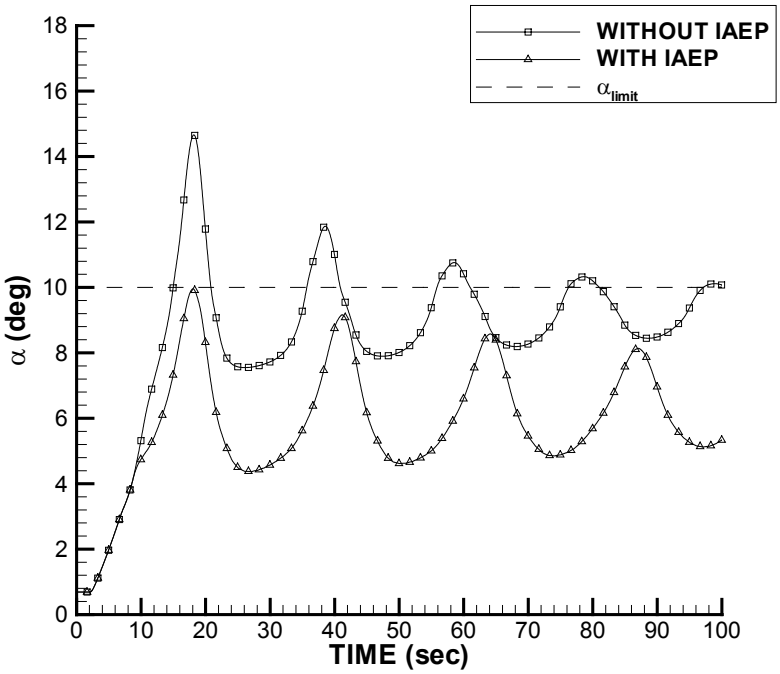

Fig. 3.18. Comparison at η=0.05

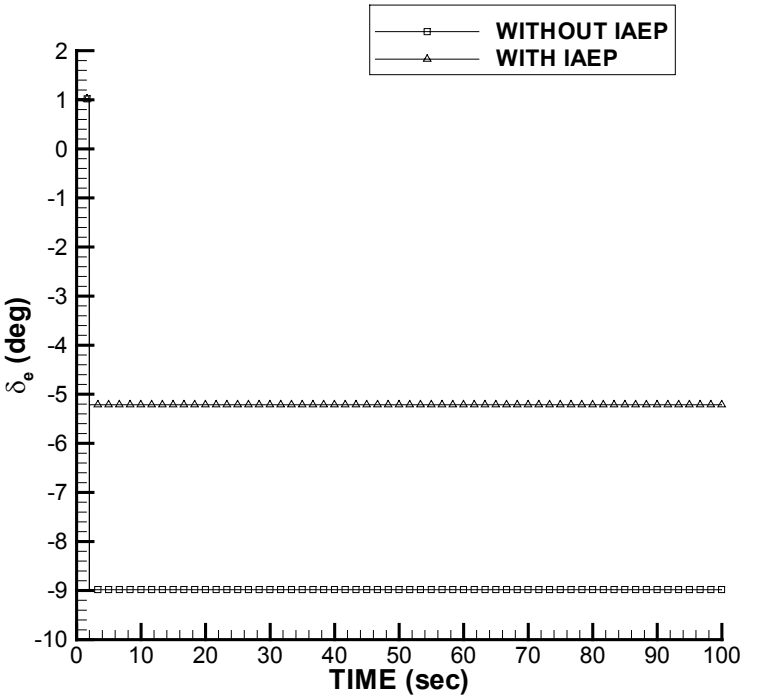

Fig. 3.19. IAEP Elevator Command at η=0.05

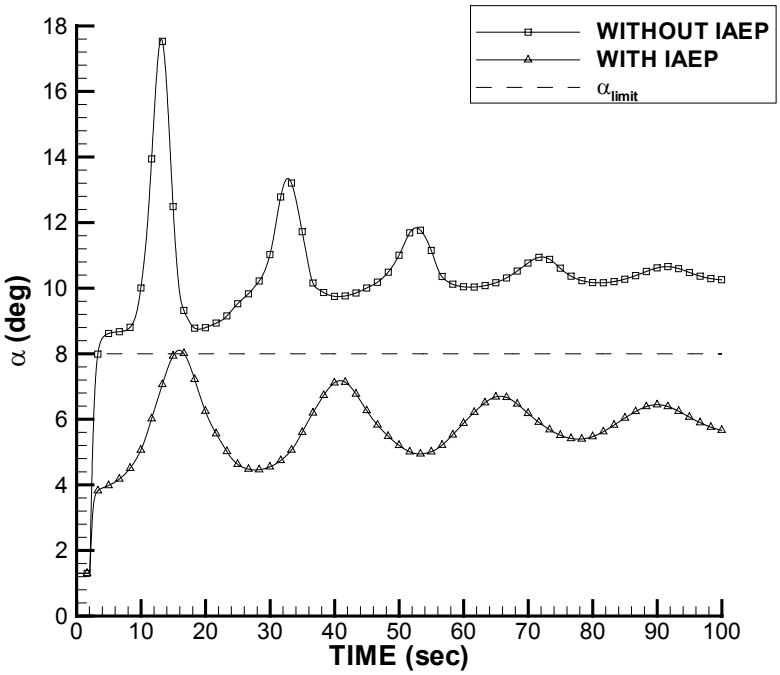

Fig. 3.20. Comparison at η=0.1

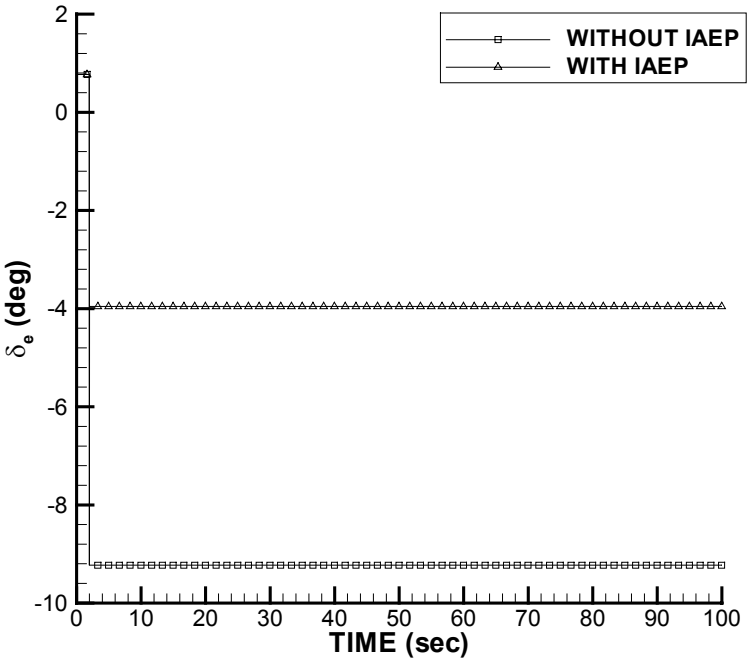

Fig. 3.21. IAEP Elevator Command at η=0.1

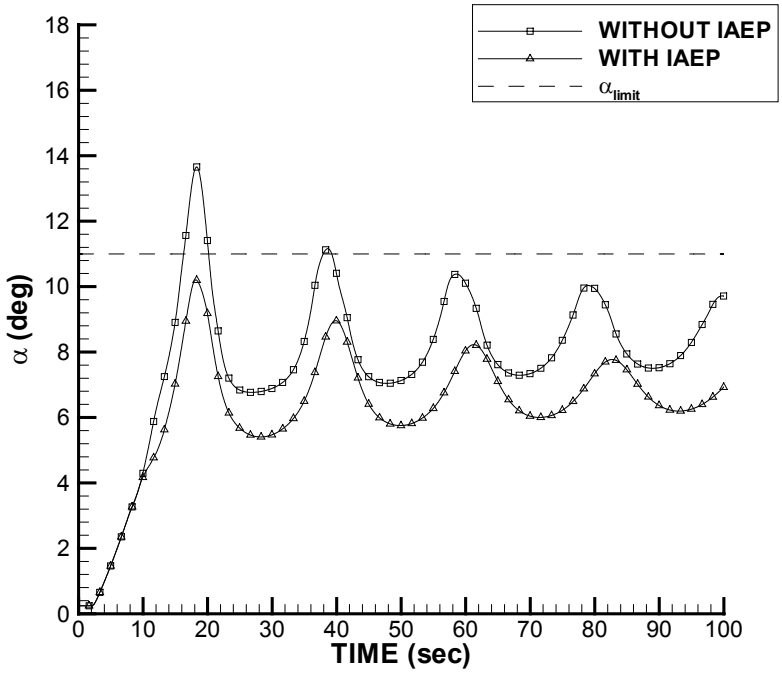

Fig. 3.22. Comparison of simulations with and without IAEP with Initial Ramp Elevator Command at η=0.01

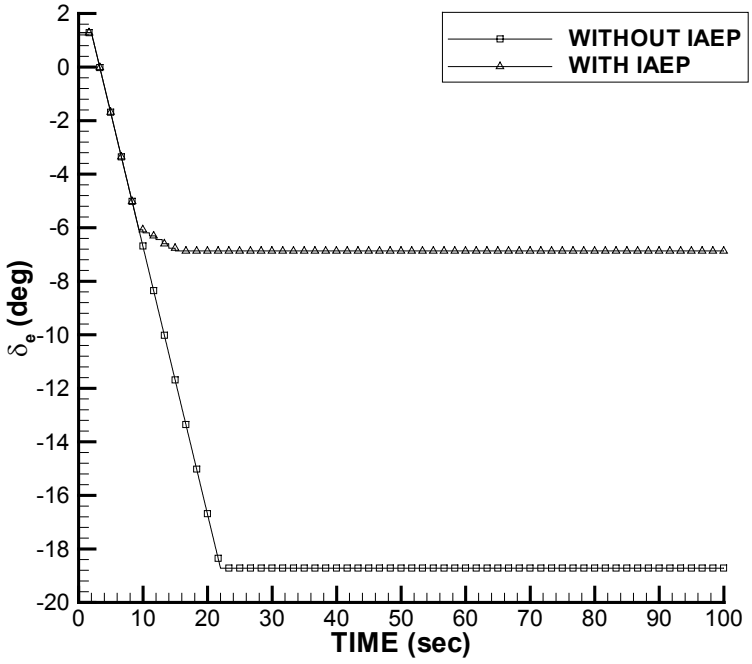

Fig. 3.23. IAEP Elevator Command at η=0.01

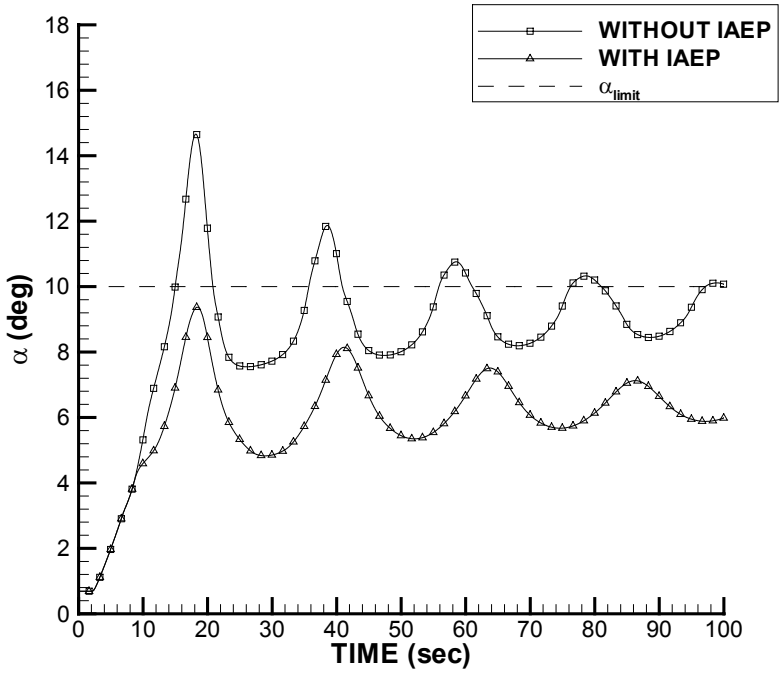

Fig. 3.24. Comparison at η=0.05

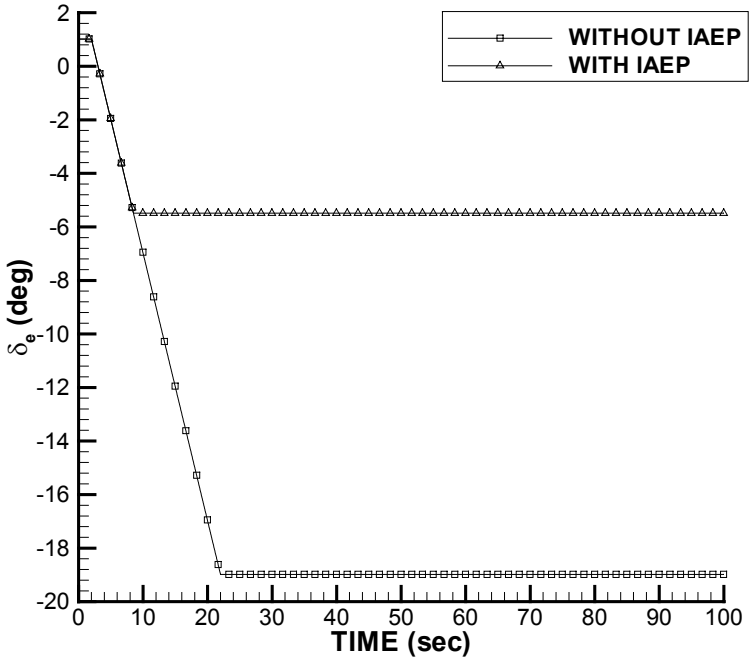

Fig. 3.25. IAEP Elevator Command at η=0.05

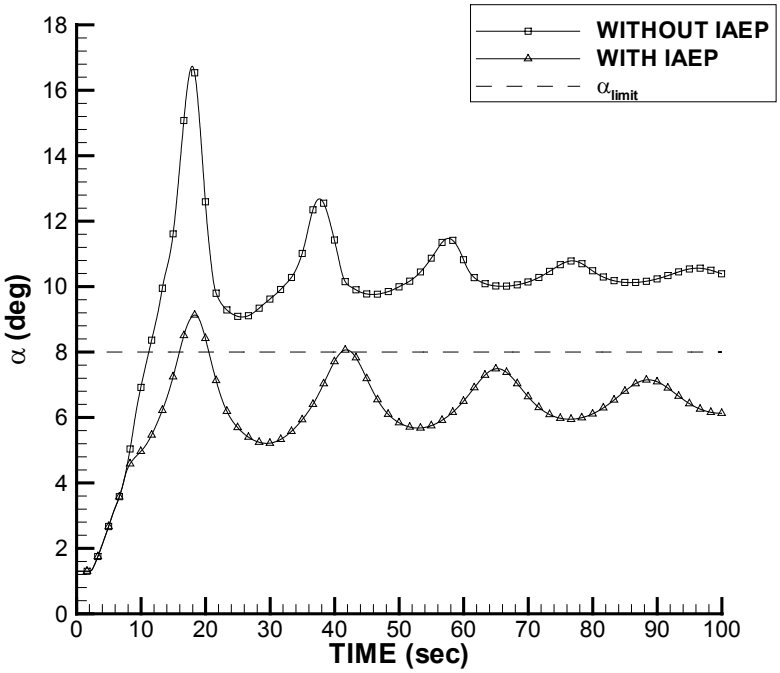

Fig. 3.26. Comparison at η=0.1

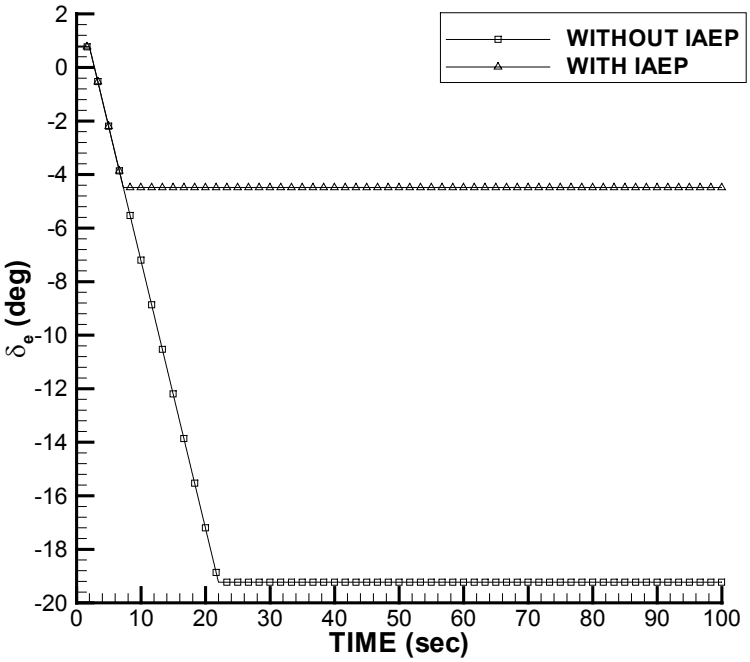

Fig. 3.27. IAEP Elevator Command at η=0.1

4 SUMMARY, CONCLUSIONS AND RECOMMENDATIONS

4.1 Summary

Iced aircraft envelope protection is a very important aspect of ice protection systems. However, there has been very little work done to date in the development of a sophisticated envelope protection system that is effective in icing conditions. The research reported in this thesis was directed toward developing such a system. This research was a part of the Smart Icing Research project. As a part of this project, a nonlinear aerodynamic model was developed for the Twin Otter aircraft. The model was based on data obtained from wind tunnel tests performed using a scaled model of the Twin Otter. The primary focus of this thesis, however, was to develop an envelope protection system to provide angle of attack limiting in icing conditions. The research conducted was two fold. It involved developing a method to calculate the aerodynamic

limits of the envelope and ascertaining whether the limit was violated. This thesis dealt with the Longitudinal Open Loop aspect of this problem where only the angle of attack was limited and the aircraft was flown manually.

In this project, available data were analyzed to determine trends between icing severity and performance degradation. The analyses led to the development of a simple scheme for estimating the stall angle of attack from the degradation in lift.

A system was also designed to estimate limit values of in advance so that the pilot could be given enough lead-time to take preventive measures to avoid stall. The system involved solving the linearized equations of motion. The IAEP scheme was implemented in the FDC[23] and several simulations were run to evaluate the effectiveness of the IAEP in avoiding stall.

4.2 Conclusions

The following conclusions may be drawn from the research reported in this thesis:

- The nonlinear aerodynamic model showed reasonable trends.
- The method developed to estimate the stall angle of attack showed promising results if the estimates were calculated when the aircraft was trimmed.
- The predictive IAEP developed as a part of this project showed promising results.
- Predictive IAEP solutions were validated against flight test data.
- The estimates of elevator limits based on the solutions of the linearized equations of motion were reasonable, however, at severe icing conditions restricting the elevator

deflection to the estimated limit value did not prevent the aircraft from exceeding its angle of attack limit.

- A more robust algorithm is necessary to calculate the elevator limits, especially in sever icing conditions.

4.3 Recommendations

This thesis documents only the initial stages of the research for developing an effective envelope protection system for icing conditions. In order to advance this research it is imperative to obtain more 3-D data in the nonlinear aerodynamic region. This may be done either through flight tests or through wind tunnel tests using scale models. This is a major task but it is crucial to the development of an iced aircraft envelope protection scheme. Based on the results obtained in this research the following recommendations were made:

- Use 3-D aircraft data at different icing conditions to formulate a more accurate non-linear aerodynamic model, which could be used in the development and testing of improved envelope protection systems.
- Investigate relationships between the encroachment of the iced aircraft flight envelope and trends in the degradation of aerodynamic parameters such as aircraft lift, drag, and moments using 3-D aircraft data.
- Incorporate aircraft performance information with stability and control identification to predict the envelope.
- Develop the capability to estimate the reduced envelope of an aircraft when it is out of trim.

- Develop a more accurate method to predict the elevator limit based on the non-linear equations of motion
- Expand the envelope protection system to include critical parameters such as the roll angle of attack and control limits such as minimum power and maximum aileron deflections
- Incorporate dynamic flight information to improve the predictions of envelope violations. Flight information could be used to evaluate the accuracy of past predictions. A method could be developed where the evaluation histories could be used to make improvements to the prediction scheme online for improving the accuracy of the predictions.
- The system developed in this project was aircraft specific. Further research needs to be completed in order to develop a generic system, which could be applied to all aircraft.

APPENDIX

This Appendix includes all the Matlab scripts that were used to implement the Iced Aircraft Envelope Protection system (IAEP) in FDC. Some of the codes were modified from previous versions while others were written only to facilitate the implementation of the envelope protection system in the Simulink/Matlab based FDC.

Solveom1.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This top level script is used to test different aspects of the
% envelope protection system.
%
% Kishwar Hossain
% 07/10/02
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

global ice_loc kn tspan env VEL
global pp qq nn deltae_corr dele newde
clear all
close all
found = 0;

for ii = 1:1;
   if ii == 1;
      qq = 'v60run01';
   elseif ii == 2;
      qq ='v50run02';
   elseif ii == 3;
      qq ='v60run06';
   elseif ii == 4;
      qq ='v60run08';
   elseif ii == 5;
      qq ='v60run09';
   elseif ii == 6;
      qq ='v60run10';
   elseif ii == 7;
      qq ='v50run07';
   elseif ii == 8;
      qq ='v50run08';
   end


   for span = 1;
   e = 10^-4;
   comp
   tspan = [0 span];
   tecsave3
```

```
   tini = 10;
   while found < 1 & tini < 100
      tini = tini+.1;
            i = find(time_in >= tini-e & time_in<=tini+e);
            solver4
    end
    env = tini;
    cd c:/matlabr11/toolbox/fdc13
    bat_kk;
    cd c:/'documents and settings'/'all users'/desktop/eom2;
    found = 0;
end
end
```

Ic.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This script is executed within the "solveom" code in order
% to provide the initial conditions for the solution of the nonlinear
% vequations of motion to obtain predictions
% of aircraft states in the future.
%
% Kishwar Hossain
% 07/10/02
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

global ice_loc eta kn tspan deltae_corr de_tt eta_r
ice_loc = 2;
eta = eta_in(i);
eta_r = eta;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Initial Conditions
ini = zeros(9,1);

%Velocity comp
ini(1) = u_in(i);%-0.002316;                    %U;
ini(2) = v_in(i);%vw_in(i);%-0.003442;          %V;
ini(3) = w_in(i);%ww_in(i);%-1.008122;          %W;

%Pitch, Roll, yaw rates
ini(4) = p_in(i)*pi/180;                        %P;
ini(5) = q_in(i)*pi/180;                        %Q;
ini(6) = r_in(i)*pi/180;                        %R;

%Euler angles
ini(7) = phi_in(i)*pi/180;                      %phi;
ini(8) = psi_in(i)*pi/180;                      %xi;
ini(9) = theta_in(i)*pi/180;                    %theta;

%Position
%ini(10)= 0;                                    %X
%ini(11) = 0;                                   %Y
%ini(12) = 0;                                   %Z

%Velocity
%ini(13) = 0; %V_in(i);                         %vinf

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Control deflections
global dele delf da dr
dele = deltae_in(i)*pi/180;
da = deltaa_in(i)*pi/180;
dr = deltar_in(i)*pi/180;
delf = deltaf_in(i)*pi/180;

if abs(de_t) > 1
   de_t = 0;
end
de_tt = de_t*180/pi;
```

```
%Thrust components
global Pxi Pyi Pzi Mti Nti Lti ice_loc
Pxi = Xp_in(i);
Pyi = Yp_in(i);
Pzi = Zp_in(i);
Mti = Mt_in(i);
Nti = Nt_in(i);
Lti = Lt_in(i);

%Atmospheric components
global rho g alpha
rho = rho_in(i);
g = g_in(i);

%Nonlinear aerodynamics

cd c:/matlabr11/toolbox/fdc13/data

global cxc czc dcx dcz cmcoeff dcm
load cxc.mat -ascii
load czc.mat -ascii
load dcx.mat -ascii
load dcz.mat -ascii
load cmcoeff.mat -ascii
load dcm.mat -ascii

cd c:\'documents and settings'\'all users'\desktop\eom2;

%Solve nonlinear differential equation
options = odeset('RelTol',1e-4,'AbsTol',[1e-6 1e-6 1e-6 1e-6 1e-6 1e-6
1e-6 1e-6 1e-6]);
[t pred] = ode45('de6',tspan,ini);


%Definition of all the output parameters
%vinf = pred(:,13);
t = t + tini*(ones(size(t)));
u = pred(:,1);
v = pred(:,2);
w = pred(:,3);
P = pred(:,4)*180/pi;
Q = pred(:,5)*180/pi;
R = pred(:,6)*180/pi;
phi = pred(:,7)*180/pi;
xi = pred(:,8)*180/pi;
theta = pred(:,9)*180/pi;
%X = pred(:,10);
%Y = pred(:,11);
%Z = pred(:,12);
vinf = sqrt(u.^2+v.^2+w.^2);
del = ones(size(theta))*dele*180/pi;

beta = asin(v./vinf)*180/pi;
alpha = (atan(w./u))*180/pi;
%alimit = (110*eta^2)-58.1*eta+17.29;
%[mh,ii]= max(H_in);
%alimit = alpha_in(ii);
```

```
delimit = dele;
dec = -1;
% Stall angles as obtained from FDC simulations

if eta == 0.1
      alimit2 = 9.99;
elseif eta == 0
      alimit2 = 18;
elseif eta == 0.05
      alimit2 = 10.5;
elseif eta == 0.15
      alimit2 = 9.61;
elseif eta == 0.2;
      alimit2 = 9.63;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if alimit2<max(alpha)
   tecsave2;
   diffa = alimit2-max(alpha)
   while diffa<0 & dele<.15
      if dec <= -0.0009
            dele = sign(dele)*(abs(dele)-0.01);
      else dele = dele+.001;
      end
      dec = dele;
      tspan2 = [0 10];
      [t1 pred1] = ode45('de6',tspan2,ini);
      t = t1 + tini*(ones(size(t1)));
      u = pred1(:,1);
      v = pred1(:,2);
      w = pred1(:,3);
      P = pred1(:,4)*180/pi;
      Q = pred1(:,5)*180/pi;
      R = pred1(:,6)*180/pi;
      phi = pred1(:,7)*180/pi;
      xi = pred1(:,8)*180/pi;
      vinf = sqrt(u.^2+v.^2+w.^2);
      theta = pred1(:,9)*180/pi;
      beta = asin(v./vinf)*180/pi;
      alpha = (atan(w./u))*180/pi;
      del = ones(size(theta))*dele*180/pi;
      tecsave2;
      al = (atan(pred1(:,3)./pred1(:,1)))*180/pi;
      diffa = alimit2-max(al)
   end
   fclose('all');
   found = 1
end
deltae_corr = dele;
if deltae_corr > -1*10^-4 & deltae_corr < 1*10^4
      deltae_corr = 0;
else
      deltae_corr = deltae_corr-deltae_in(1)*pi/180;            %for
step
      %deltae_corr = deltae_corr-de_t*tini-deltae_in(1)*pi/180; %for
ramps
end
```

De6.m

```
function[ydot] = de6(t,y);
global ice_loc eta kn tspan env pp qq nn deltae_corr

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%     Equations of Motion using the Flat-Earth Approximation
%     This is used to solve the equations of motion
%
%     Kishwar Naz Hossain
%     07/12/02
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Velocity comp
U       = y(1);
V       = y(2);
W       = y(3);

%Pitch, Roll, Yaw rates
P       = y(4);
Q       = y(5);
R       = y(6);

%Euler angles
phi     = y(7);
xi      = y(8);
theta   = y(9);

%Velocity
vinf = sqrt(U^2+V^2+W^2);

%A/C Specs
global g alpha

Ix      = 21787.0;            % kgm^2 in Fr
Iy      = 31027.0;            % kgm^2 in Fr
Iz      = 48639.00;           % kgm^2 in Fr
Jxy     = 0.0;                % kgm^2 in Fr
Jxz     = 1498.0;             % kgm^2 in Fr
Jyz     = 0.0;                % kgm^2 in Fr
m       = 4600.00;            % kg

cbar    =    1.9810;         %
b       =    19.81;         % m
S       =    39.02;         % m^2

%Aerodynamic Angles
beta    = asin(V/vinf);
if U    ~= 0
    alpha = atan(W/U);
else alpha = pi/2;
end

%Total Velocity
global rho
dynv = 0.5*rho*vinf^2;
```

```
%Assign values of Stability and Control Derivatives

CX0   =  -0.0489;   CZ0    = -0.36;       Cm0  =  0.040;
CXa   =  0.157;     CZa    = -5.66;       Cma  = -1.31;
CXa2  =  4.0006;     CZa3   =  0.0  ;      Cma2 = -0.0;

CXa3  = -0.0  ;      CZq    = -19.97;      Cmq  = -34.2;
CXq   = -0.0   ;     CZde   = -0.608;      Cmde = -1.74;
CXdr  =  0.0    ;    CZdeb2 = -0.0  ;      Cmb2 =  0.0;
CXdf  = -0.0    ;    CZdf   = -0.0  ;      Cmr  = -0.0;
CXadf =  0    ;      CZadf  = 0     ;      Cmdf =  0.;

CY0   =  0.;         Cl0    =  0.;         Cn0  =  0.;
CYb   =  -0.6;       Clb    =  -0.08;      Cnb  =  0.1;
CYp   =  -0.2;        Clp    = -0.5;        Cnp  =  -0.06;
CYr   =  0.4;        Clr    =  0.06;       Cnr  =  -0.18;
CYda  =  0.;         Clda   = -0.15;       Cnda = -0.001;
CYdr  =  0.15;       Cldr   =  0.015;      Cndr =  -0.125;
CYdra =  0.;         Cldaa  =  0.;         Cnq  =  0.;
CYbdot=  0.;                               Cnb3 =  0.;
CZa2 = 0;   CXde = 0;


%Define coefficient matrix multiplier
global da dr dele delf Pxi Pyi Pzi Mti Nti Lti ice_loc

de = dele;
df = delf;
Px = Pxi;
Py = Pyi;
Pz = Pzi;
Mt = Mti;
Nt = Nti;
Lt = Lti;


contr = [1, alpha, alpha^2, alpha^3, beta, beta^2, beta^3,...
         P*b/2/vinf, Q*cbar/vinf, R*b/2/vinf, de, df, da, dr,...
      alpha*df, alpha*dr, alpha*da, de*beta^2, 0];

global AM;

%Define coefficient matrix
AM = [CX0     CY0     CZ0     Cl0     Cm0     Cn0  ;
       CXa     0       CZa     0       Cma     0    ;
       CXa2    0       CZa2    0       Cma2    0    ;
       CXa3    0       CZa3    0       0       0    ;
       0       CYb     0       Clb     0       Cnb  ;
       0       0       0       0       Cmb2    0    ;
       0       0       0       0       0       Cnb3 ;
       0       CYp     0       Clp     0       Cnp  ;
       CXq     0       CZq     0       Cmq     Cnq  ;
       0       CYr     0       Clr     Cmr     Cnr  ;
       CXde    0       CZde    0       Cmde    0    ;
       CXdf    0       CZdf    0       Cmdf    0    ;
       0       CYda    0       Clda    0       Cnda ;
       CXdr    CYdr    0       Cldr    0       Cndr ;
```

```
       CXadf   0       CZadf   0       0       0    ;
       0       CYdra   0       0       0       0    ;
       0       0       0       Cldaa   0       0    ;
       0       0       CZdeb2  0       0       0    ;
       0       CYbdot  0       0       0       0    ];


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Nonlinear aerodynamics

global cxc czc dcx dcz cmcoeff dcm


      a2 = alpha*180/pi;      %Need to convert AOA to deg.
      a = -10:24;             %The tables are from -10 deg to 24 deg

      if a2 > max(a)
            a2 = max(a);
      elseif a2 < min(a);
            a2 = min(a);
      end

      AM(1,1) = -interp1(a,cxc,a2);
      AM(2,1) = -(interp1(a,dcx,a2))*180/pi;
      AM(3,1) = 0;

      AM(1,3) = -interp1(a,czc,a2);
      AM(2,3) = -interp1(a,dcz,a2);

      AM(1,5) = interp1(a,cmcoeff,a2);
      AM(2,5) = (interp1(a,dcm,a2))*180/pi;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Iced Nonlinear model
global ice_loc eta

eta_ref_wing = 0.08;
eta_ref_tail = 0.2;

if ice_loc == 1
   eta_wing = eta;
   eta_tail = 1.843.*eta;
elseif ice_loc == 2
   eta_wing = eta;
   eta_tail = 0;
elseif ice_loc == 3
   eta_wing = 0;
   eta_tail = 1.843*eta;
end

alpha1 = alpha*180/pi;
de1 = de*180/pi;

if alpha1 < 16
```

```
   dcl = 0.088449 + 0.004836*alpha1 -0.0005459*alpha1^2 + (4.0859*10^-
5)*alpha1^3;
else dcl = -11.838 + 1.6861*alpha1 - 0.076707*alpha1^2 +
0.001142*alpha1^3;
end

KCL = -dcl/eta_ref_wing;
dcl = eta_wing*KCL;

dcd = -0.0089 + 0.001578*alpha1 - 0.00046253*alpha1^2 - 4.7511e-
5*alpha1^3 + 2.3384e-6*alpha1^4;
KCD = -dcd/eta_ref_wing;
dcd = eta_wing*KCD;

dcma = (-0.01892 - 0.0056476*alpha1 +1.0205e-5*alpha1^2 - 4.0692e-
5*alpha1^3 + 1.7594e-6*alpha1^4);
dcmde = (-0.014928 - 0.0037783*de1 + 0.00039806*(-de1)^2 - 1.1304e-5*(-
de1)^3 + 1.8439e-6*(-de1)^4);
DCM = -dcma + dcmde;
KCMA = dcma/eta_ref_wing;
KCMDE = dcmde / eta_ref_tail;
DCM = (0.75*eta_wing + 0.25*eta_tail)*KCMA + eta_tail*KCMDE;

DCX = -dcl*sin(alpha) - dcd*cos(alpha);
DCZ = dcl*cos(alpha) - dcd*sin(alpha);
DCM = DCM;

%Return the total force coef.
global Caero1;
Caero1 = AM'*contr';

%Return Iced coefficients
Caero1(1,1) = Caero1(1,1) + DCX;
Caero1(3,1) = Caero1(3,1) - DCZ;
Caero1(5,1) = Caero1(5,1) - DCM;

A = Caero1;

%Moment
Fax = A(1)*dynv*S;
Fay = A(2)*dynv*S;
Faz = A(3)*dynv*S;
La = A(4)*dynv*S*b;
Ma = A(5)*dynv*S*cbar;
Na = A(6)*dynv*S*b;

Ftx = Px;
Fty = Py;
Ftz = Pz;
Mt = Mt;
Lt = Lt;
Nt = Nt;
P = 0;
R = 0;

%Force and Moment Equations
udot = 1/m*(-m*g*sin(theta) + Fax + Ftx) + V*R - W*Q;
```

```
vdot = 1/m*(m*g*sin(phi)*cos(theta) + Fay - Fty) - U*R + W*P;
wdot = 1/m*(m*g*cos(phi)*cos(theta) + Faz + Ftz) + U*Q - V*P;

Qdot = 1/Iy*(Ma + Mt - (Ix - Iz)*P*R - Jxz*(P^2 - R^2));
Pdot = 0;
Rdot = 0;

%Kinematic equations
phidot = P + Q *sin(phi)*tan(theta) + R*cos(phi)*tan(theta);
xidot = (Q*sin(phi) + R*cos(phi))*sec(theta);
thetadot = Q*cos(phi) - R*sin(phi);

clear AM
ydot = [udot;vdot;wdot;Pdot;Qdot;Rdot;phidot;xidot;thetadot];
```

Bat_env1.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This code is used to test out a "batch" file for the operation of
% the FDC13 code in different scenarios.
% Devesh Pokhariyal
% March 28, 2000
% Modified for Envelope Protection analysis
% Kishwar Hossain
% September, 2001
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear variables;
warning('off');
version5_bat; % File contains aircraft characteristics.
clear variables;

global kn irun COEF_CLEAN deltae iaepdeinit fres;
irun = 0;
kn = [];
global simtimek deltae_corr simres
simtimek = 300;

global ice_loc ice_cond ramp_rate ramp_start ramp_stop dur grav iaep
iaepnl vel1 alimit2;
ice_loc = 2;                    % 1 - all, 2 - wing only, 3 - tail only
ice_cond = 0;                   % 0 constant; 1 ramped
ramp_start = 0;
ramp_stop = 150;
global sigma_turb
sigma_turb = 0; % Values in "g"'s
for iaep = 2;
alimit2 = 18;
simres=[];
for iscen = 1:1,  % should 1-2
fres = [];
      for ivel = 70,
         for eta_f = [0] ;
            for ramp_rate = 0
                  dur = simtimek-ramp_start,
               for deltae = -6;
tic
                        irun = irun + 1;
                        load aircraft.dat -mat; % Load the aircraft-
parameters file
                                                % representing the Twin
Otter.
                        xfix = 1;

                        if iscen == 1
                              const_vel = 0; % Represents constant
power case
                        elseif iscen == 2
                              const_vel = 1; % Represents constant
velocity case
```

```
                        end

                        % Load the actuator model:
                        load actmod55.dat -mat; %  Choose model for V =
55 m/s.
                        refvelocity = 55;

                        % Icing variables:
                        global eta_final eta2 iaepde
                        eta2 = [];

                        % Nonlinear model force and moment coefficients
                        global cxc czc dcx dcz cafcoeff cnfcoeff
cmcoeff dcnf dcaf dcm cla clcoeff cda cdcoeff
                        load cxc.mat -ascii
                        load dcx.mat -ascii
                        load czc.mat -ascii
                        load dcz.mat -ascii
                        load cmcoeff.mat -ascii
                        load dcm.mat -ascii

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Initialize
ACTRIM_BAT%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                        % DEFINE INITIAL FLIGHT CONDITION USING THE
ACTRIM_BAT.file
                        % Will always run the trim program (as opposed
to loading a trimmed
                        % flight condition).

                        % Input the initial conditions around which the
Twin Otter aircraft
                        % will be trimmed - this is done before the
trim routine,
                        % "actrim_bat.m", is called.
                        % **Note**: The variables, such as, V, H etc.
that
                        % define the initial conditions, are cleared in
actrim_bat.m.
                        sysname = 'twinvernl_bat'; % Aircraft system
model used in ACTRIM_BAT.M

                        opt = 1; % This is used to chose the type of
trim flight condition in
                                  % actrim_bat.m: 1 = STEADY WINGS
LEVEL FLIGHT, 2 = steady
                                  % turning flight etc..

                        H = 3048; %2438.4; % Initial Altitude [m]  orig
2300
                        V = ivel;
                        eta_final = eta_f;

                        psi = 0; % Heading angle [deg]

                        % Use specified manifold pressure or flight-
path angle ([m]/f)?
                        gammatype = 'f';
```

```
                        gamma = 0;

                        pz = 20; % ["Hg] Since gammatype is "f",
manifold pressure is
                  % adjusted by numerical trim algorithm.

                        rolltype = 's';   %No rolling, so default
setting, which is stability
                                          %axis roll will be used.

                        deltaf = 0.0;  % Trim flap angle in degrees.

                        n = 2200% Engine speed [RPM] used in trim
routine.

                        run actrim_bat; % Variables used are cleared
before
%%%%%%%%%%%%%%%% INITIALIZE VOR AND/OR ILS SYSTEM %%%%%%%%%%%%%%%%

                        % Initialize the VOR system:
                        HVOR = 0;
                        xVOR = 0;
                        yVOR = 0;
                        CD   = 0;
                        disp(' ');
                        disp('Default values for VOR set');
                        disp(' ');

                        % Initialize the ILS system:
                        HRW   = 0;
                        xRW   = 0;
                        yRW   = 0;
                        gamgs = -3*pi/180;
                        psiRW = 0;
                        xloc  = 2000;
                        xgs   = 300;
                        ygs   = -100;
                        Sgs   = 625/abs(gamgs);
                        Sloc  = 1.4*xloc;
                        disp(' ');
                        disp('Default values for ILS set');
                        disp(' ');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

                        % Fixstate of aircraft

                        if (const_vel == 0);
                              opt = 1; % Chooses option of "Fix
asymmetrical states" in fixstate_bat.m
                        elseif (const_vel == 1);
                              opt = 3;
                              fix = 1; % 1 represents velocity.
                        end

                        %opt = 4;
```

```
                        sysname = 'twinvernl_bat'; % Used in
fixstate_bat.m

                        fixstate_bat;

                        apmode_bat2; % Run file to determine autopilot
modes.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                        eta2 = [];

                        % Start simulation
                        sim('apilotnlenv3_bat'); % Note:  The screen
appears to be "frozen" when the
                                                                  %
simulations are running.

            m = [size(time) size(eta2)]
            results2_bat;     % Creates results in [degs] and [degs/s]
                        subplot(221);plot(time,alpha);hold on;
                        subplot(222);plot(time,deltae);hold on;
                        subplot(223);plot(time,V);hold on;
                        subplot(224);plot(time,H);hold on;

            eta = eta_final;
            Chrms_calc;
                        Ch_calc;
            clear eta
            if ice_cond == 0;
               eta = eta_final*ones(length(time),1);
            else
               eta = zeros(length(time),1);
               i = find(time == ramp_start);
               i1 = find(time == ramp_stop);
               i3 = find(time == simtimek);
               eta(1:i)=0;
               for i2 = (i+1):i1
                  eta(i2)=time(i2)*ramp_rate;
               end
               eta(i2+1:i3)=ramp_rate*time(i1);
            end

                        % For filtering data will need to run Matlab on
a unix machine with
                        % filtering capabilities.

                        my_interp2;

            reno_filt_jim;

                        %disp('Finished Run number: %g', irun);
                        irun

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                        %SAVE RESULTS
                        vel = V(1);
```

```
                        Xpcalc_int=-
Xa_int+(9.81*4600.*sin((alpha_int*3.14159)/180));
                        cd c:/matlabr11/toolbox/fdc13
                        saveout;
                        %savefiles7;      %khossain 42202 Changed from
savefiles6.  savefiles7 only ouputs the relevant
                                                        %variables in a
tecplot compatible file.  None of the no-noise
                                                        %data are saved.
It may be incorporated in the future

      %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
fres = [fres;eta_final alimit2 iaepde max(alpha)];
clear global iaepde
                  end
            end
   end
   end

end
end
toc
```

Icesev_oliaep.m

```
function Caero = icesev2_jmelody(stuff)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ICESEV Computes the change in the aircraft force and moment
% coeffecients
%
%           Inputs:     input
%
% input = [time, 1, alpha, alpha^2, alpha^3, beta, beta^2, beta^3,
%        pb/2V, qc/V, rb/2V, deltae, deltaf, deltaa, deltar,
%        alpha*deltaf, alpha*deltar, alpha*deltaa, deltae*beta^2 0
q_gc/V
%          ice_loc]
%
%  Using the matrix AM and the time, each coefficent is calculated
%  using C()iced = C()clean*(1 + eta*k()) where eta is the icing
%  severity factor and k() is a function of time and coefficent.
%
%  ice_loc is used to specify the location of icing:
%    = 1   is all-iced
%    = 2   is wing icing only
%    = 3   is tailplane icing only
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


global kn %(error flag)

time = stuff(1);

for n = 1:(length(stuff)-26),
   ytmp(n) = stuff(n+1);
end

% Variables needed for solveroliaeplin & solveroloaepnonlin

ii = length(stuff);
ice_loc=stuff(ii-17);
uu = stuff(ii-16);
vv = stuff(ii-15);
ww = stuff(ii-14);
rho1 = stuff(ii-13);
grav1 = stuff(ii-9);
phi1 = stuff(ii-8);
theta_corr = stuff(ii-7);
psi1 = stuff(ii-6);
XP1 = stuff(ii-5);
YP1 = stuff(ii-4);
ZP1 = stuff(ii-3);
LP1 = stuff(ii-2);
MP1 = stuff(ii-1);
NP1 = stuff(ii);


ytmp(9) = ytmp(9) - stuff(21);
```

```
global AM;
global eta_final eta2 ice_cond ramp_start ramp_stop ramp_rate iaepde
grav1 iaep iaepnl vel1 alimit2 iaepdeinit;
COEF_CLEAN = AM';


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

global simtimek
if ice_cond == 0
   eta = eta_final;
elseif ice_cond == 1
   if time < ramp_start
      eta = 0;
   elseif time >= ramp_start & time <= ramp_stop
      eta = ramp_rate*time;
   else eta = ramp_rate*ramp_stop;
   end
end



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Define Nonlinear Coefficients
% Axial forces
% Nonlinearity due to AOA
COEF_CLEAN(3,1) = 0;
COEF_CLEAN(2,1) = -stuff(22)*180/pi;
COEF_CLEAN(1,1) = -stuff(26);

% Nonlinear elevator effectiveness  (10/9/02)
if ytmp(11) ~= 0
   p = [-7.0498e-010 -4.4111e-008 -6.3960e-007  6.2418e-006  1.7003e-
004 -3.0427e-004];
   CXde = (polyval(p,ytmp(11)*180/pi))*180/pi;
   %COEF_CLEAN(11,1) = CXde;
end

if ytmp(9) ~= 0
   p = [-2.3900e+012  2.4680e+009  1.9581e+008 -1.5525e+005 -
4.2255e+003 -8.7539e-001];
   CXq = (polyval(p,ytmp(9)*2));
   %COEF_CLEAN(9,1) = -CXq;
end


% Normal Force
%Nonlinearity due to AOA
COEF_CLEAN(2,3) = -stuff(23);
COEF_CLEAN(1,3) = -stuff(25);

% Nonlinearity due to elevator  (10/9/02)
if ytmp(11) ~= 0
      p = [-3.4105e-010  1.1457e-008  1.1469e-007 -1.4806e-005
3.5570e-006  6.6608e-003];
   CZde = (polyval(p,ytmp(11)*180/pi))*180/pi;
```

```
   %COEF_CLEAN(11,3) = CZde;
end

%Nonlinear q response
if ytmp(9) ~= 0
   p = [-2.4578e+012 -6.0202e+010  1.5859e+008  3.1347e+006 -
3.5192e+003  3.2525e+000];
   CZq = (polyval(p,ytmp(9)*2));
   %COEF_CLEAN(9,3) = -CZq;
end
% Pitching Moment
COEF_CLEAN(2,5) = stuff(24)*180/pi;
COEF_CLEAN(1,5) = stuff(27);
% Nonlinear elevator effectiveness
if ytmp(11) ~= 0
   p = [1.1701e-008  9.0801e-008 -6.3480e-006  4.0674e-006  6.9649e-004
-2.9388e-002];
   Cmde = (polyval(p,ytmp(11)*180/pi))*180/pi;
   %COEF_CLEAN(11,5) = Cmde;
end
%Nonlinear pitchrate response
if ytmp(9) ~= 0
   p = [-9.9327e+011  1.5793e+010  1.0560e+008 -6.9364e+005 -
1.1468e+003 -2.0615e+001];
   Cmq = (polyval(p,(ytmp(9)*2)));
   %COEF_CLEAN(9,5) = -Cmq;
end
clear stuff
COEF_CLEAN = COEF_CLEAN';

%Compute Iced Coefficients

COEF_ICED = COEF_CLEAN;  %Use only Sam's nonlinear model

clear k f;

%Return the total force coef.
Caero = COEF_ICED*ytmp';

eta_ref_wing = 0.08;
eta_ref_tail = 0.2;

if ice_loc == 1
   eta_wing = eta;
   eta_tail = 1.843.*eta;
elseif ice_loc == 2
   eta_wing = eta;
   eta_tail = 0;
elseif ice_loc == 3
   eta_wing = 0;
   eta_tail = 1.843*eta;
end

ytmp(2) = ytmp(2)*180/pi;
ytmp(11) = ytmp(11)*180/pi;

if ytmp(2) < 16
```

```
   dcl = 0.088449 + 0.004836*ytmp(2) -0.0005459*ytmp(2)^2 +
(4.0859*10^-5)*ytmp(2)^3;
else dcl = -11.838 + 1.6861*ytmp(2) - 0.076707*ytmp(2)^2 +
0.001142*ytmp(2)^3;
end


KCL = -dcl/eta_ref_wing;
dcl = eta_wing*KCL;

dcd = -0.0089 + 0.001578*ytmp(2) - 0.00046253*ytmp(2)^2 - 4.7511e-
5*ytmp(2)^3 + 2.3384e-6*ytmp(2)^4;
KCD = -dcd/eta_ref_wing;
dcd = eta_wing*KCD;

dcma = (-0.01892 - 0.0056476*ytmp(2) +1.0205e-5*ytmp(2)^2 - 4.0692e-
5*ytmp(2)^3 + 1.7594e-6*ytmp(2)^4);
dcmde = (-0.014928 - 0.0037783*ytmp(11) + 0.00039806*(-ytmp(11))^2 -
1.1304e-5*(-ytmp(11))^3 + 1.8439e-6*(-ytmp(11))^4);
DCM = -dcma + dcmde;
KCMA = dcma/eta_ref_wing;
KCMDE = dcmde / eta_ref_tail;
DCM = (0.75*eta_wing + 0.25*eta_tail)*KCMA + eta_tail*KCMDE;

ytmp(2) = ytmp(2)*pi/180;
ytmp(11) = ytmp(11)*pi/180;

DCX = -dcl*sin(ytmp(2)) - dcd*cos(ytmp(2));
DCZ = dcl*cos(ytmp(2)) - dcd*sin(ytmp(2));

%Return the total force coef.
Caero1 = Caero;
Caero1(1,1) = Caero1(1,1) + DCX;
Caero1(3,1) = Caero1(3,1) - DCZ;
Caero1(5,1) = Caero1(5,1) - DCM;
Caero = Caero1;
a_in = ytmp(2);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%5
% Stall angles using  estimative IAEP code

if eta ~= 0
      alimit2 = astall(a_in*180/pi,0,dcl,0);
end
% IAEP Initialization

if iaep == 0
      run solveroliaeplin;
elseif iaep == 1
      if time == round(time) %| time == round(time)+0.25 | time ==
round(time)+0.5
            run solveroliaepnonlin2;
      end
end

clear eta dc* KC* Caero1
clear global Caero1
```

Solverlin.m

```
global ice_loc eta kn deltae_corr iaepde

global Caero1 AM clq1 cda1 clde1 cdde1 cmde1 cma1 cmq1 cm1 cl1 cd1
alimit2 vel1 rho1 theta_corr1 a_in1 grav1

% Calculating the max safe elevator deflection using linearized
approach
vel1 = (uu^2 + vv^2 +ww^2)^(0.5);
theta_corr1 = theta_corr;
a_in1 = a_in;
AM = AM';
clq1 = conv1(AM(9,1),AM(9,3),a_in);
cda1 = conv2(AM(2,1),AM(2,3),a_in);
cdde1 = conv1(AM(11,1),AM(3,1),a_in);
clde1 = conv2(AM(11,1),AM(3,1),a_in);

cmde1 = AM(11,5);
cma1 = AM(2,5);
cmq1 = AM(9,5);

cm1 = Caero1(5,1);
cl1 = conv1(Caero1(3,1),Caero1(1,1),a_in);
cd1 = conv2(Caero1(3,1),Caero1(1,1),a_in);

new = mean(invlap('de_oliaeplin',0.1));

iaepde = new;

AM = AM';
```

Solvernonlin.m

```
global ice_loc eta tspan deltae_corr iaepde iaepnl rho1 alimit2
tspan = [0 2];
b1 = 19.8;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Initial Conditions
ini = zeros(9,1);

%Velocity comp
ini(1) = uu;                          %U;
ini(2) = vv;                          %V;
ini(3) = ww;                          %W;

V1 = (uu^2 + vv^2 + ww^2)^0.5;

%Pitch, Roll, yaw rates
ini(4) = ytmp(8)/b1*2*V1;             %P;
ini(5) = ytmp(9)/b1*2*V1;             %Q;
ini(6) = ytmp(10)/b1*2*V1;            %R;

%Euler angles
ini(7) = phi1;                        %phi;
ini(8) = psi1;                        %xi;
ini(9) = theta_corr;                  %theta;

iaepnl = zeros(1,25);

for i = 1:19
      iaepnl(i) = ytmp(i);
end

iaepnl(29) = XP1;
iaepnl(30) = YP1;
iaepnl(31) = ZP1;
iaepnl(32) = LP1;
iaepnl(33) = MP1;
iaepnl(34) = NP1;



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Control deflections
global dele delf da dr
dele = ytmp(11);
da = ytmp(13);
dr = ytmp(14);
delf = ytmp(12);

%Check for whether elevator is constant or ramped
global AM Caero1

%Thrust components
global Pxi Pyi Pzi Mti Nti Lti ice_loc
Pxi = XP1;
```

```
Pyi = YP1;
Pzi = ZP1;
Mti = MP1;
Nti = NP1;
Lti = LP1;

%Atmospheric components
global rho1 grav1 alpha elev contr
g = grav1;
elev = ytmp(11);

%Solve nonlinear differential equation
options = odeset('RelTol',1e-4,'AbsTol',[1e-6 1e-6 1e-6 1e-6 1e-6 1e-6
1e-6 1e-6 1e-6]);
[t pred] = ode45('oliaepde6',tspan,ini);

%Definition of necessary output parameters

u1 = pred(:,1);
v1 = pred(:,2);
w1 = pred(:,3);

alpha1 = (atan(w1./u1))*180/pi;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if alimit2<max(alpha1)
      solveroliaeplin;
   else iaepde = ytmp(11);
   end
end
```

De_oliaeplin.m

```
function F1 = de_oliaeplin(s);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%     This code calculates the step elevator deflection
%     that corresponds to a given increase in angle of attack
%     These calculations are based on the small perturbation
%     equations of motion.  This code serves as input to a code
%     "invlap.m" which computes inverse laplase transforms numerically.
%
%     Kishwar Hossain
%     02/05/03
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
global kn
% Stability & control derivatives = 0

cdu1        = 0;
cladot1     = 0;
cmadot1     = 0;
ctxu1       = 0;
clu1        = 0;
cmu1        = 0;
cmtu1       = 0;
cmta1       = 0;

% Non-zero stability and control derivatives

global Caero1 AM clq1 cda1 clde1 cdde1 cmde1 cma1 cmq1 cm1 cl1 cd1
alimit2 vel1 rho1 theta_corr1 a_in1 grav1

% Required Inputs

V1          = vel1;
Iy1         = 31027;
q1          = 0.5*(V1^2)*rho1;
m1          = 4600;
g1          = grav1;
theta1      = theta_corr1;
alim1       = alimit2*pi/180-a_in1-2*pi/180;
cbar1       = 1.98;
b1          = 19.8;
S1          = 39.2;

% Longitudinal Dimensional Stability Derivatives (Ramer pg 413)

xu1         = -q1*S1*(cdu1 + 2*cd1)/m1/V1;
xtu1        = q1*S1*(ctxu1 +ctx1)/m1/V1;
xa1         = -q1*S1*(cda1-cl1)/m1;
xde1        = -q1*S1*cdde1/m1;
zu1         = -(q1*S1*(clu1+2*cl1))/m1/V1;
za1         = -(q1*S1*cladot1*cbar1)/2/m1/V1;
zadot1      = -q1*S1*cladot1*cbar1/2/m1/V1;
zq1         = -(q1*S1*clq1*cbar1)/2/m1/V1;
zde1        = -(q1*S1*clde1)/m1;
mu1         = q1*S1*cbar1*(cmu1+2*cm1)/Iy1/V1;
```

```
mtu1        = q1*S1*cbar1*(cmtu1+2*cmt1)/Iy1/V1;
ma1         = q1*S1*cbar1*cma1/Iy1;
mta1        = q1*S1*cbar1*cmta1/Iy1;
madot1      = 0;
mq1         = q1*S1*(cbar1^2)*cmq1/2/Iy1/V1;
mde1        = q1*S1*cbar1*cmde1/Iy1;

% D1 (Longitudinal Airplane Transfer Function Ramer pg 416)

A1          = V1 - zadot1;
B1          = -(V1-zadot1)*(xu1+xtu1+mq1)-za1-madot1*(V1+zq1);
C1          = (xu1+xtu1)*(mq1*(V1-zadot1)+za1+madot1*(V1+zq1))+mq1*za1-
zu1*xa1+madot1*g1*sin(theta1)-(ma1+mta1)*(V1+zq1);
D1          = g1*sin(theta1)*(ma1+mta1-
madot1*(xu1+xtu1))+g1*cos(theta1)*((zu1*madot1+(mu1+mtu1)*(V1-
zadot1))+(mu1+mtu1)*(-
xa1*(V1+zq1))+zu1*xa1*mq1+(xu1+xtu1)*((ma1+mta1)*(V1+zq1)-mq1*za1));
E1          = g1*cos(theta1)*((ma1+mta1)*zu1-
za1*(mu1+mtu1))+g1*sin(theta1)*((mu1+mtu1)*xa1-(xu1+xtu1)*(ma1+mta1));

% Na (Longitudinal Airplane Transfer Function Ramer pg 416)

AN1         = zde1;
BN1         = xde1*zu1+zde1*(-mq1-(xu1+xtu1))+mde1*(V1+zq1);
CN1         = xde1*((V1+zq1)*(mu1+mtu1)-
mq1*zu1)+zde1*mq1*(xu1+xtu1)+mde1*(-g1*sin(theta1)-
(V1+zq1)*(xu1+xtu1));
DN1         = -
xde1*(mu1+mtu1)*g1*sin(theta1)+zde1*(mu1+mtu1)*g1*cos(theta1)+mde1*((xu
1 +xtu1)*g1*sin(theta1)-zu1*g1*cos(theta1));

% Calculating the elevator deflection (Ramer pg 416+417)

D1          = A1*s.^4 + B1*s.^3 + C1*s.^2 + D1*s + E1;
NA1         = AN1*s.^3 + BN1*s.^2 + CN1*s + DN1;

F1          = alim1.*D1./NA1./s;
```

Iaep1.m

```
function ep = iaep1(input)

global kn iaepde iaep simres

if iaep == 0 | iaep == 1
      if input < iaepde
            envp = iaepde;
      else envp = input;
      end
else envp = input;
end

ep = envp;
```

Astall.m

```
function astall=astall(a,x,y,z)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%
% This function is used in conjuction with the clean and iced lift
values   %
% and the angle of attack at with the lift values are extracted
%
%
%
% In the function above: a = angle of attack
%
%                        x = clean lift
%
%                        y = iced lift
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%

%-------------------------------------------------------

deltacl = x - y;
aoa = 0:11;
dclcoeff = [-3.33,-5.09,-4.609,-2.95,-2,-1.61,-1.4015,-1.32,-1.088,-
0.957,-0.915,-0.79];
clmconst = [1,1.15,1.18,1.06,1.03,1.025,1.06,1.3,1.164,1.21,1.26,1.34];
curr_dclcoeff = lineark(aoa',dclcoeff',a);
curr_clmconst = lineark(aoa',clmconst',a);
clmax = curr_dclcoeff*deltacl + curr_clmconst;
astall = 9*clmax+0.516; % Previously the constant = 0.36

%-------------------------------------------------------
function F=lineark(x,y,u)

[nrows,ncols] = size(y);
% Scale and shift u to be indices into Y.
[m,n] = size(x);
u = 1 + (u-x(1))/(x(m)-x(1))*(nrows-1);

if isempty(u), F = []; return, end

siz = size(u);
u = u(:); % Make sure u is a vector
u = u(:,ones(1,ncols)); % Expand u
[m,n] = size(u);

% Check for out of range values of u and set to 1
uout = find(u<1 | u>nrows);
if ~isempty(uout), u(uout) = 1; end

% Interpolation parameters, check for boundary value.
s = (u - floor(u));
u = floor(u);
if isempty(u), d = u; else d = find(u==nrows); end
if length(d)>0, u(d) = u(d)-1; s(d) = s(d)+1; end
```

```
% Now interpolate.
v = (0:n-1)*nrows;
ndx = u+v(ones(m,1),:);
F =  ( y(ndx).*(1-s) + y(ndx+1).*s );

% Now set out of range values to NaN.
if ~isempty(uout), F(uout) = NaN; end
```

if (min(size(F))==1) & (prod(siz)>1), F = reshape(F,siz); end

REFERENCES

[1] Bragg, M.B., Basar, T., Perkins, W.R., Selig, M.S., Voulgaris, P.G., Melody, J.W. and Sarter, N. B., "Smart Icing Systems for Aircraft Safety," AIAA Paper No. 2002-0813, Reno, NV, January, 2002.

[2] Ratvasky, T. P., Blankenship, K., Rieke, W., Brinker, D. J., "Iced Aircraft Flight Data for Flight Simulator Validation," SAE paper no. 2002-01-1528, SAE 2002 General Aviation Technology Conference and Exhibition, April 16-18, 2002, Wichita, Kansas.

[3] National Transportation and Safety Board, "In-Flight Icing Encounter and Uncontrolled Collision with Terrain, Comair Flight 3272, Embraer EMB-120RT, N265CA, Monroe, Michigan, January 9, 1997", NTSB Report Number: AAR-98-04, adopted on 11/4/1998.

[4] Bragg, M.B., "Aircraft Aerodynamics Effects due to Large-Droplet Ice Accretions," AIAA paper no. 96-0932, Reno, NV, Jan. 1996.

[5] National Transportation Safety Board, "Aircraft Accident Report: Inflight Icing Encounter and Loss of Control Simmons Airlines, d.b.a. American Eagle Flight 4184 Avions de Transport Regional (ATR) Model 72-2122, N401AM, Roselawn, Indiana, October 31, 1994," Safety Board Report, *NTSB/AAR-96/01, PB96-910401, Volume 1,* July 1996.

[6] Johnson, C., " Wing Loading, icing and Associated Aspects of Modern Transport Design," Journal of the Aeronautical Sciences, December 1940.

[7] Trunov, O.K., Ingelman-Sundberg, M., " On the Problem of Horizontal Tail Stall Due to Ice, Aerodynamic Background and Comparison Between Results from Wind Tunnel and Flight Tests," Report no. JR-3, 1985.

[8] Trunov, O.K., Ingelman-Sundberg, M., " Methods for Prediction of the Influence of Ice on Aircraft Flying Characteristics," Report no. JR-1, 1977.

[9] Trunov, O.K., Ingelman-Sundberg, M., " Wind Tunnel Investigation of the Hazardous Tail Stall Due to Icing," Report no. JR-2, 1979.

[10] Mikkelsen, K. L., McKnight, R. C., and Ranaudo, R. J., "Icing Flight Research: Aerodynamic Effects of Ice and Ice Shape Documentation With Stereo Photography," AIAA paper 85-0468 also NASA/TM 86906, Reno, NV, Jan., 1985.

[11] Ranaudo, R. J., Mikkelsen, K. L., McKnight, R. C., Ide, R. F., Reehorst, A. L., Jordan, J. L., Schinstock, W. C., and Platz, S. J., "The Measurement of Aircraft Performance and Stability and Control After Flight Through Natural Icing Conditions," AIAA paper 86-9758 also NASA/TM 87265, Las Vegas, NV, April 1986.

[12] Ratvasky, T. P., Van Zante, J. F., and Riley, J., T., "NASA/FAA Tailplane icing Program Overview," AIAA Paper No. 99-0370, Reno, NV, January 1999.

[13] Van Zante, J. F., Ratvasky, T. P., Investigation of Dynamic Flight Maneuvers with an Iced Tailplane," AIAA Paper No. 99-0371, Reno, NV, January 1999.

[14] S. Lee, M.B. Bragg, "Effects of Simulated-Spanwise-Ice Shapes on Airfoils: Experimental Investigation," AIAA Paper No. 99-0092, Reno, NV, January 11-14, 1999.

[15] S. Lee, H.S Kim, and M.B. Bragg, "Investigation of Factors that Influence Iced-Airfoil Aerodynamics," AIAA Paper No. 2000-0099, Reno, NV, January 2000.

[16] Raymer, D., "Aircraft Design: A Conceptual Approach," AIAA Education Series, AIAA.

[17] North, M. D., "Finding Common Ground in Envelope Protection Systems," Aviation Week and Space Technology, August 28, 2000.

[18] National Transportation Safety Board, "Aircraft Accident Report: Inflight Icing Encounter and Loss of Control Simmons Airlines, d.b.a. American Eagle Flight 4184 Avions de Transport Regional (ATR) Model 72-2122, N401AM, Roselawn, Indiana, October 31, 1994," Safety Board Report, *NTSB/AAR-96/01, PB96-910401, Volume 1,* July 1996.

[19] Horn, J., Calise, A. C., Prasad, J. V. R.., and O'Rourke, M., "Flight Envelope Cueing on a Tilt-Rotor Aircraft Using Neural Network Limit Prediction," American Helicopter Society 54th Annual Forum, Washington, D. C., May 1998.

[20] Bragg, M.B., Perkins, W.R., Sarter N.B., Basar, T., Voulgaris, P.G., Gurbacki, H.M., Melody, J.W. and McCray, S.A., "An Interdisciplinary Approach to Inflight Aircraft Icing Safety," AIAA Paper No. 98-0095, Reno, NV, January 12-15, 1998.

[21] Bragg, M.B., Hutchison, T., Merret, J., Oltman, R., and Pokhariyal, D., "Effects of Ice Accretion on Aircraft Flight Dynamics," AIAA Paper No. 2000-0360, Reno, NV, Jan. 2000.

[22] Pokhariyal D., Bragg M. B., Hutchison, T., Merret, J., "Aircraft Flight Dynamics with Simulated Ice Accretion," AIAA Paper No, 2001-0541, Reno, NV, Jan.2001.

[23] Rauw, M., "FDC 1.3–A SIMULINK Toolbox for Flight Dynamics and Control Analysis," 1998.

[24] Pokhariyal, D., "Effect of Ice Accretion on Aircraft Performance and Control During Trimmed Flight," M.S. Thesis, University of Illinois, Urbana, IL, December 2000.

[25] Melody, J. W., Pokhariyal, D., Merrett, J., Basar, T., Perkins W., Bragg, M., "Sensor Integration for Inflight Icing Characterization using Neural Networks," AIAA Paper No. 2001-0542, Reno, NV, January 8-11, 2001.

[26] Sharma, V. and Voulgaris, P., "Effect of Icing on the Autopilot Performance," AIAA Paper 2002–0815, Aerospace Science Meeting and Exhibit, Reno, NV, January 2002.

[27] Sharma, V., "Twin Otter Autopilot Analysis, Design and Envelope Protection for Icing Conditions," Master's Thesis, University of Illinois, Urbana-Champaign, August 2002.

[28] Sehgal, B., Deters, R. and Selig, M.S., "Icing Encounters Flight Simulator," AIAA Paper 2000-0817, Reno, NV, January 2002.

[29] Bragg, M. B., Hutchinson, T., Merret, J., Oltman, R., and Pokhariyal, D., "Effects of Ice Accretion on Aircraft Flight Dynamics," AIAA Paper No. 2000-0360, Reno, Nv, January 2000.

[31] Kim, H. S., and Bragg, M. B., "Effects of Leading Edge Ice Accretion Geometry on Airfoil Aerodynamics," AIAA paper 99-3150, Norfolk, VA, July 1999.

[33] Lee, S., Bragg, M., B., "The Effect of Ridge-Ice Location and the Role of Airfoil Geometry," AIAA paper 2001-2481, Reno, NV, January 2001.

[34] Sharma, V., "Twin Otter Autopilot Analysis, Design and Envelope Protection for Icing Conditions," Master's Thesis, University of Illinois, Urbana-Champaign.

[35] Etkin, B., "Dynamics of Atmospheric Flight," John Wiley and Sons, Inc., New York, 1972.

[36] Dormand, J. R. and P. J. Prince, "A family of embedded Runge-Kutta formulae," *J. Comp. Appl. Math.*, Vol. 6, 1980, pp 19-26.

[37] Roskam, J., "Airplane Flight Dynamics and Automatic Flight Controls," Roskam Aviation and Engineering Corporation, 1982.

[38] Hollenbeck, K. J. (1998) INVLAP.M: A matlab function for numerical inversion of Laplace transforms by the de Hoog algorithm,
http://www.isva.dtu.dk/staff/karl/invlap.htm

[39] Ratvasky, Tom, (Private Communication), 2002.