STABILITY CHARACTERISTICS OF A COMBAT AIRCRAFT WITH CONTROL SURFACE FAILURE

Thesis

Captain Stephen M. Zaiser

AFIT/GAE/ENY/89D-42

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GAE/ENY/89D-42

(1)

STABILITY CHARACTERISTICS
OF A COMBAT AIRCRAFT
WITH CONTROL SURFACE FAILURE
Thesis
Captain Stephen M. Zaiser
AFIT/GAE/ENY/89D-42

Approved for Public Release; Distribution unlimited

STABILITY CHARACTERISTICS OF A COMBAT AIRCRAFT WITH CONTROL SURFACE FAILURE

Thesis

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University

in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Aeronautical Engineering

By

Stephen M. Zaiser, B.S.

Captain, USAF

November, 1989

Approved for Public Release; Distribution unlimited

Acknowledgements

Performing the work associated with this thesis has been one of those life experiences that reinforces in my mind the fact that many people have had a part in helping me to be where I am today. As I look back there is a long unbroken line; I have been very blessed. Words on a piece of paper are small thanks for the contributions the people acknowledged on this page and a host of those who will go unnamed have made to this thesis and my life.

To my committee. Many thanks to Major Lannie Hudson who provided the initial kick in the pants that I needed to get off and running and who toiled with almost endless patience to help me understand what I should have known already. Major Hudson also provided most of the FORTRAN code that went into developing the curve fitting routines. Thanks. To Captain Curt Mracek, you helped me to remember the important things, like when you push forward on the stick the nose should go down! I have appreciated your inputs and timely support. Captain Brett Ridgely, thank you for the insights you have given both as one of my instructors and a committee member.

To my Advisor. Dr Spenny, thank you for taking an active interest in this problem and encouraging me at some very key times. I will always carry fond memories of our lengthy "debates" over Euler angles and other issues related to this project. By the way, I still owe you one! Please accept my very genuine thanks.

To Don and Caron Ward. Don and Caron, you took an assortment of text files, plot files, scrap paper, and not a few odd ideas and transformed them into a thesis. Thank you for a labor of love and for a freindship that I treasure.

To my Class Mates. Thanks for making life at AFIT not only bearable but down right interesting and enjoyable. If it had not been for the encouragement some of you provi[illegible] robably would not have made it far enough to be writing this page right now.

To Jim and Marilyn Zaiser (ie Mom and Dad). Mom and Dad, you have taught me most of what I hold to be dear and important. Your encouragement, support, love, and just who you both are have been a well spring in my life. I love you.

And finally, I express my deepest gratitude to Jesus Christ to whom I owe my very life. Sola Deo Gloria!

Stephen M. Zaiser

STABILITY CHARACTERISTICS OF A COMBAT AIRCRAFT WITH CONTROL SURFACE FAILURE

Table of Contents

List of Figures

List of Tables

List of Symbols

α	Angle of Attack
AOA	Angle of Attack
β	Side Slip Angle
δ	Control Surface Deflection
θ	Pitch Angle
ϕ	Roll Angle
ψ	Heading Angle
γ	Flight Path Angle
$\bar{c}$	Aerodynamic Chord
MAC	Mean Aerodynamic Chord
S	Reference Wing Area
b	Reference Wing Span
q	Dvnamic Pressure
V	Free Stream Velocity
U	Body X axis component of Velocity
V	Body Y axis component of Velocity
W	Body Z axis component of Velocity
L	Lift
D	Drag
Y	Side Force
l	Rolling Moment
m	Pitching Moment
n	Yawing Moment

P	Roll Rate (Body X axis)
Q	Pitch Rate (Body Y axis)
R	Yaw Rate (Body Z axis)
X	Body X axis Force Component
Y	Body Y axis Force Component
Z	Body Z axis Force Component
$\overline{P}$	Linear Momentum Vector
$\overline{H}$	Angular Momentum Vector
$\underset{\sim}{I}$	Inertial Dyad
LEF	Leading Edge Flaps
LLE	Left Leading Edge Flap
RLE	Right Leading Edge Flap
LFL	Left Flaperon
RFL	Right Flaperon
LHT	Left Horizontal Tail
RHT	Right Horizontal Tail
RUD	Rudder
TEU	Trailing Edge Up
TED	Trailing Edge Down
GW	Aircraft Gross Weight
KEAS	Knots Equivalent Air Speed
M	Mach Number
g_c	Gravitational Constant
C_L	Lift Coeficient
C_D	Drag Coeficient
C_Y	Side Force Coeficient

C_l Rolling Moment Coeficient

C_m Pitching Moment Coeficient

C_n Yawing Moment Coeficient

ABSTRACT

In this thesis, an investigation of the stability characteristics of an aircraft which has sustained damage to a primary control surface was performed. The analysis was performed using wind tunnel data taken on an F-16 model in a test conducted by Turhal [12]. The coupled, non-linear, aircraft equilibrium equations for constant altitude, rectilinear flight were derived. The aircraft stability and control derivatives were developed and analyzed to identify aerodynamic coupling with implications for an aircraft with failed control surface(s). Three control schemes which allow for progressively greater independence among the control surfaces were formulated for use in the evaluation of an aircraft with an actuator failure of the rudder. The investigations were conducted at two flight conditions representative of the aircraft at cruise and landing approach velocities. Regions in α / β space where equilibrium is obtainable were investigated to identify remaining control authority, drag characteristics, and aircraft orientation. The matrix decomposition techniques of Singular Value Decomposition and the Row Reduced Echelon Form of the augmented matrix were used to provide additional insight into the interrelationship of the control surfaces at different points within the defined trim region.

STABILITY CHARACTERISTICS OF A COMBAT AIRCRAFT WITH CONTROL SURFACE FAILURE

I. INTRODUCTION

Control. It is the essence of practical aerospace flight and has long been recognized as one of the difficult technical challenges to be addressed as aircraft have gained improved performance. Modern high performance aircraft have many costly and intricate devices onboard which have the sole purpose of either enabling the pilot to maintain control of the aircraft or making the task of controlling the aircraft easier. Yet as Rubertus has noted, [11:1280], these systems presuppose the availability and functionality of all the control surfaces that they have been designed to employ. In the event that a control surface is damaged or lost the control law which has been designed to make control of the aircraft possible has ceased to be valid. He further notes that up to 20 percent of the aircraft lost in combat have been lost due to damage to the aircrafts Flight Control System (FCS).

In recent years, several methodologies have been advanced under the broad category of Reconfigurable Flight Control Systems (RFCS) to address the problem of damage to or failure of one or more control surfaces. That is, techniques that will assess the location and nature of the damage to the control surface(s) and reconstruct the FCS control law so that the aircraft can continue to fly. The degree to which these techniques are successful obviously has massive ramifications for aircraft flight safety, sortie generation in a combat environment, and reliability and maintainability. Most important, of course, is the return of a pilot who otherwise would have been lost.

In his paper "Self-Repairing Flight Control Systems Overview" [11:1285], Rubertus makes the following comments,

> Analysis must be performed to better define the aircraft characteristics in an impaired state. An aircraft with a jammed, floating, or missing control surface will exhibit stability characteristics different than a normal aircraft. The cross-coupling effects are expected to be significant. Are the cross-coupling terms (driven to zero or into second and third order effects in current designs) changing sufficiently to be-

come first order effects? Neither current models nor wind tunnel data define what these effects are. Until the effects are better defined, understood, and included in the analyses, the full impact of control reconfiguration will not be known.

The object of this thesis is to provide a greater understanding of the stability characteristics of an aircraft with damaged control surfaces.

Problem Definition

In the event that a control surface is damaged or becomes inoperable several negative effects might be encountered. First, the FCS has lost the use of the control power of the failed surface to effect control over the attitude of the aircraft. For example, in the event that the right aileron is lost the pilot now has only half of the authority to perform a rolling maneuver that was present before the failure. A second effect is the introduction of coupling effects between the longitudinal and lateral modes of the aircrafts motion. The loss of half of the horizontal tail, for instance, would have the result that when the pilot commanded a pitching moment, the aircraft would also experience unwanted, and unexpected, yawing and rolling moments and possibly side force. Thus, not only has the pilot's maneuvering ability been reduced, perhaps substantially, but also the introduction of coupling makes it necessary for him to fly an aircraft with which he is unfamiliar. And of course in a combat environment all this may be occurring at a time when his attention is required for other tasks [8:3].

There is yet a third effect that becomes most prominent in the event of a control surface actuator failure that results in the control surface being locked into a position other than zero. The "hardover" failure of a control surface not only introduces the complications already noted but it also generates substantial forces and moments which must now be overcome by the remaining "healthy" surfaces in order to prevent departure of the aircraft. The question arises quite naturally that, given a prescribed failure, is it possible to maintain the aircraft in an equilibrium or trimmed state? This thesis seeks to address that question, to provide a better understanding of the nature of the problem and the means available for addressing it.

Previous Work

Raza, [8], investigated techniques for modifying the control laws to compensate for the failure of either a flaperon or a horizontal tail element. His linear model employed the use of constant coefficient control derivatives. His model assumed that only small perturbations away from the nominal trim condition would occur as a result of the control surface failure. Although limited to small deflections the analysis did incorporate the coupling of the longitudinal and lateral modes and the introduction of perturbation forces and moments by the failed surface. Reconfigurable Flight Control techniques were investigated using the AFTI F-16 as an aircraft model by Eslinger [1]. Eslinger investigated a failure of the aircrafts right horizontal tail such that the tail was left free floating in the airstream. As he notes, [1,4] the failed control surface in this case does not generate undesirable forces and moments. Eslinger's aircraft model utilized constant aerodynamic derivatives at the selected flight conditions. Weiss et al, [13], investigated a technique for automatically trimming an aircraft where the failure of the control surface is treated as the introduction of a disturbance away from the nominal trim condition. Their paper contains a rigorous definition of the linear trim problem [13:402]. Although the analysis they present deals with the runaway trim of the aircraft stabilator they point out that the failure of the rudder represents the most difficult single control surface failure to be addressed, [13,405].

In 1986, Turhal, [12], conducted wind tunnel tests to investigate the effect of various types of control surface failures on an aircrafts aerodynamic stability derivatives. The tests were conducted in the AFIT five foot wind tunnel using a one-twentieth scale model of a F-16. Three configurations of the model were tested, with each configuration representing a potential failure mode.

The data collected by Turhal has several interesting features. One feature of interest is that the data includes information regarding the coupling of the aerodynamic stability derivatives as the aircraft is placed in an unsymmetric orientation; β nonzero. Secondly, the force and moment coefficients are recorded for the deflection of a single control surface. For example, the flaperons are usually deployed asymmetrically as ailerons and the rolling moment for the total aileron would be recorded. Simply recording the data for total aileron might mask the presence of coupling that is of interest when the surfaces are deployed independently. In Turhal's tests, the effect of sideslip angle and Angle of Attack (AOA) on the right flaperon, for instance, is imbedded in the data recorded in the tests. Consequently, the control derivatives developed for use in the present thesis will be functions of AOA and sideslip angle rather then constants developed for the aircraft at a specified trim condition.

At the conclusion of his thesis, Turhal made several recommendations for follow-on work based on the test data that he had recorded, [12:62]. First, he stated that the optimization studies performed to find trim conditions for the "damaged"aircraft had yielded unsatisfactory results. He postulated that the problem may have been related to the curve fitting that formed on the wind tunnel data. Second, he suggested that other means of investigating "optimum" trim conditions be explored. Third, he recommended that a more comprehensive study of the data should be performed numerically to identify any significant phenomena which might be present.

Purpose

This research will encompass a thorough investigation of the stability characteristics of an aircraft which has sustained damage to a primary control surface. The presence of significant aerodynamic coupling will be identified and the interrelationship of the aircraft control derivatives, which are developed as functions of Angle of Attack (α) and sideslip angle (β), will be examined. As a means of gaining insight into the nature of the damaged aircraft the following questions will be addressed:

A: For a stated flight condition and control surface failure, can a state of equilibrium be achieved using the remaining functional surfaces?

B: If equilibrium is achievable, how large is the region in α/β space in which equilibrium may be obtained? Questions regarding the orientation of the aircraft and the use of available control authority to achieve this state will also be addressed.

C: Will the use of more advanced control schemes, i.e. allowing the control surfaces currently on the aircraft to act with greater independence, significantly augment the equilibrium region and/or improve other aircraft characteristics with-in this space?

Approach

To accomplish the stated purposes of this thesis several specific tasks are accomplished and represent the major sections of the thesis. The data collected by Turhal is placed in to a functional form that can be used to perform the desired analysis. In general, these functional representations of the force and moment coefficients are nonlinear in α and β, and so the restriction of constant coefficients is not a limitation imposed on the analysis performed in this thesis. Contour plots of the basic aerodynamic coefficients are constructed to identify any significant aerodynamic coupling which might impact the trim investigations. The relative authority of each control surface for each force and moment is also examined to identify the significance of each surface for achieving trim and for answering the question of whether the relative importance of the surfaces changes at different points in α/β .space

An actuator failure of the rudder is assumed to represent the most significant single primary control surface failure. This assumption is consistent with the findings of Weiss [13:405]. This failure mode is investigated at two flight conditions which are deemed to be representative of two phases of the aircrafts flight profile. The equilibrium equations for constant altitude, rectilinear flight are solved to identify points in α/β space where an equilibrium state is achievable for a specified degree of rudder failure. Three different control schemes of increasing complexity are employed to investigate how significantly the equilibrium region can be altered by employing greater degrees of freedom in the use of the available control surfaces.

Two math techniques are used to provide a greater insight into the nature of the problem being addressed. Singular Value Decomposition (SVD) and the Row Reduced Echelon Form (RREF) are used to analyze the problem. Restructuring the problem via these techniques provides useful information regarding not only the null space of the problem, but also illuminates the interaction of the various control surfaces in achieving a solution to the equilibrium problem.

Presentation

The analysis performed in this thesis is presented in the following chapters. Chapter II details the techniques used to transform the data collected by Turhal into polynomial functions which can be used for the equilibrium analysis. Observed aerodynamic coupling of the control and aircraft stability derivatives is detailed in Chapter III. The relative significance of the control surfaces is also discussed in this chapter. Chapter IV outlines the formulation of the nonlinear equations of motion into the form that is used to identify the regions of equilibrium for control surface failure. The results of this analysis are presented and discussed in Chapter V and Chapter VI outlines a summary of the results of this research and recommendations for further study.

II DATA PREPARATION

Introduction

The analysis performed in this thesis is based on wind tunnel data collected by Turhal, [12], for a Master's thesis in 1986. The data preparation phase of the current research involved the formation of functional representations of the stability derivative data collected in Turhal's wind tunnel work. A least squares curve fitting technique was used to develop polynomial functions which describe the aircraft stability derivatives. Since the equilibrium analysis was a static analysis the dynamic derivatives of the aircraft were not estimated. In this chapter a short description of the F-16 is given along with a brief discussion of the tests conducted by Turhal. The functional form of the equations used to describe the aircraft stability derivatives and the techniques used to develop them are also discussed.

Aircraft Description

The F-16 is a single engine, low aspect ratio, fighter aircraft currently in the inventory of the USAF. There are seven control surfaces located on the aircraft which are of interest for the studies to be performed in this thesis: right and left Leading Edge Flaps (LEFs), right and left Flaperons, right and left Horizontal Tails, and the rudder. The following paragraphs provide a short discussion of these control surfaces and their significance for the trim study. The location of each of the control surfaces may be identified by referring to Figure 1. A detailed discussion of the F-16 may be found in the open literature in Jane's, [4:345].

Leading Edge Flaps (see following page): The LEFs primary function is to vary the camber of the wing; causing the lift curve to slide to the right as they are deployed. The net effect of this is to cause $C_{L_{max}}$ to occur at higher AOA then would be experienced by the clean wing.

The LEFs are designed to deploy in a symmetric fashion and their deflection is scheduled as a function of AOA and Mach number. It should be noted that the pilot does not exercise direct control over the LEFs and so, as they are employed on the current aircraft, they are not truly a control surface.

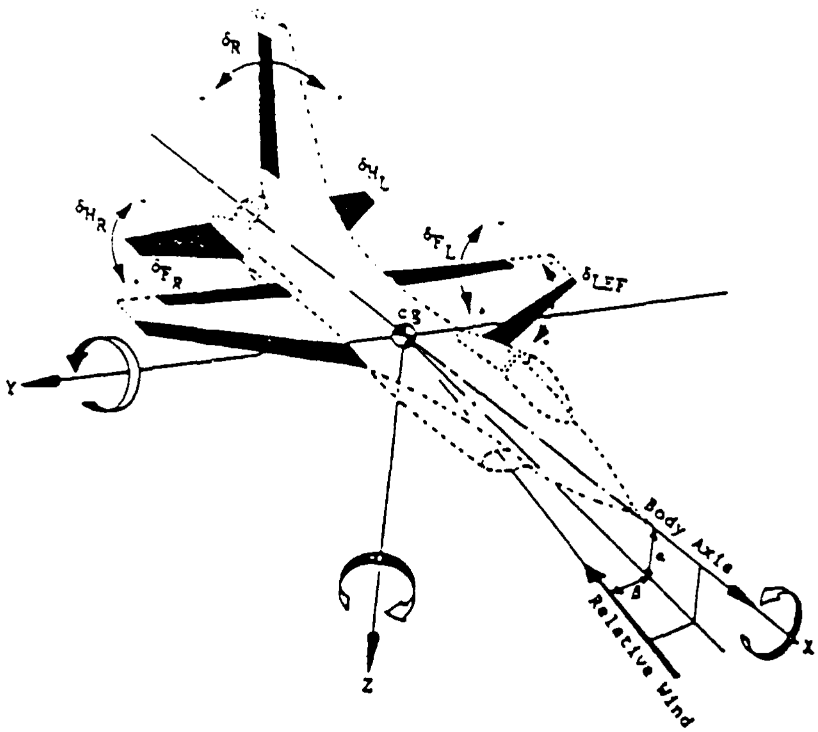


Figure 1 F-16 Control Surfaces

Flaperons: When deployed as flaps the flaperons provide direct lift to the aircraft and also some pitching moment. For control purposes, however, the pilot's stick can only command the flaperons to deflect asymmetrically or as ailerons. The flaperons therefore are the primary means by which rolling moment is applied to the aircraft to execute banking maneuvers.

Horizontal Tails: The horizontal tails, on the other hand, can be employed via the pilot's stick in two fashions. If the differential tails are deployed symmetrically they act as an elevator and are used to exert pitching moment on the aircraft. The tails can also be deployed in an asymmetric manner to augment the rolling moment generated by the flaperons. Simply stated, pulling back on the control stick will result in the symmetric deflection of the horizontal tails and will pitch the nose of the aircraft up. Pushing the stick to the side will result in asymmetric deflection of the flaperons and tails resulting in a rolling maneuver about the axis of the aircraft.

Rudder (see previous page): The rudder is employed in the same fashion as on a conventional aircraft and is the primary control surface available for yawing the aircraft.

Sign Convention

The following sign convention is adopted for use in this thesis:

1. For the flaperons and the horizontal tails positive deflection is defined Trailing Edge Down (TED).

2. For the leading edge flaps positive deflection is defined to be Leading Edge Down (LED).

3. Positive deflection of the rudder will be defined as deflection of the rudder toward the left side of the aircraft. looking forward (port)

4. Positive sideslip angle, β, is defined for the free stream velocity vector approaching from the right side of the aircraft nose. looking forward (starboard)

5. All the aircraft control and aerodynamic coefficients are recorded in the aircraft Stability axis system.

F-16 Wind Tunnel Data

In 1986 Turhal, [12], conducted wind tunnel tests to investigate the effect of various types of control surface failures on an aircrafts aerodynamic coefficients. The tests were conducted in the AFIT five foot wind tunnel using a one twentieth scale model of a F-16A. All of the tests were conducted at low speeds, holding Mach number at approximately 0.118 and dynamic pressure at 20 pounds per square foot. For a detailed discussion of the experimental procedure used in recording the test data see [12:25-34]

Three configurations of the model were tested, with each configuration representing a potential failure mode. The first configuration had all the control surfaces but one fixed at a zero deflection angle. The remaining control surface was then placed at a specified deflection and the resulting forces and moments were recorded. In the second configuration, the left flaperon was allowed to float free. The remaining surfaces were then cycled through their deflection ranges. As in the first configuration, only one control surface was deflected at a time. The final configuration, had the left flaperon removed from the model entirely. As in the prior tests, the effects of the deflection of the remaining control surfaces on the forces and moments was then observed. The aerdynamic coefficients calculated by the wind tunnel data reduction program were recorded in the Stability Axis system.

For each of the configurations noted above the wind tunnel data has been placed into data sets. The "zero" case represents the data collected when the models controls were all set at zero deflection and the model was placed at various angles of attack and side slip angles. The same procedure was used to develop data sets for the right and left leading edge flaps, the right flaperon, the right horizontal tail, and the rudder. For the configurations where the left flaperon was floating free or missing a data set was also developed for the left horizontal tail.

Aerodynamic Forces

The data which is output by the wind tunnel data reduction program are the total aircraft force and moment coefficients. These coefficients are a non-dimensional representation of the forces and moments experienced by the aircraft at given a AOA and side slip angle. The aerodynamic coefficients may be converted into forces and moments in the aircraft Stability axis system via the following relationship:

$$L_s = C_L \bar{q} S \tag{2.1}$$

$$D_s = C_D \bar{q} S \tag{2.2}$$

$$Y_s = C_Y q \bar{S} \tag{2.3}$$

$$\ell_s = C_\ell \bar{q} \bar{S} b \tag{2.4}$$

$$M_s = C_M \bar{q} S \bar{c} \tag{2.5}$$

$$N_s = C_N \bar{q} S b \tag{2.6}$$

The appropriate reference data for the full scale aircraft is given in [12:27], and is represented

Table 1 F-16 Reference Data

Wing Area	S	300 Sq Ft
Span	b	29ft
MAC	$\bar{C}$	10.94ft
Cg	ω	0.35MAC

here in Table 1. By necessity, the data collected in the wind tunnel is taken at a finite number of discrete data points. Turhal's wind tunnel data, in general, is a function of three variables; that is, the force and moment coefficients are recorded for a specific setting of angle of attack, sideslip angle, and single control surface deflection. Since the analysis performed in this research will require data at points other than those points at which experimental data was collected some functional representation of the data is required. A polynomial is selected as the functional form which will be used to describe the data. Each aircraft force or moment coefficient may then be described with a polynomial of the following form:

$$C_f = \sum_{j=0}^{J} \sum_{i=0}^{I} A_{ij} \, \alpha^i \beta^j + \sum_{\ell=1}^{7} \sum_{m=0}^{M} \sum_{n=0}^{N} B_{\ell mn} \, \alpha^n \beta^m \delta_\ell \qquad (2.7)$$

Note that in general the polynomial will be nonlinear but that δ will always be held to a first power.

Force and Moment Coefficients

Turhal's test included recording force and moment coefficients where all of the control surfaces were held at zero deflection and α and β were varied. Equation (2.7) shows that the polynomial used to predict the total force or moment coefficient is composed of two summation terms. The first of these represents the coefficient strictly as a function of α and β and should describe the wind tunnel data taken when all of the control surfaces were held at zero. The coefficients, Aij, associated with each polynomial term were obtained by performing a least squares curve fit on this "zero" case data. A short discussion of the theory and mathematics involved in the least squares curve fitting technique may be found by referring to Appendix A.

To accomplish the three dimensional curve fitting of the wind tunnel data a FORTRAN computer code, POLYFITA, was written which will read in the data files compiled by Turhal, request the order of the polynomial and perform the curve fit. Appendix B contains the FORTRAN codes used to accomplish the curve fits. Two measures of the "goodness" of the selected polynomials fit of the data were employed to determine the suitability of the polynomial for use in the future analysis. The first measure of the accuracy of the fit was the calculation of correlation coefficient, r^2, for each fit of the data.

$$r^2 = 1 - \frac{\sum_{k=1}^{npts} \left[C_{f_{exp_k}} - C_{f_{anal_k}} \right]^2}{\sum_{k=1}^{npts} \left[C_{f_{exp_k}} - C_{f_{mean}} \right]^2} \tag{2.8}$$

This measure of merit provided a means for estimating how well the polynomial fit captured the variation in the experimental data. It is possible, through the use of a polynomial of high enough order, to obtain a curve fit which will pass through each data point. This polynomial will accurately predict the value of the data at the point at which the data was collected but its behavior between points may be very ill behaved. The second measure of merit for the curve fits provides a means for avoiding the selection of such a function. Primarily qualitative, this second measured involved the construction of graphs and contour plots. The graphs, for example Figure 2, provided a direct comparison of the polynomial fit with the data collected in the tunnel. An evaluation of the curve with respect to the expected behavior of the force or moment coefficients could also be made. For example, the lift coefficient should be linear in α, the drag a parabolic function of α etc.. It should be noted, however, that a graph such as Figure 2 requires that the two remaining variables be held constant to see this "slice" of the curve in the three dimensional variable space. For this reason, the data was also plotted as contour plots so that the behavior of the data as a function of two variables could be observed. An example of such a contour plot is Figure 3 and a complete set of these plots may be found in Appendix D. .

The curve fits of all the "zero" case data were accomplished with the noted computer codes and applying the following criteria.

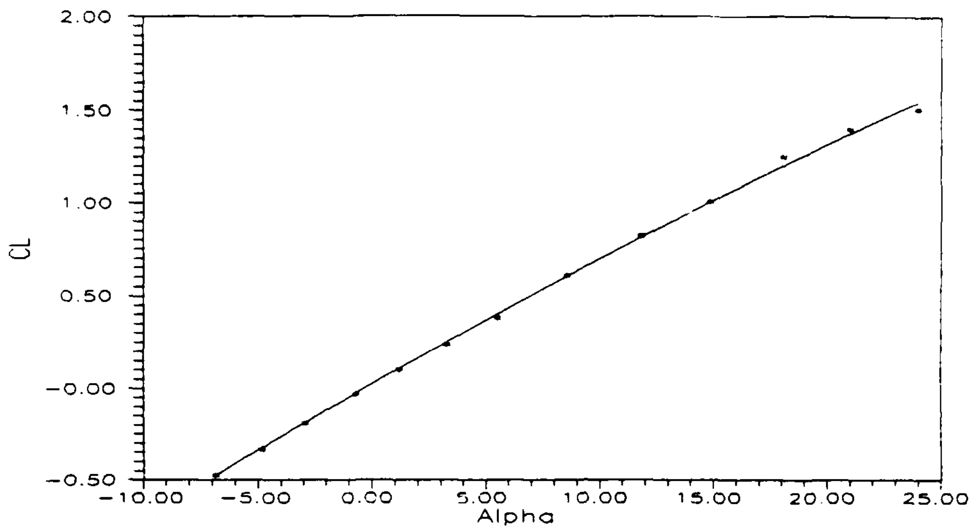


Figure 2 Lift Coefficient -vs- α

- 1.Keep it accurate. The accuracy of the curve fits was established by trying to achieve very high r^2 value, $.95 \leq r^2 \leq 1.0$, and by evaluating the graphs and contour plots.
- 2. Keep it simple. To avoid future numerical problems, and the undesirable behavior noted above the lowest order polynomial which provided a reasonable level of accuracy was selected.

Aircraft Control Derivatives

The second summation contained in equation (2.1) represents the contribution of all the control surfaces to the total force or moment coefficient. The polynomial associated with each control surface is in effect the control derivative associated with that surface. The experimental method employed for collecting the derivative data assumed that the effects of each control surface could be added together with the "zero" case to obtain the total aircraft force or moment coefficient. The assumption that the superposition principle may be applied is premised on linear terms in δ. For this reason, all of the control derivatives were developed holding the δ term to a first power.

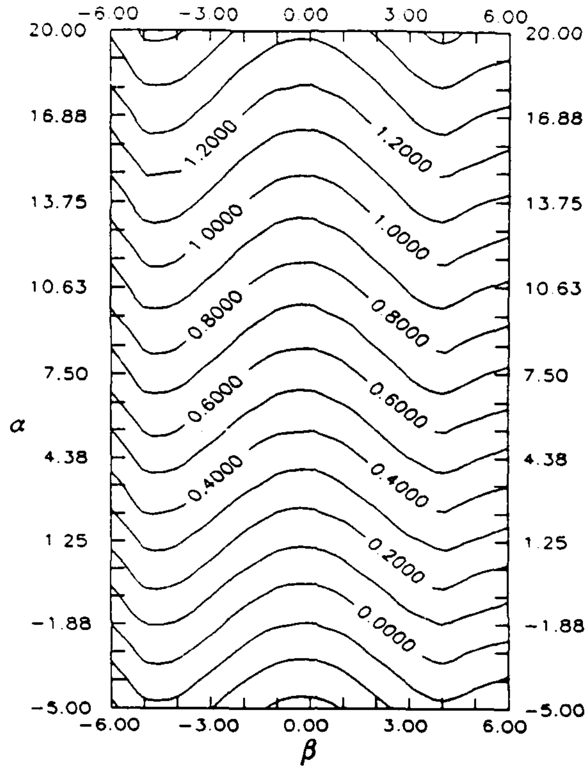


Figure 3 Contours of Constant Lift Coefficient

The control derivatives were assumed to be of the form

$$\left[C_{f_{\alpha\delta}} \alpha + C_{f_{\beta\delta}} \beta + C_{f_{\delta}} \right] \delta \tag{2.9}$$

which is a linear equation once α and β have been specified. To obtain the coefficients contained in equation (2.9) the stability data contained in the data sets associated with the respective controls was curve fit using the program POLYFITB;(see Appendix B). Here the effect of the deflection of the specified control surface is treated as a perturbation of the force or moment above, or below, the force or moment experienced by the model with the surfaces set to zero. Hence, the function supplied to the least squares routine for fitting was the polynomial form arrived at for the "zero" case plus the terms in equation (2.10). The coefficient which were related to control deflections were then stripped off to become the descriptors of that control derivative. The r^2 value for each fitting of the control sur-

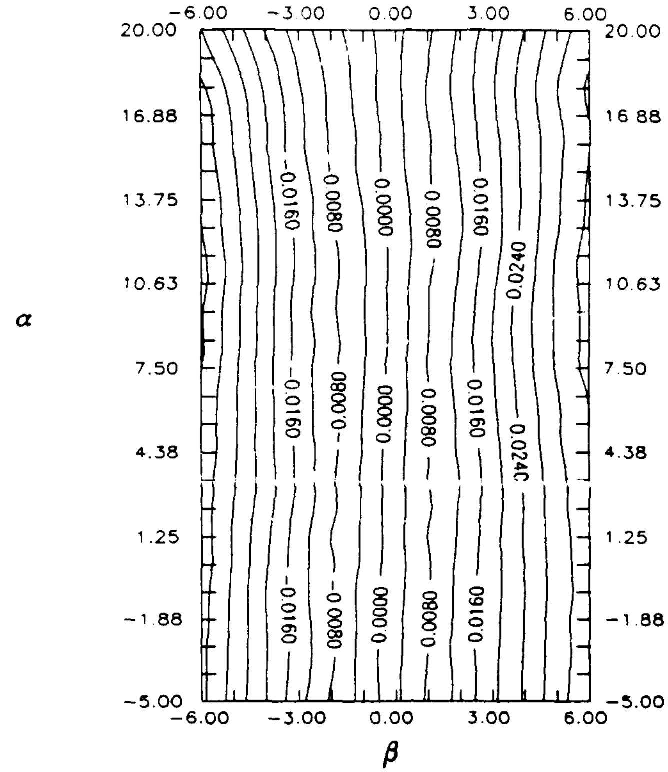


Figure 4 Contours of Constant C_N

face data sets was compared to a fitting performed with only the "zero" case polynomial terms to insure that the effect of the control surface was reasonably well represented. This was indicated by a significant rise in the value of the correlation coefficient when the δ terms were added to the polynomial. The control derivative predictor equations are presented in Appendix C.

Lateral Bias

In the initial phases of conducting the trim analysis it became evident that the aircraft was developing significant lateral forces and moments at zero AOA, zero side slip angle, and zero control deflections. This bias in the lateral data may be seen by observing Figure 4 where the yawing moment coefficient does not take on a zero value at β and α equal to zero. For an aircraft which is geometrically symmetrical about the X-Z plane of the aircraft the forces and moments should be zero at this zero condition, [10:139-156]. In light of this, the predictor equations for the aircraft lateral coefficients were modified to remove this unresolved bias. The modification was effected by setting the constant term in each lateral equation equal to zero. The corrected predictor equations are the ones listed in Appendix C.

Summary

In the data preparation phase of the thesis the wind tunnel data generated by Turhal was placed into functional forms for later use in the analysis. These functional representations of the aircraft control and stability derivatives were formed as polynomials which in general are nonlinear in α and β. Lateral biassing in the wind tunnel data was identified and appropriate changes accomplished to correct this anomaly.

III COUPLING OF AERODYNAMIC DERIVATIVES

Introduction

The polynomial equations developed in Chapter II to describe the behavior of the aircraft control and stability derivatives are nonlinear functions in α and β. Through these terms coupling may be introduced between the longitudinal and lateral modes of the aircraft. A longitudinal coefficient, such as the pitching moment for instance, may be found to have a significant dependence on side slip angle. Further, the control derivatives, which are usually treated as constants for a given flight condition, may in fact exhibit a dependence on α and β which should be noted. Coupling as defined in this thesis does not refer to inertia effects or the interaction of the various control surfaces. In this research, coupling refers to two specific effects. First, coupling indicates the presence of stability derivatives which couple the effect of AOA and sideslip angle together. Second coupling occurs when the failure of a control surface imparts forces and moments to the aircraft which are not usually associated with that surface. As was noted in Chapter I, Rubertus makes the following comments, " ...The cross-coupling effects are expected to be significant. Are the cross-coupling terms (driven to zero or into second and third order effects in current designs) changing sufficiently to become first order effects? " This chapter seeks to explore this question and its attending implications for the equilibrium analysis addressed in this thesis.

Aircraft Stability Derivatives

The contour plots of the force and moment coefficients developed in Chapter II provided the primary means by which coupling was identified. A complete set of the plots may be found in Appendix E. Note that there are two plots for each coefficient. The plots labeled "EXP" represent a contour plot of the experimental data. Plots that are labeled "CF" represent plots of the polynomial fit of the experimental data.

Figure 5 represents the variation of the drag coefficient as a function of α and β. Note that a function that is strictly dependent on α would result in contours that intersect the α axis perpendicularly. Conversely, a strict dependence on β has contours which show no variation as one moves along the

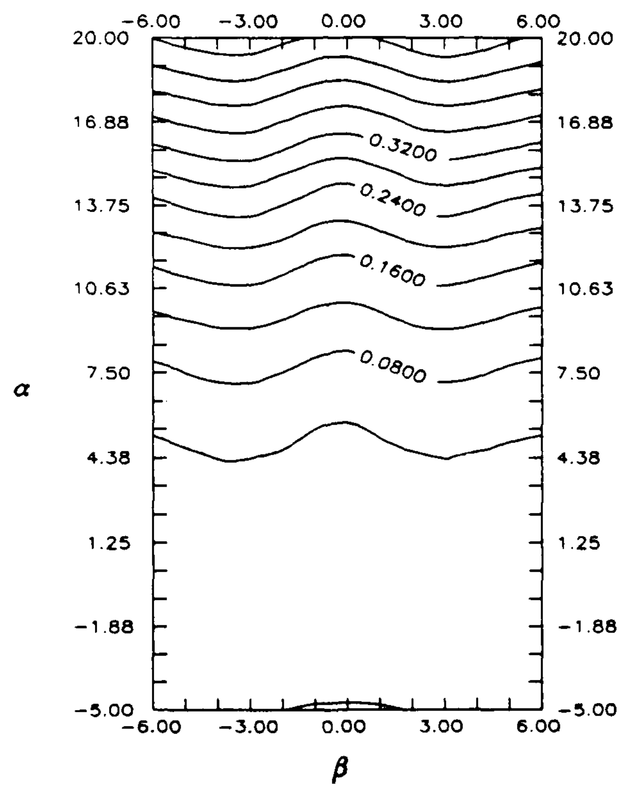


Figure 5 Contours of C_D

α axis. Note that while the drag coefficient exhibits a strong dependence on α, with the characteristic quadratic term, it also shows a significant dependence on β. All of the longitudinal coefficients exhibited a similar dependence on β and both the polynomial fit and the plotting routine (SURFER) generated the same characteristic shape. In addition to this, the correlation coefficients developed for all the longitudinal data indicated a good capture of the behavior of the data and therefor this coupling is assumed to exist.

The lateral derivatives, see Figure 6, exhibited the expected strong dependence on β., of all the lateral derivatives the rolling moment coefficient exhibited the strongest α / β coupling; which may be observed in Figure 7. Note that the contours break rather sharply at a given AOA and for the aircraft in an unsymmetric orientation.

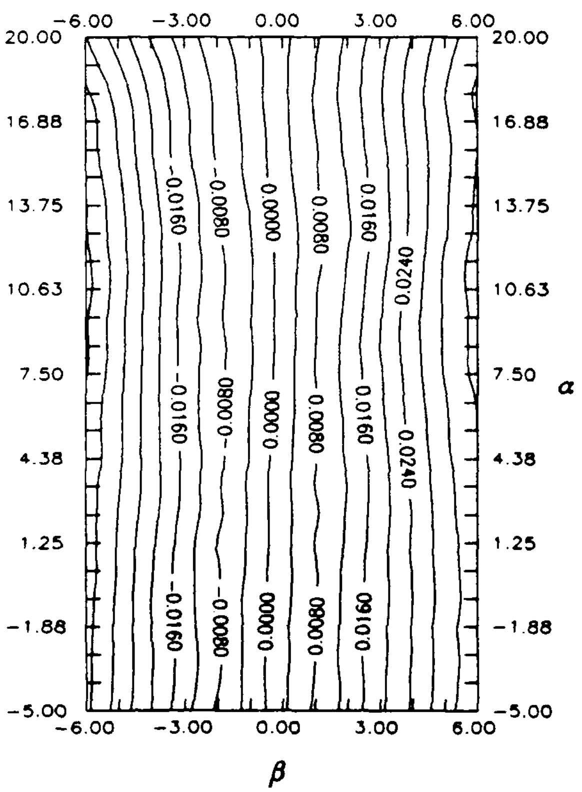


Figure 6 Contours of Constant C_N

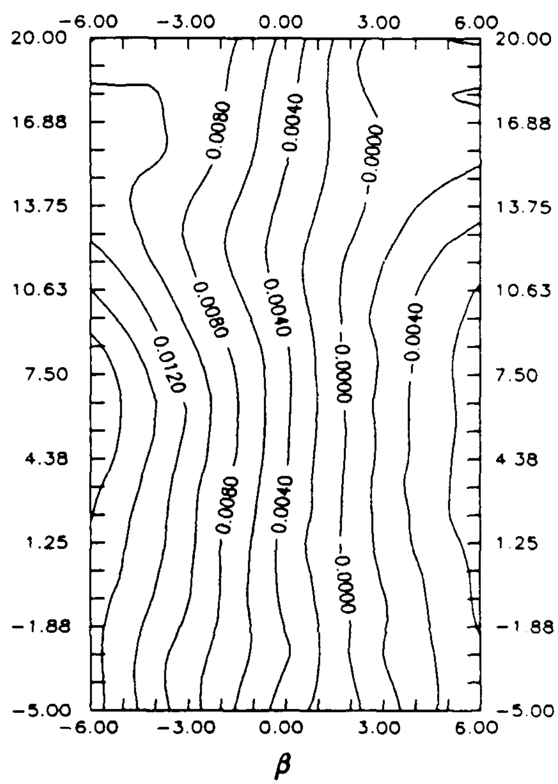


Figure 7 Contours of Constant C_l

Aircraft Control Derivatives

Given that the control derivatives are not constant values, it is appropriate to address questions of how they may vary as a function of AOA and sideslip angle. Also, since this thesis involves the investigation of situations where a control surface has failed it is also important to gain some appreciation of how important each surface is relative to the others in effecting a given force or moment. To accomplish these purposes the control derivatives for the seven surfaces were calculated at different locations in α / β space. As will be seen in Chapter IV control derivatives effecting a given force can be arranged as a row vector. For this reason, the control derivatives were normalized in a vectorial sense by creating a vector in 7 space whose magnitude is one. The normalization was accomplished as follows. First, each control derivative was multiplied by the maximum deflection available for that surface.

$$C_{f_{i_{max}}} = C_{f_i} * \delta_{imax} \tag{3.1}$$

All of these values were then squared and summed.

$$C_f^2 = \sum_{i=1}^{7} C_{f_{imax}}^2 \tag{3.2}$$

The normalized derivative is then defined to be:

$$C_{f_{inorm}} = \frac{C_{f_{imax}}}{C_f} \tag{3.3}$$

and will be a number whose magnitude is between zero and one. By observing the relative size of each component, information can be obtained about the relative importance of each control.

To observe the variation of the normalized derivatives as a function of α and β, contour plots were constructed showing lines of constant values of the normalized derivatives. Several points are worth remembering in observing these charts, which may be found in Appendix E. First, the plots do not provide information about the actual value of the control derivative and how it is changing with α and β. They indicate how the relationship of that surface is changing relative to the others at different points. Second, the numbered contours do not represent percentages since it is the sum of the squares of all the derivatives which are equal to unity. Third, when noting changes that are occurring to the contour lines it is important to remember that all seven surfaces must be observed to have an accurate understanding of the changes indicated.

As would be expected, the rudder exerts essentially zero influence on either pitching moment or the normal force coefficients. The horizontal tails, Figure 8, show that they are the most significant player with respect to pitching moment; with the primary variation in the normalized derivative occurring as a function of β. Figure 9 indicates that while the flaperons are not as significant an effector of pitching moment as the tails they do contribute to the overall pitching moment. A slight dependence on α is indicated for the flaperons with in the range examined. The LEFs are relatively small effectors. The normal force is most strongly influenced by the flaperons and the horizontal tails; see Figures 10 and 11.

It is in the lateral derivatives that the most dramatic results are observed. The plots for yawing moment indicate that the rudder, Figure 12, is far and away the most significant surface in effecting this moment. Figures 13 and 14 illustrate that some yawing capability is exchanged between the flaperons and the horizontal tails as the angle of attack is changed. The rudder is also observed to be the most dominant control surface for introducing side force into the aircraft; see Figure 17. Figure 18 indicates that the horizontal tails also are capable of generating side force. This capability can be accounted for

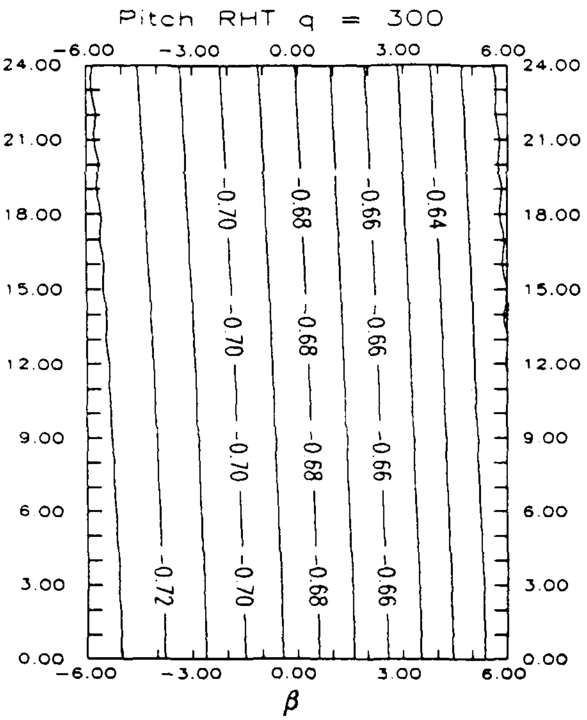


Figure 8 Pitch RHT q = 300

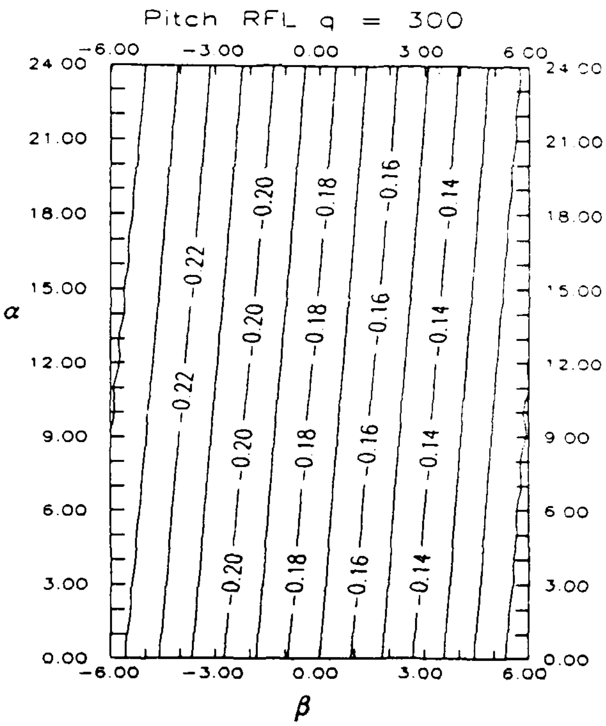


Figure 9 Pitch RFL q = 300

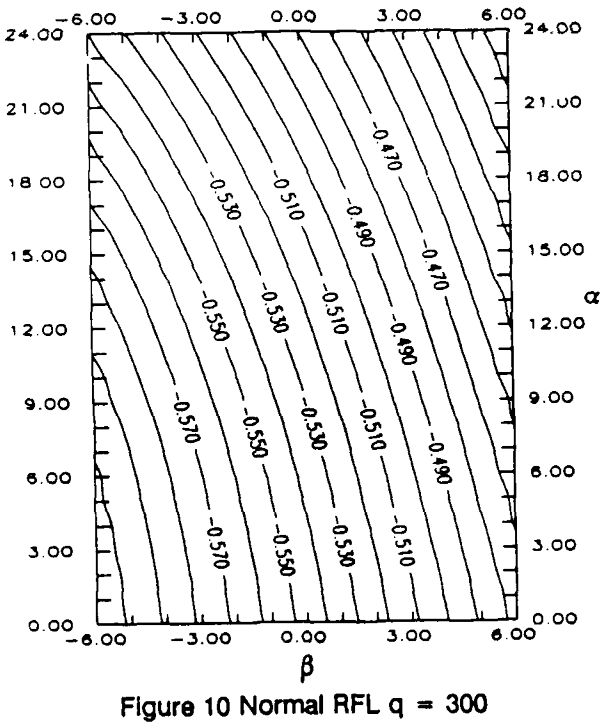


Figure 10 Normal RFL q = 300

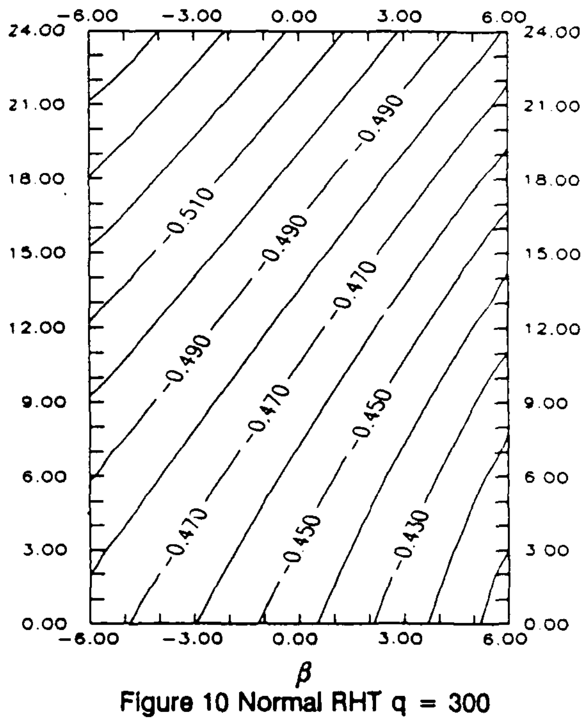


Figure 10 Normal RHT q = 300

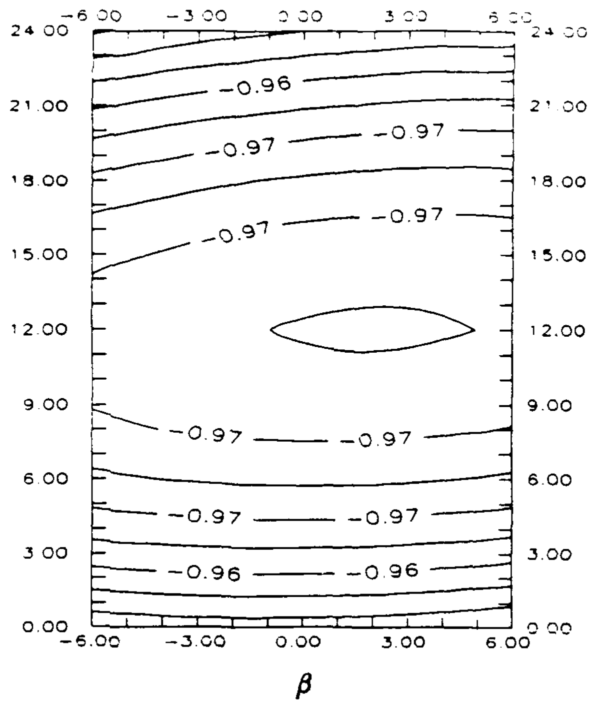


Figure 12 Yaw Rudder q = 300

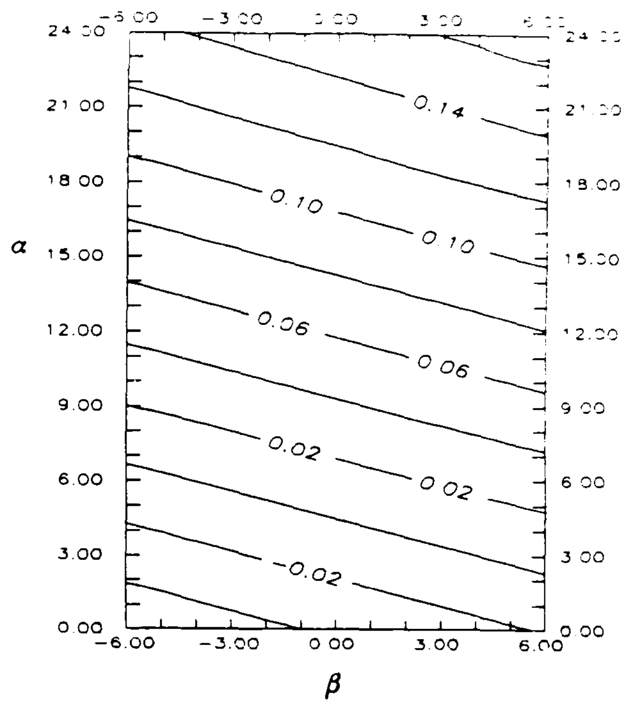


Figure 13 Yaw RFL q = 300

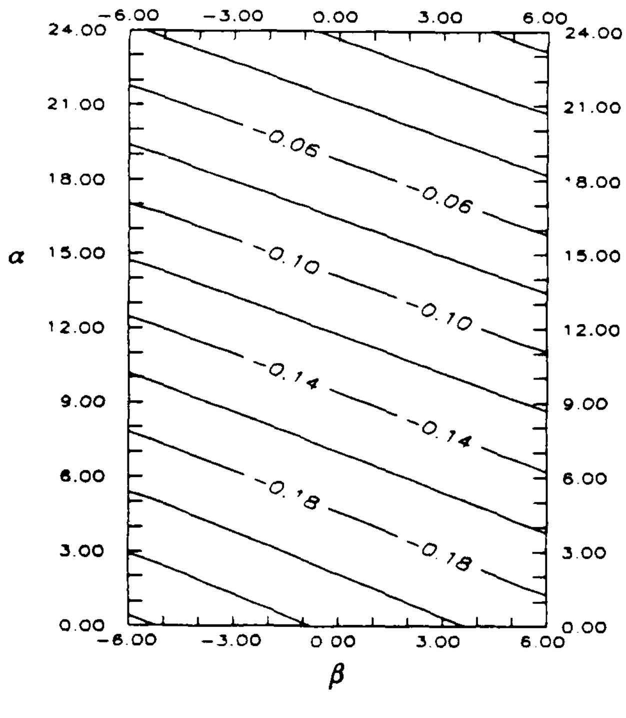


Figure 14 Yaw RHT q = 300

by noting the anhedral in the horizontal tail which produces a component of force in the Y direction when the horizontal tails are deflected unsymmetrically. Another point of interest in the side force plots is that the LEFs, whose influence is negligible at low AOA, become more significant as the AOA is increased; see Figures 15 and 16. At the higher AOA the LEFs make a small, but notable, contribution to the side force relative to the other surfaces. These plots of side force control derivatives establish a very significant point for the analysis performed in this thesis; even a relatively small deflection of the rudder can not be "overpowered" by a maximum asymmetric deflection of the remaining surfaces.

Examining the plots of the rolling moment control derivative, Figure 19, will show that not only are the rudder contours almost entirely dependent on α but also that the rolling moment produced by deflection of the rudder changes sign at 12.9 degrees AOA. This results because the moments are recorded in the stability axis system and there will be an AOA at which the X Stability axis will pass through the effective point of application of the side force developed by the rudder. The zero moment arm results in zero moment about this axis. Again the flaperons and horizontal tails are observed to be exchanging relative importance as effectors of rolling moment. Note that the islands for the flaperon and horizontal tail plots appear below and above the zero line on the rudder plot respectively, see Figures 20 and 21.

Not only did the contour plots of the normalized derivatives provide useful information about the relative importance of the control derivatives but they also indicated that an error had been made in developing the control derivatives for the left flaperon and the left horizontal tail. In Chapter II it was noted that the wind tunnel tests did not provide data for the left flaperon and left horizontal tail and that it was assumed that the data from the right surfaces could simply be reflected across the X-Z plane. This was accomplished by negating the sign on the lateral derivatives and assigning the same longitudinal derivatives. Note that the LEFs Figures 15 and 16 not only exhibit opposite sign but also an opposite slope as a function of β. The change in slope results from the fact that the right and left surfaces react differently to positive and negative β. For example, the right leading edge flap becomes more effective, relative to the left leading edge flap, with positive β since the right LEF is now seeing

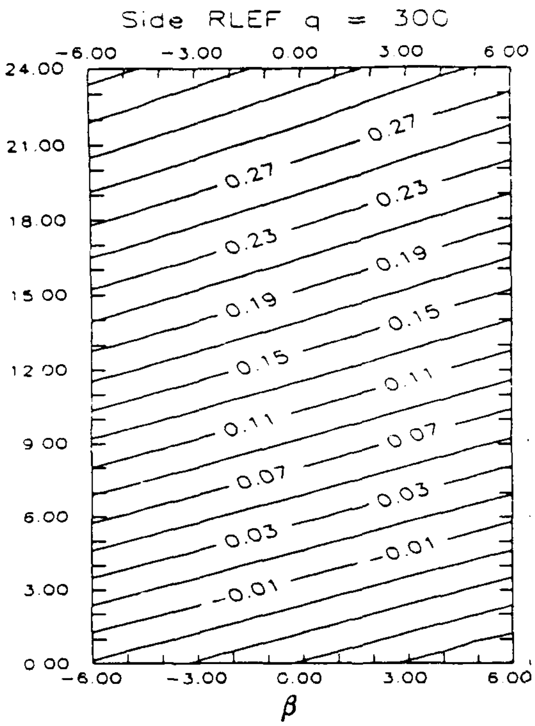


Figure 15 Side RLEF

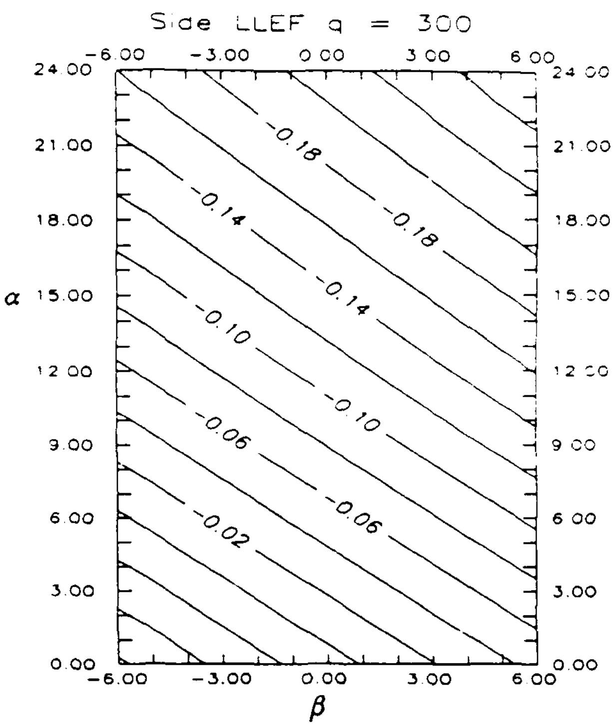


Figure 16 Side LLEF

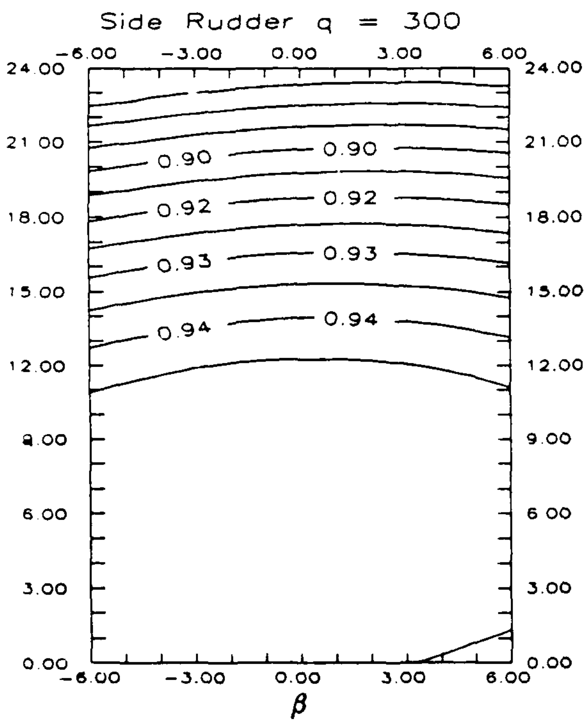


Figure 17 Side Rudder

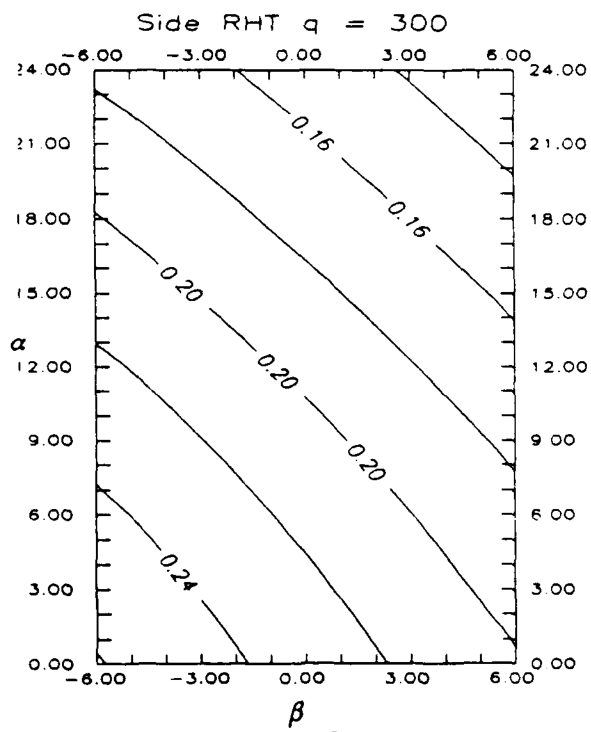


Figure 18 Side RHT

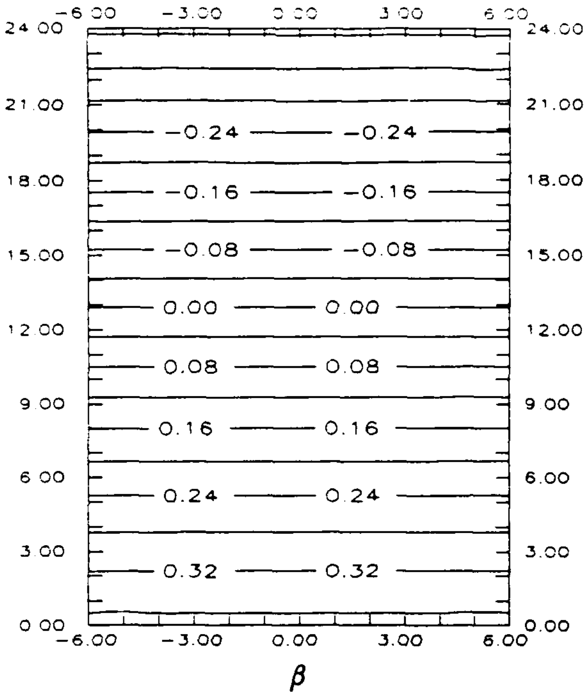


Figure 19 Roll Rudder q = 300

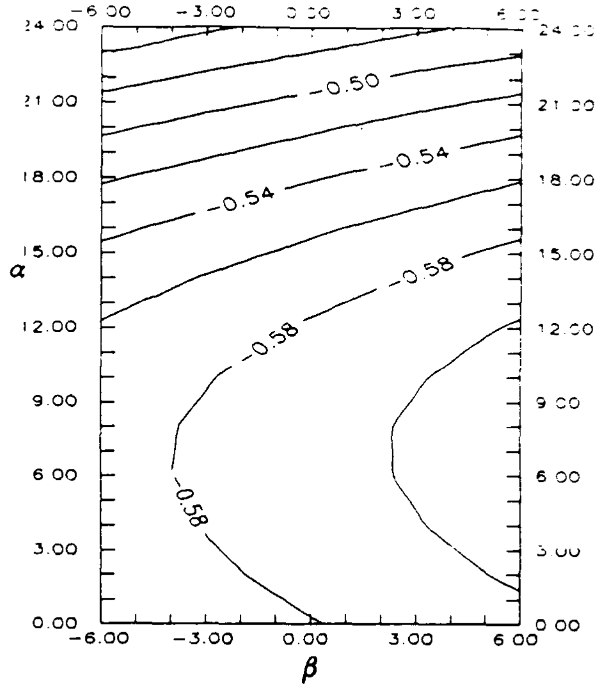


Figure 20 Roll RFL q = 300

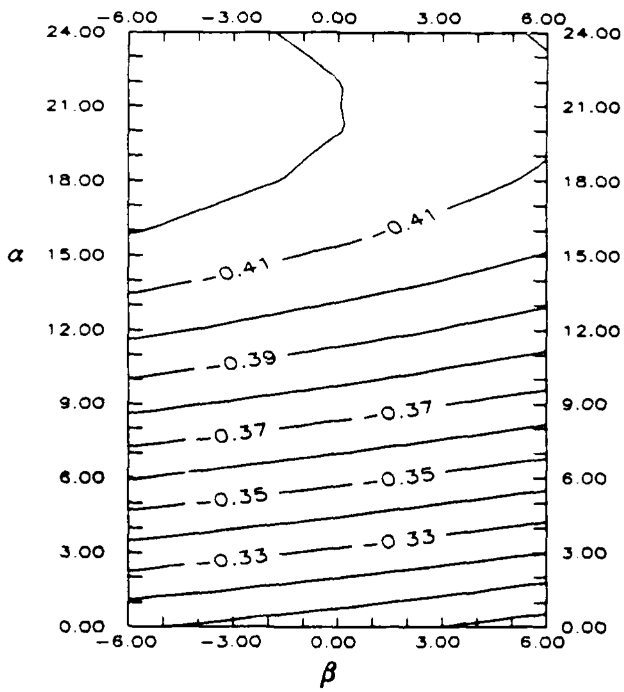


Figure 21 Roll RHT q = 300

"more of the wind". The initial method for creating the left control derivatives had missed this fact and while the right and left lateral derivatives appropriately had opposite signs they inappropriately exhibited the same slope. The plots contained in Appendix E and the control derivatives listed in Appendix C have been corrected to be consistent with the behavior described above.

Summary

In this chapter the contour plots of the aircraft stability derivatives were examined to identify coupling . All of the longitudinal coefficients exhibited a similar variation as a function of β. Among other things, this will be shown to produce a trimmed condition at a lower AOA when the aircraft is in a slightly unsymmetric orientation. The lateral coefficients showed a significant coupling occurring at the higher angles of attack, indicating that the requirements for opposing moments for the aircraft in an unsymmetric orientation will change significantly as AOA is increased. Perhaps the most prominent result is that the failure of the rudder is demonstrated to be the most significant failure of a single control surface. Failures of the other surfaces can be compensated for by the remaining functional surfaces, but the rudder so dominates the vectors for side force and yawing moment that a failure of this surface will almost certainly indicate either unsymmetric flight or departure of the aircraft.

IV PROBLEM FORMULATION

Introduction

In chapter I, three questions were posed regarding the stability characteristics of an aircraft with failed control surfaces. Specifically: given a failure, can a trim solution be achieved? If trim is achievable, how large is the region in α / β space and what are the stability characteristics of the aircraft within this space? And, finally, can the space be augmented or improved by allowing for greater independence of the control surfaces? In this chapter the equations of motion derived in Appendix D are used in conjunction with the aerodynamic predictor equations developed in Chapter II to provide techniques for addressing these questions. The three control schemes and the flight conditions studied in this thesis are defined. A discussion of the use of the trim equations and the order of their solution is provided along with an overview of the FORTRAN codes developed to solve the trim problem. The matrix decomposition techniques of Singular Value Decomposition (SVD) and the Row Reduced Echelon Form are advanced as a mean of gaining further insight into the nature of the stability characteristics of the aircraft.

Problem Scope

The trim condition which is desired is that equilibrium state which results in the aircraft flying in constant altitude, rectilinear flight. While other flight conditions, which might be less difficult to achieve in the event of a failure of a control surface; only constant altitude flight is examined in this thesis. A failure of the rudder, which results in the rudder being locked at some deflection is the failure mode which will be studied in depth. This failure is selected since it appears to be one of the more challenging conditions to be addressed. The discussions which follow will be primarily concerned with rudder failure, but the techniques developed and some of the results are pertinent to failures of other surfaces as well.

The investigation will also be limited by the range of the test data which was collected by Turhal [12]. Therefore, the dimensions of the α/β space which will be examined are limited to; $-6.0 \leq \beta \leq 6.0$ and $0 \leq \alpha \leq 20$. A final set of assumptions which are pertinent to the formulation of this investigation are the assumptions associated with the derivation of the equations of motion; they are as follows

1. The aircraft is assumed to be a rigid airframe.
2. The earth is assumed to be an inertial frame of reference.
3. The Aircraft mass and mass distribution are assumed to be constant.
4. The X-Z plane of the aircraft is assumed to be a plane of inertial symmetry.

The implications of these assumptions are discussed in detail in Appendix D, where the equations of motion are derived.

Control Schemes

As was noted in Chapter II, the current implementation of the control surfaces on the F-16 allows the pilot to command both differential (HA) and symmetric (HE) deflections of the horizontal tails and strictly asymmetric deflection of the flaperons (FA). While the flaperons may be deployed symmetrically, as flaps, this is not part of the normal control of the aircraft. In the same manner, the leading edge flaps (LEF) are deployed via scheduling and are not under the direct control of the pilot.

The control schemes used in this thesis were derived by allowing the control surfaces currently on the aircraft to deploy with successively greater independence. It should be noted that the control schemes discussed in this thesis do not refer to control laws.

In Case A the control schemes investigated are essentially the current control scheme, described above, with the improvement that the LEFs are now controlled directly. Consistent with their current deployment, they are limited to symmetric deflection. The rudder is not listed in Table 2 since

Table 2 Control Schemes

Case A	Case B	Case C
δ_{LEF}	δ_{LFL}	δ_{LFL}
δ_{FA}	δ_{RFL}	δ_{RFL}
δ_{HA}	δ_{LHT}	δ_{LHT}
δ_{HE}	δ_{RHT}	δ_{RHT}
		δ_{LLE}
		δ_{RLE}

in all the studies performed in this research the rudder is the failed surface and is therefore not available for control. The deflection of the individual control surfaces in Case A are related as follows:

$$\delta_{lef} = \frac{1}{2}(\delta_{RLE} + \delta_{LLE}) \tag{4.1}$$

$$\delta_{FA} = \frac{1}{2}(\delta_{RFL} - \delta_{LFL}) \tag{4.2}$$

$$\delta_{HA} = \frac{1}{2}(\delta_{RHT} - \delta_{LHT}) \tag{4.3}$$

$$\delta_{HE} = \frac{1}{2}(\delta_{RHT} + \delta_{LHT}) \tag{4.4}$$

For Case B, the LEFs return to being scheduled surfaces but now the flaperons are permitted, like the horizontal tails, to deflect independently of one another. This should provide the aircraft with greater control over the lift experienced at a given AOA and some additional pitch control. Case C represents the situation where all the available surfaces are allowed to deploy independently. While the feasibility of implementing such a control scheme might be argued the object here is to study what advantages might be gained if such a scheme were achievable. It might also be noted that each scheme is related to the others. In fact, Case A and Case B are special cases of Case C. The original control scheme then is simply a more constrained version of Case A.

Problem Set-up

In Appendix D the equilibrium equations for rectilinear flight were derived along with an expression for the aircraft pitch angle that specified constant altitude flight. Repeating these for clarity

$$F_{A_X} + F_{T_X} - mg \sin \theta = 0 \tag{4.5}$$

$$F_{A_Y} + mg \cos \theta \sin \phi = 0 \tag{4.6}$$

$$F_{A_Z} + mg \cos\theta \cos\phi = 0 \tag{4.7}$$

$$M_{A_X} = 0 \tag{4.8}$$

$$M_{A_Y} = 0 \tag{4.9}$$

$$M_{A_Z} = 0 \tag{4.10}$$

$$\theta = \tan^{-1} \left\{ \tan \alpha \cos \phi + \frac{\tan \beta}{\cos \alpha} \sin \phi \right\} \tag{4.11}$$

The aerodynamic force and moment coefficients, as functions of α, β, and the control surface deflections, were defined in Chapter II to be expressions of the form

$$C_f = \sum_{j=0}^{J} \sum_{i=0}^{I} A_{ij} \alpha^i \beta^j + \sum_{\ell=1}^{7} \sum_{m=0}^{M} \sum_{n=0}^{N} B_{\ell mn} \alpha^n \beta^m \delta_\ell \tag{4.12}$$

These nondimensional coefficients may be converted into forces and moments by means of the relationships defined in Chapter II. Since the wind tunnel data was recorded in the Stability Axis system, a transformation will have to be performed to express the forces in the Body Axis system, which are the forces specified in equations (4.5) - (4.7). The Body and Stability Axis Systems are defined in Appendix D and are shown in Figure 22

Studying equations (4.5) - (4.12) reveals that the equations are nonlinear due to the powers on α and β and the trigonometric functions in equations (4.5) - (4.7). Not only are the equations nonlinear, but they are also coupled in several ways. Equation (4.6) (side force) and equation (4.7) (normal force) both include terms which have θ and ϕ in them. This effectively couples the lateral and longitudinal equations of the aircrafts motion. Second, the aircraft control derivatives and stability

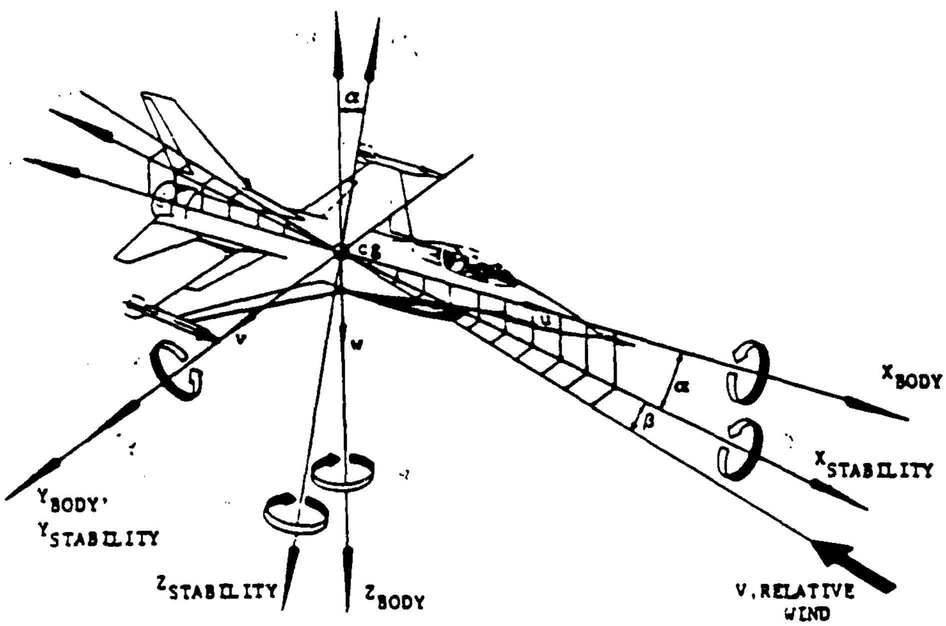


Figure 22 F-16 Body and Stability Axis Systems

derivatives have coupled terms of α and β in them. Third, equation (4.11) introduces a strong coupling of the lateral and longitudinal equations. Obviously, as currently expressed the problem is not conducive to linear solution techniques and must be manipulated to produce a solvable problem.

If the equations were fully expanded by substituting in the aircraft forces and moments defined in Chapter II, the following unknowns would be identified: dynamic pressure, gross weight, 7 control surface deflections, AOA, sideslip angle, thrust, pitch angle, and roll angle. As stated, that amounts to fourteen unknowns and six equations. Several unknowns can be removed by stating the aircraft configuration and the flight conditions at which the analysis is to be performed. Two flight conditions are defined in Table 3 for use in the analysis. Condition I is representative of the aircraft at an approach speed and Condition II permits the analysis of a cruise condition. Note that the thrust term only appears in the axial force equation. For this reason, the assumption is made that at any condition where equilibrium can be achieved, within other limits, the aircraft engine can develop sufficient thrust to satisfy equation (4.5). Equation (4.5) is not included in the analysis from this point forward. Since one surface is assumed to be failed this will remove another unknown as will the constraint of constant altitude flight which defines θ in terms of α, β, and ϕ (4.11). At this point the problem has been reduced to five equations in nine unknowns. The nonlinearities and coupling noted earlier still remain to be addressed. Since one of the stated objectives of this investigation is to define the region in α/β space in

Table 3 Flight Conditions

	I	II
Gross Weight	19000 lbf	1900lbf
Mach	0.22	0.6
Altitude	Sea level	15000 ft
Velocity	150 KEAS	297 KEAS
$\bar{q}$	75 psf	300 psf

which trim is achievable it is reasonable to specify a value for α and β. The problem then reduces to seven unknowns. Table 2 indicates, however, that Cases A and B involve only four independent controls. The number of unknowns is now five and equal to the number of equations. Further, by specifying α and β we have reduced all of the aerodynamic forces and moments to linear functions. For Cases A and B, and with the specification of α and β, the forces and moments may be written in the form

$$F_i = A_0 + B + \sum_{\ell=1}^{4} \sum_{m=0}^{1} \sum_{n=0}^{1} C_{\ell mn} \alpha^n \beta^m \delta_\ell \tag{4.13}$$

Here Ao represents the force or moment of the "zero" case, B the contributions of the failed control surface and in Case B the LEFs, and the last term the force or moment that will result from the unknown deflections of the control surfaces.

Solving the Trim Problem

Figure 23 is a schematic flow chart of the FORTRAN codes developed to solve the defined trim problem and provides a useful aid for following the solution technique employed. The previous discussion follows the flow chart down to the point where the forces and moments due to the failed control surface, the rudder, have been calculated. Given that for zero flight path angle θ is equal to α an initial estimate for θ is given as α. Further, since the remaining control surfaces do not exert a strong influence on the aircraft side force it is initially assumed that the unknown control surfaces do not appear in equation (4.6). With these assumptions equation (4.6) may be solved for an initial estimate of ϕ. At this point all of the angles in the problem have either been specified or estimated and hence the only remaining unknowns in the problem are the control surface deflections. Based on the restrictions placed on the control derivatives in Chapter II the problem is now a linear problem of four equations and four unknowns which may be formed as follows

$$-(A_z + B_z + mg\ \mathrm{Cos}\theta\ \mathrm{Cos}\phi) = \sum_{i=1}^{4} C_{zi}\ \delta_i \tag{4.14}$$

$$-(A_m + B_m) = \sum_{i=1}^{4} C_{mi}\ \delta_i \tag{4.15}$$

$$-(A_\ell + B_\ell) = \sum_{i=1}^{4} C_{\ell i}\ \delta_i \tag{4.16}$$

$$-(A_n + B_n) = \sum_{i=1}^{4} C_{ni}\ \delta_i \tag{4.17}$$

Since everything on the left hand side of each equation is known the problem may be rewritten in the familiar form:

$$b = [A]\ \delta \tag{4.18}$$

The b vector contains all the known forces and moments and has as its rows; normal force, pitching moment, rolling moment, and yawing moment. The δ vector is the unknown control deflections and the 4 x 4 A matrix contains the control derivatives of the respective controls. Solving equation (4.18) will define the control deflections needed to achieve trim.

Earlier in the problem solution an assumption was made that the side force did not contain terms from the unknown control surfaces. Further, the pitch angle was estimated as α though in Appendix D it is demonstrated that this is not true in general. These assumptions are now accounted for by recalculating the sideforce including the force due to the deflections found via equation (4.18) and calculating a new pitch angle with equation (4.11). A new roll angle is then calculated with these updates and the problem is iterated until the errors between the estimates for θ and ϕ become small.

While the deflections determined by solving equation (4.18) will result in the satisfaction of the equilibrium equations these deflections may not represent a solution to the aircraft trim problem. To be a bonafide solution the deflections determined by equation (4.18) may not exceed the deflection limits defined in Table 4. If the calculated deflections are within these constraints then that point has been determined to be a point in α/β space at which trim can be effected.

Computer Codes

A FORTRAN computer code was written for each of the three control schemes defined. The order of solution and logic are essentially the same for each code with one important distinction. The discussion provided above only covered the cases where there are four independent control deflections to be solved for. Case C incorporates six control surfaces and therefore may not be solved directly by the technique described above. Case C was solved by placing an additional two loops outside of the α/β loops of the problem flow charted in Figure 23 ; one loop for each of the leading edge flaps. The

Table 4 Control Surface Deflection Limits

LEF	$-2° \leq \delta \leq 25°$
FLAPERONS	$-20° \leq \delta \leq 20°$
HRZT Tails	$-25° \leq \delta \leq 25°$
Rudder	$-30° \leq \delta \leq 30°$

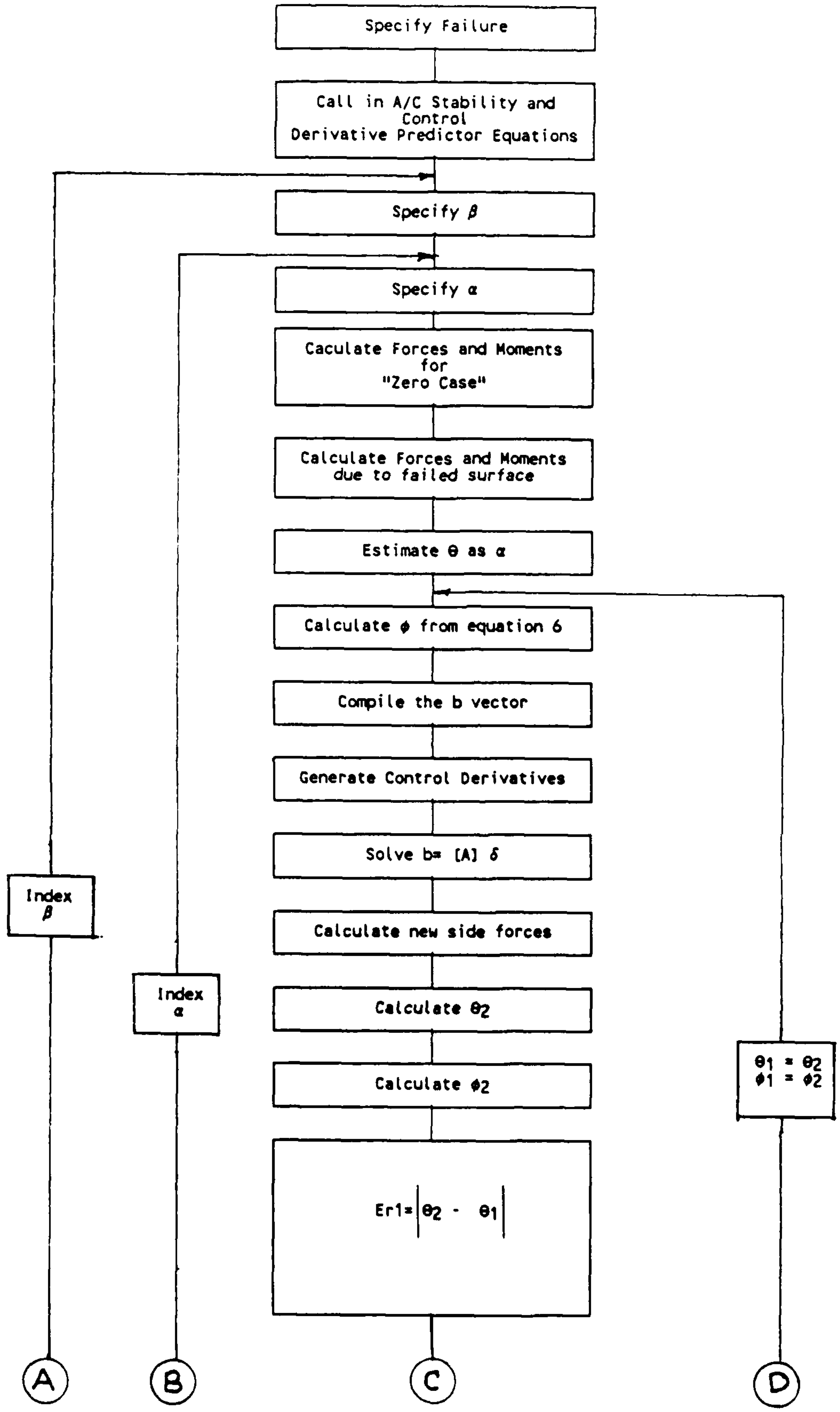
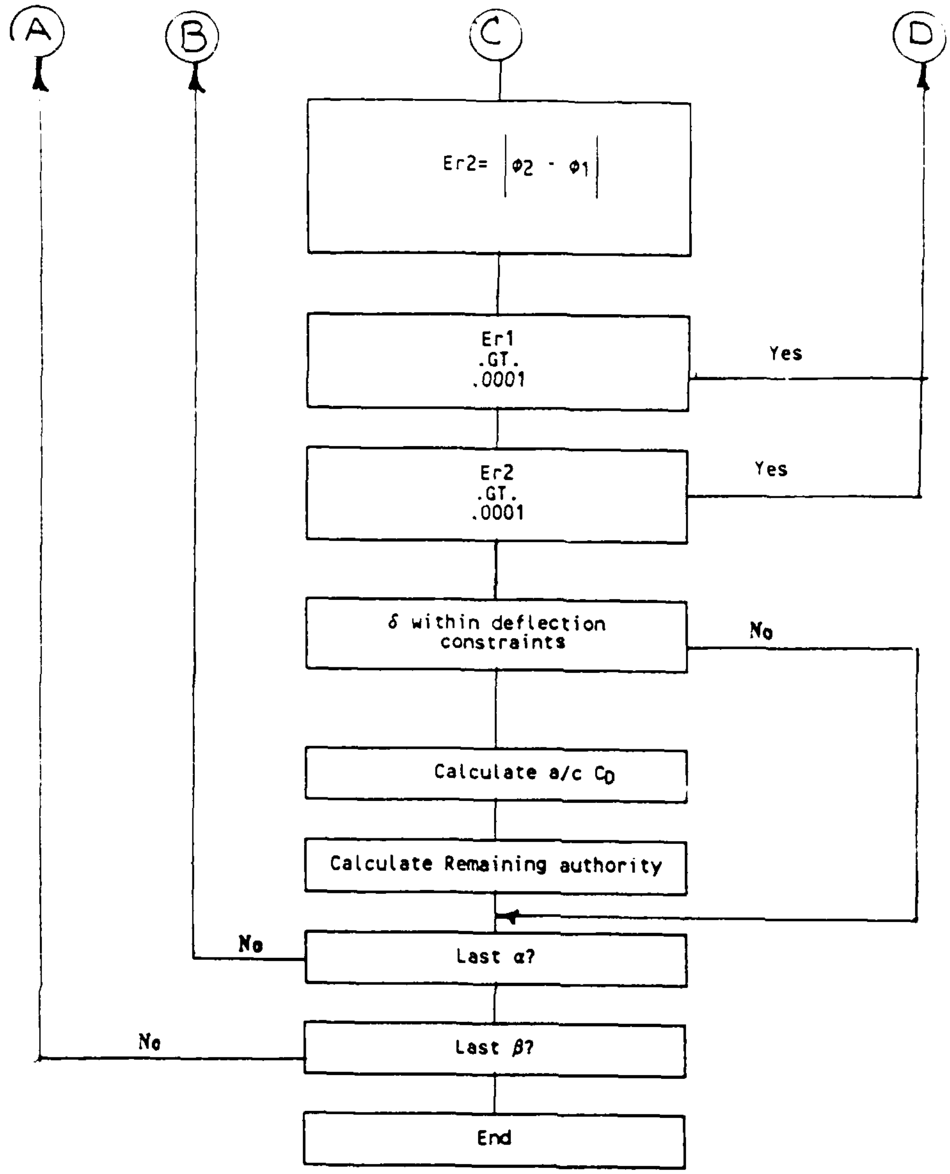


Figure 23 Autrim Flowchart



Autrim Flowchart Continued

LEFs were then allowed to vary through their ranges in one degree increments. As will be seen later in this chapter, the introduction of the two additional degrees of freedom to the problem means that there may be multiple solutions at a given point. The coding logic is such that only one solution is recorded for a given point in α/β space. The three computer codes are included in Appendix F.

Matrix Decomposition Techniques

Two techniques for decomposing the linear problem which has been defined were investigated as means for gaining additional insight into the nature of the problem. These techniques are particularly helpful for Case C where a unique solution to the problem does not exist. The problem is stated in the following form

$$b = [A] \; \delta \tag{4.19}$$

Here b is a 4 x 1 vector, A is 4 x 6 matrix of control derivatives, and δ is a 6 x 1 vector of unknown control deflections. By augmenting the A matrix with the b vector and placing the augmented matrix in Row Reduced Echelon Form (RREF)[5:40-41], the problem can be decomposed into the form

$$\left[\begin{array}{cccccc|c} 1 & 0 & 0 & 0 & A & B & b_1' \\ 0 & 1 & 0 & 0 & C & D & b_2' \\ 0 & 0 & 1 & 0 & E & F & b_3' \\ 0 & 0 & 0 & 1 & G & H & b_4' \end{array}\right] \tag{4.20}$$

Which may be rewritten as:

$$
\begin{aligned}
b_1' &= \delta_1 + A\delta_5 + B\delta_6 \\
b_2' &= \delta_2 + C\delta_5 + D\delta_6 \\
b_3' &= \delta_3 + E\delta_5 + F\delta_6 \\
b_4' &= \delta_4 + G\delta_5 + H\delta_6
\end{aligned}
\tag{4.21}
$$

In turn equation (4.21) is manipulated to place the problem in the desired form.

$$
\{\delta\} = \{b'\} - \delta_5 \begin{Bmatrix} A \\ C \\ E \\ G \end{Bmatrix} - \delta_6 \begin{Bmatrix} B \\ D \\ F \\ H \end{Bmatrix} \tag{4.22}
$$

Stated in this way several things may be observed. First, when (δ 5) and (δ 6) are zero the b' vector represents the solution to the four independent control problem. Secondly, equation (4.22) defines the range of available solutions that may be obtained at the specified point in α / β space. Any solution in the span defined by equation (4.22) is a viable solution provided that the control deflections are within the defined limits. Also note that the failure of any control surface may be represented simply by changing the control surfaces whose control derivatives are contained in the A matrix. Or, viewed from another angle, it can be seen that equation (4.22) defines the degree to which any two additional surfaces may be failed and equilibrium still be achieved.

Given a matrix A it may be decomposed via Singular Value Decomposition into the following form, [9:451].

$$A = [U_1 \quad U_2] \begin{bmatrix} \Sigma & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix} \tag{4.23}$$

Here the columns of U are defined to be the left singular vectors, the columns of V are the right singular vectors and (Σ) is a diagonal matrix with the nonzero singular values of A on the diagonal. If A is an m x n matrix and their are r nonzero singular values then the following dimensions will be established. U_1 will contain r columns and U_2 will contain m-r columns. V_1 will have r columns and V_2 n-r columns, [9,452]. The range space is defined by the span of the columns of U_1 and the null space by the span of the columns of V_2. SVD provides two insights into the problem that are immediately apparent. If the matrix A is found to have any singular values that are zero then A is rank deficient by the number of zero singular values and a unique solution to the linear problem, as formulated, does not exist. The columns of V_2 span the null space of the problem with the attending implication that any combination of control deflections that are in that span will map to zero. Stated another way, if the controls are combined in such a manner that the vector of control deflections (δ) is equal to one of the vectors in V_2 times a constant, then that combination of controls will have no effect on the forces and moments represented in the b vector of equation (4.19).

Summary

In this chapter the nonlinear equilibrium equations derived in Appendix D are used to develop a methodology for determining if and where trim may be achieved for a given control surface failure. The solution technique and order are discussed using a schematic flow chart, which describes the FORTRAN codes which were written to perform the trim investigations. The matrix decomposition techniques of Singular Value Decomposition and the Row Reduced Echelon Form are presented as methods for gaining a deeper understanding and appreciation of the defined problem.

V INVESTIGATION RESULTS

Introduction

In this chapter the results of the equilibrium analysis performed via the methodology of Chapter IV are presented. The relative merits of each of the three control schemes will be discussed with respect to not only their ability to augment the region in which trim is achievable but also their ability to affect the aircraft characteristics within the defined regions. Specific attention will be given to addressing why a particular control scheme gives the results that it does and what the ensuing implications are. A short discussion will be provided concerning preferred locations within the equilibrium space and what the attending pros and cons of being located at that point are. Contour plots of the aircraft roll angle, drag coefficient, and residual pitch and roll authority are used to support this analysis. The Row Reduced Echelon Form and Singular Value Decomposition are used to provide additional insight into the problem.

Trim Availability

In the event of a failure of a control surface one of the first questions to be addressed is whether the aircraft can be maintained in a state of equilibrium. An investigation of rudder failure was performed to address this question with the analysis subject to the constraints listed in Table 5. Only failures of the rudder resulting in a negative deflection, rudder deflected towards the starboard side of

Table 5 Problem Constraints

$$0^\circ \leq \alpha \leq 20^\circ$$
$$-6^\circ \leq \beta \leq 6^\circ$$
$$\delta_{MIN} \leq \delta_i \leq \delta_{MAX}$$

the aircraft, were investigated since the aircraft was otherwise assumed to be symmetrical. Table 6 presents the results of this initial analysis indicating that while a complete or "hardover" failure can be

Table 6 Maximum Trimable Rudder Failure

Flight Condition	I	II
Case A	-20°	-30°
Case B	-20°	-30°
Case C	-21°	-30°

tolerated at the second flight condition it is not possible to trim the aircraft at the lower dynamic pressure of Flight Condition I. Note that the increasingly complex control schemes do not significantly alter the degree of deflection that may be tolerated at Flight Condition I. While not essential, it seems desirable to be able to place the aircraft in a condition of symmetry or zero β. Table 7 indicates the de-

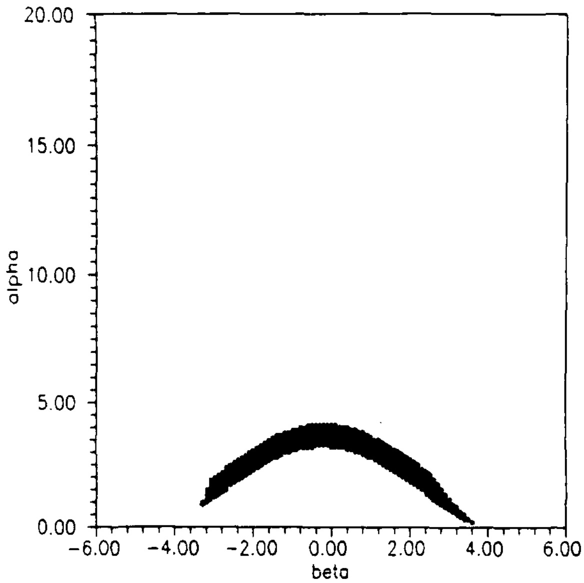
Table 7 Maximum Rudder Failure for $\beta = 0$

Flight Condition	I	II
Case A	-1°	-9°
Case B	-1°	-9°
Case C	-5°	-10°

gree of rudder deflection that can be sustained and the aircraft still returned to a zero sideslip condition. While Case C does provide a measure of improvement over the other control schemes it is hardly a substantial one. The results presented in these two tables indicate that with the control surfaces currently on the aircraft, even when employed with complete independence, equilibrium can not be achieved at all flight conditions if the rudder fails at its maximum deflection. This statement is made with the caveat of the constraints within which the analysis was performed. Even a partial failure of the rudder may necessitate flight in an unsymmetric orientation. While a symmetric orientation might be preferable, the fact that an equilibrium condition exists for a "hardover" failure should be noted as significant. The aircraft may not be able to be correctly oriented for a landing, but at least the occurrence of a rudder failure need not result in an uncontrollable departure of the aircraft.

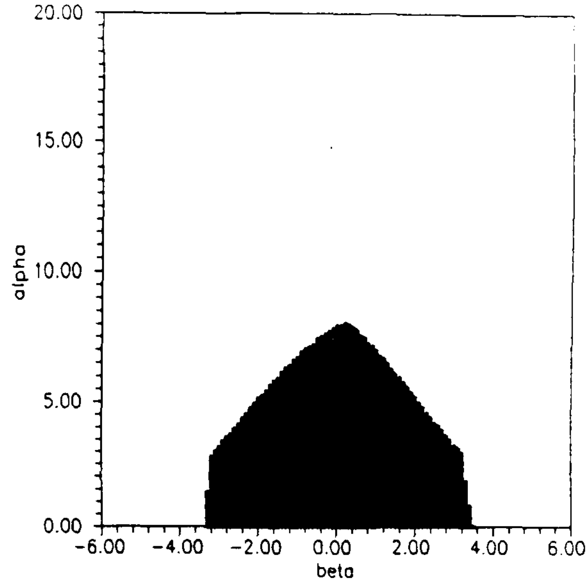
The information presented in Tables 6 and 7 indicate that the increasingly complex control schemes do not significantly change the aircrafts ability to sustain rudder damage. There are, however, advantages to be gained from permitting greater independence among the control surfaces. The shaded regions of Figure 24 show the positions in α/β space where the aircraft can be trimmed when the rudder is locked in a neutral position. Anywhere within this envelope, the correct application of controls will zero all of the accelerations and place the aircraft in an equilibrium state of constant altitude, rectilinear flight. It is immediately apparent that Case B provides the most significant improvement from one control scheme to the next at this flight condition. Also, the results discussed in the proceeding paragraph may be substantiated by observing that the Figures 25 and 26 which represent the equilibrium regions for rudder failures of ten and twenty five degrees respectively. At this flight condition, Flight Condition II, a significant improvement in the aircrafts ability to return to a zero β condition is not achieved by allowing more freedom among the control surfaces.

Note that as was discussed in Chapter IV each case is contained within the next, more complex, control scheme. Hence, the trim region of the control set-up of the current F-16 would be a line located with in the Case A trim region. Allowing the LEFs to be controlled, but in a strictly symmetric fashion, expands this line into the band which is shown in Figure 24. The substantial improvement from Case A to Case B results from allowing the flaperons to act as flaps in Case B. With this new symmetric deflection capability the aircraft now has the ability to significantly change its lift at a given point in the α/β space. One further note of interest is that the characteristic shape discussed in Chapter II for the longitudinal aerodynamic coefficients is evident in Figure 24. If desired, the aircraft can be trimmed at a lower AOA by assuming an unsymmetric orientation.



Equilibrium Area
for
0 Degs Rudder Failure

Case A

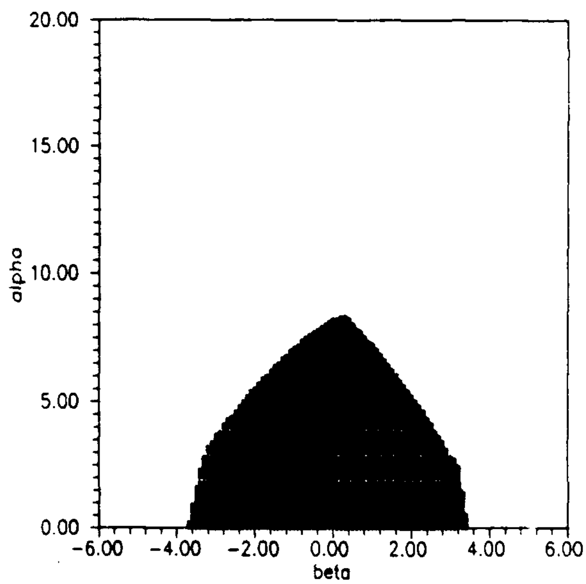


Equilibrium Area
for
0 Degs Rudder Failure

Case B

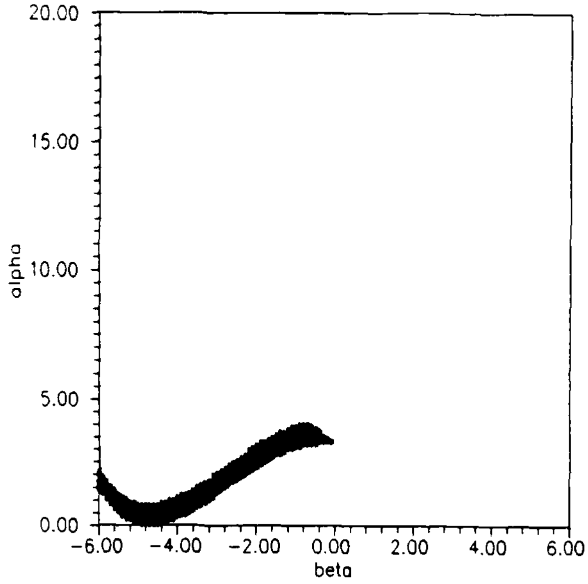
Figure 24 Equilibrium Regions for Flight Condition II

0 degrees rudder failure



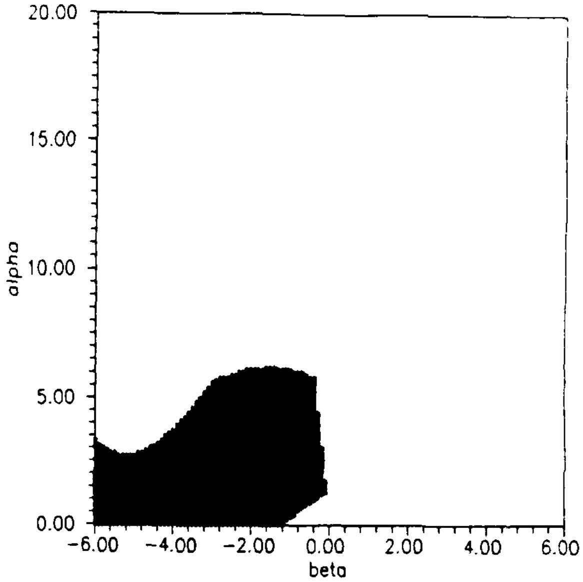
Equilibrium Area
for
0 Degs Rudder Failure

Case C



Equilibrium Area
for
−10 Degs Rudder Failure

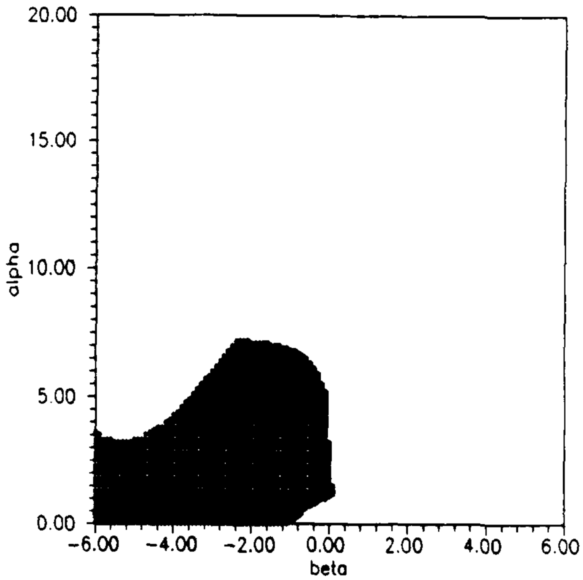
Case A



Equilibrium Area
for
−10 Degs Rudder Failure

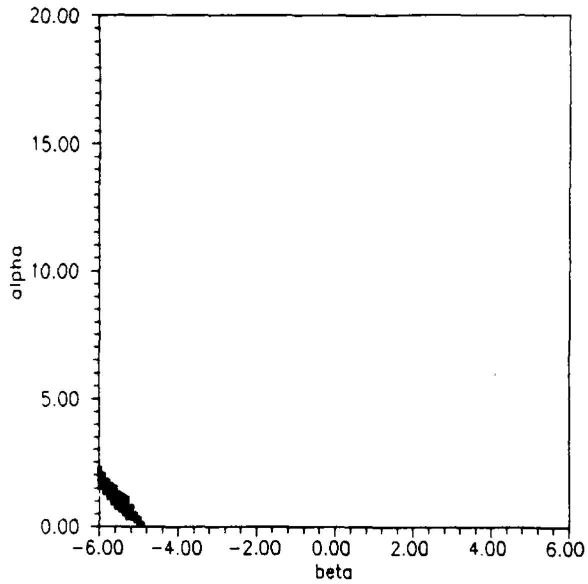
Case B

Figure 25 Equilibrium Regions for Flight Condition II
-10 degs. Rudder Failure



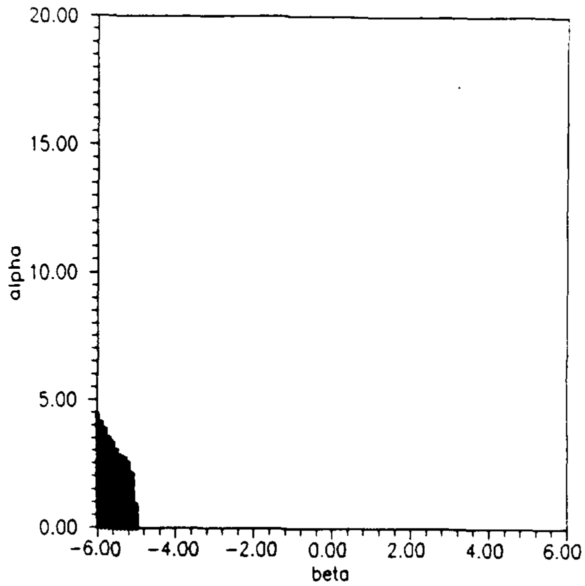
Equilibrium Area
for
−10 Degs Rudder Failure

Case C



Equilibrium Area
for
-25 Degs Rudder Failure

Case A

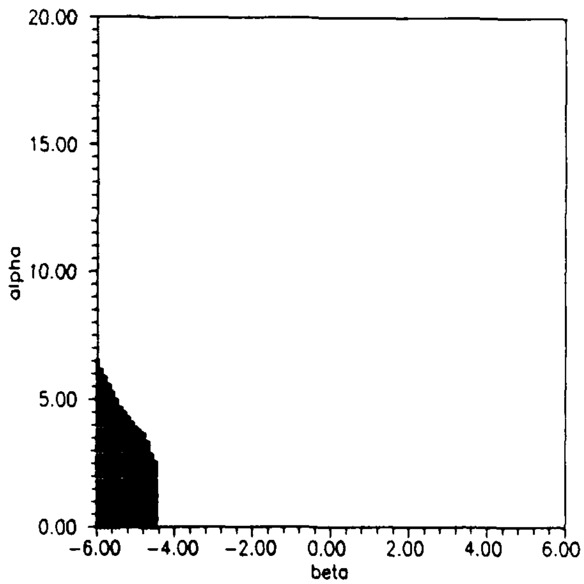


Equilibrium Area
for
-25 Degs Rudder Failure

Case B

Figure 26 Equilibrium Regions for Flight Condition II

-25 degrees rudder failure



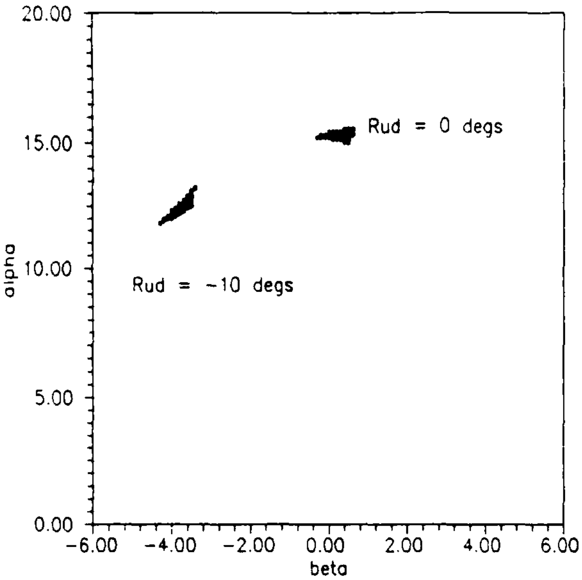
Equilibrium Area
for
-25 Degs Rudder Failure

Case C

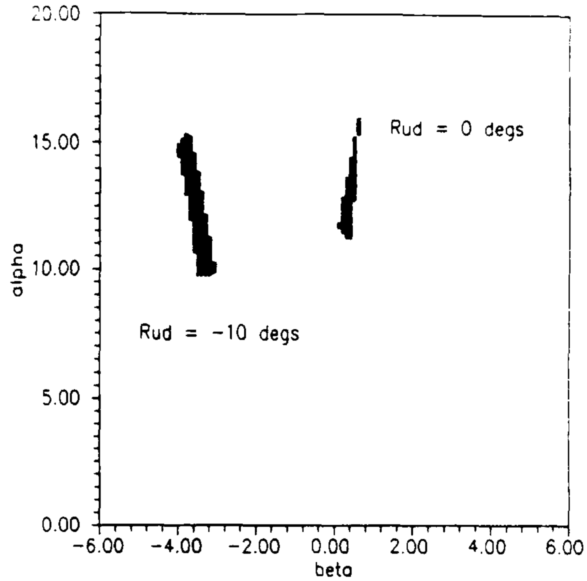
The equilibrium regions in which trim could be achieved for Flight Condition I provided perhaps the most dramatic evidence of the differences between the three control schemes. It can be observed from Figure 27 that the improvement gained from Case A to Case B is the ability to trim over a greater range of angle of attacks. Very little if any improvement is gained in the ability to move the aircraft laterally. This observation is substantiated by noting that the means by which the control surfaces generate lateral forces and moments is through asymmetric deflections. No additional capabilities for asymmetric control deflection exist between Case A and Case B. This is not the situation, however, for Case C. Case C augments Case B by allowing the LEFs to be deployed with complete independence. The advantage gained also is evidenced in Figure 27. Here the equilibrium region is visibly improved both in α and in β. The question naturally arises as to why Case C shows such a marked enlargement of the equilibrium region at Flight Condition I when its improvement is marginal at the higher dynamic pressure of Flight Condition II. The answer may be found by investigating the normalized derivative contour plots developed in Chapter III. Studying the contour plots, Appendix E, of the lateral derivatives for the LEFs will reveal that while they are almost insignificant relative to the other surfaces at the lower AOAs, they become quite prominent as angle of attack is increased. Therefore, at Flight Condition I where a fairly large α is required, a regime is entered where the LEFs have a significant role to play.

Tables 8, 9, and 10 provide a quantitative representation of the same information which is contained in the equilibrium region figures already observed. The computer codes, which performed the trim surveys, indexed through the α/β space searching for points at which trim could be achieved. Each trim point was located inside a square of area 0.01 deg ^{2}. The areas listed in Tables 8, 9 and 10 were obtained by summing all the "points" where a trim solution was found.

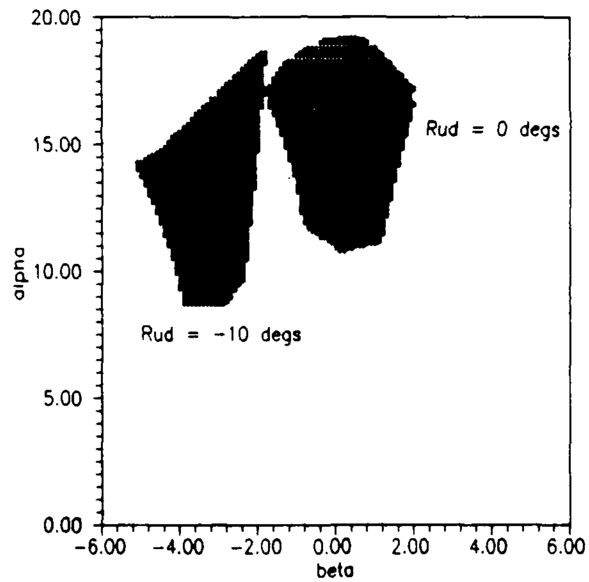
It is true that in most instances a single point at which trim can be achieved is considered to be sufficient. For the investigation performed here, two reasons are advanced for why it is desirable to achieve a large trim region. First, in the event that a control surface fails at some large deflection, the accompanying forces and moments generated may be so large that the aircraft will move rapidly



Equilibrium Area
for
Case A q = 75



Equilibrium Area
for
Case B q = 75



Equilibrium Area
for
Case C q = 75

Figure 27 Equilibrium Regions for Flight Condition I

towards departure. A large equilibrium region indicates that with the correct application of controis, equilibrium may be regained with a greater degree of certainty and ease then if equilibrium can only be achieved at some obscure location in α/β space. Second, given that equilibrium can be obtained, issues of residual control authority and aircraft orientation, become first order considerations. It is postulated that the larger trim region will allow for greater latitude in selecting a trim location that is preferable in light of the considerations listed above.

Table 8 Areas of Equilibrium Regions Rudder = 0

Flight Condition	I	II
Case A	0.35	5.95
Case B	1.03	37.75
Case C	20.61	38.18

Table 9 Areas of the Equilibrium Regions Rudder = -10

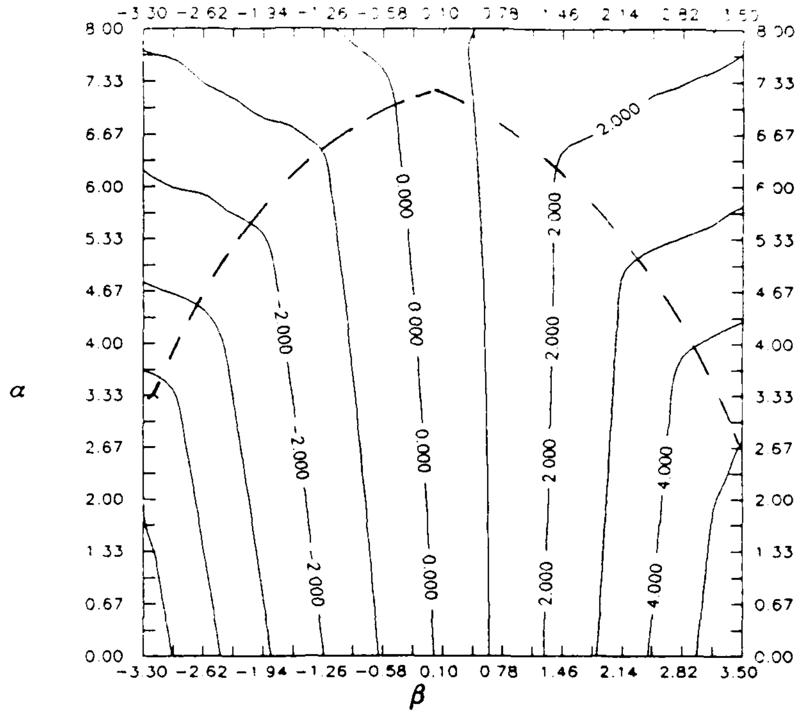
Flight Condition	I	II
Case A	.24	5.14
Case B	2.32	27.3
Case C	18.75	28.93

Table 10 Areas of the Equilibrium Regions Rudder = -25

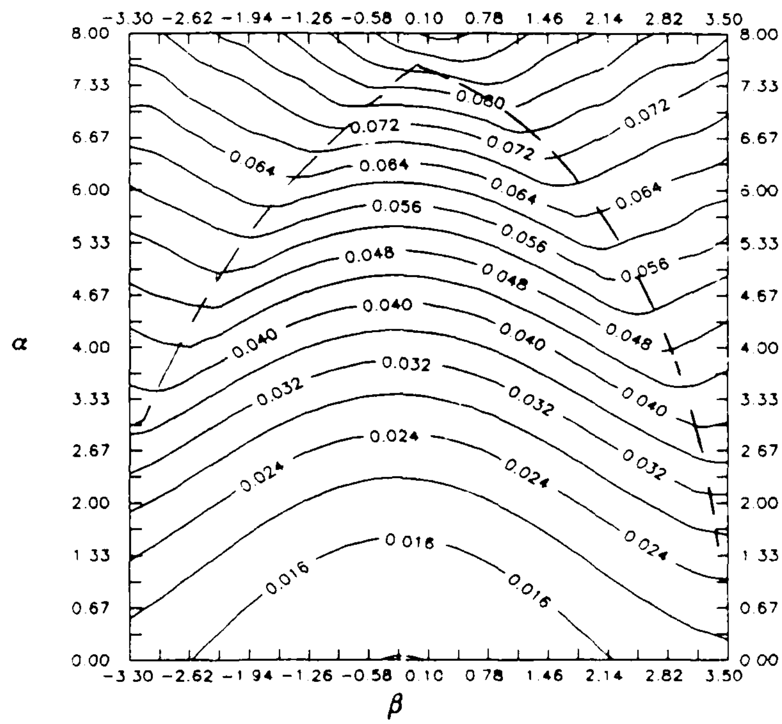
Flight Condition	I	II
Case A	0	.81
Case B	0	3.42
Case C	0	6.85

To investigate the aircrafts orientation and characteristics within the regions of equilibrium, contour plots of the following were constructed: aircraft roll angle, total drag coefficient, residual pitch authority, and residual roll authority. The residual authorities are percentages representing of the maximum authority remains that could be developed by the functional surfaces at that point in α/β space. Note that the dashed lines define the boundary of the trim region defined earlier. Taken together the plots in Figures 28 -29 provide a fairly complete picture of the aircraft characteristics with the rudder locked in a neutral position. As β is increased there is a steady increase in the aircraft roll angle. This result is consistent with the observation made in Chapter III that the rudder is the only control surface which effectively counters aircraft side force. Hence if the rudder is unavailable, and equilibrium must be maintained, some roll angle must be sustained. Since these plots were developed for Flight Condition II, the aircraft is not limited by its pitching authority. Further, Figure 29 provides a clear indication of the lateral freedom which is available and what the attending costs are in reduced residual control authority, aircraft drag, and roll angle.

This point is sharpened by observing a similar set of plots (Figures 30-31) which were generated for the Case B control scheme at Flight Condition II but now with the rudder locked at a deflection of -10 degrees. Here it is evident that the preferred location within the equilibrium region is driven by what is most important to the pilot. If maintaining maximum control authority is a first order consideration, Figures 30 and 31 show the pilot that he must be willing to accept flight in an unsymmetric orientation of about three degrees of β and eight degrees of roll angle. Conversely, if he desires to fly as close to a symmetric condition as possible, he can approach it at this flight condition, but at the substantial price of retaining only twenty percent of his pitch authority and forty percent of his roll authority. Minimizing the drag coefficient, as seen in Figure 30, would require trimming at a slightly lower AOA. Also note, that even as the aircraft approaches the zero β condition, the roll angle is not zero here.



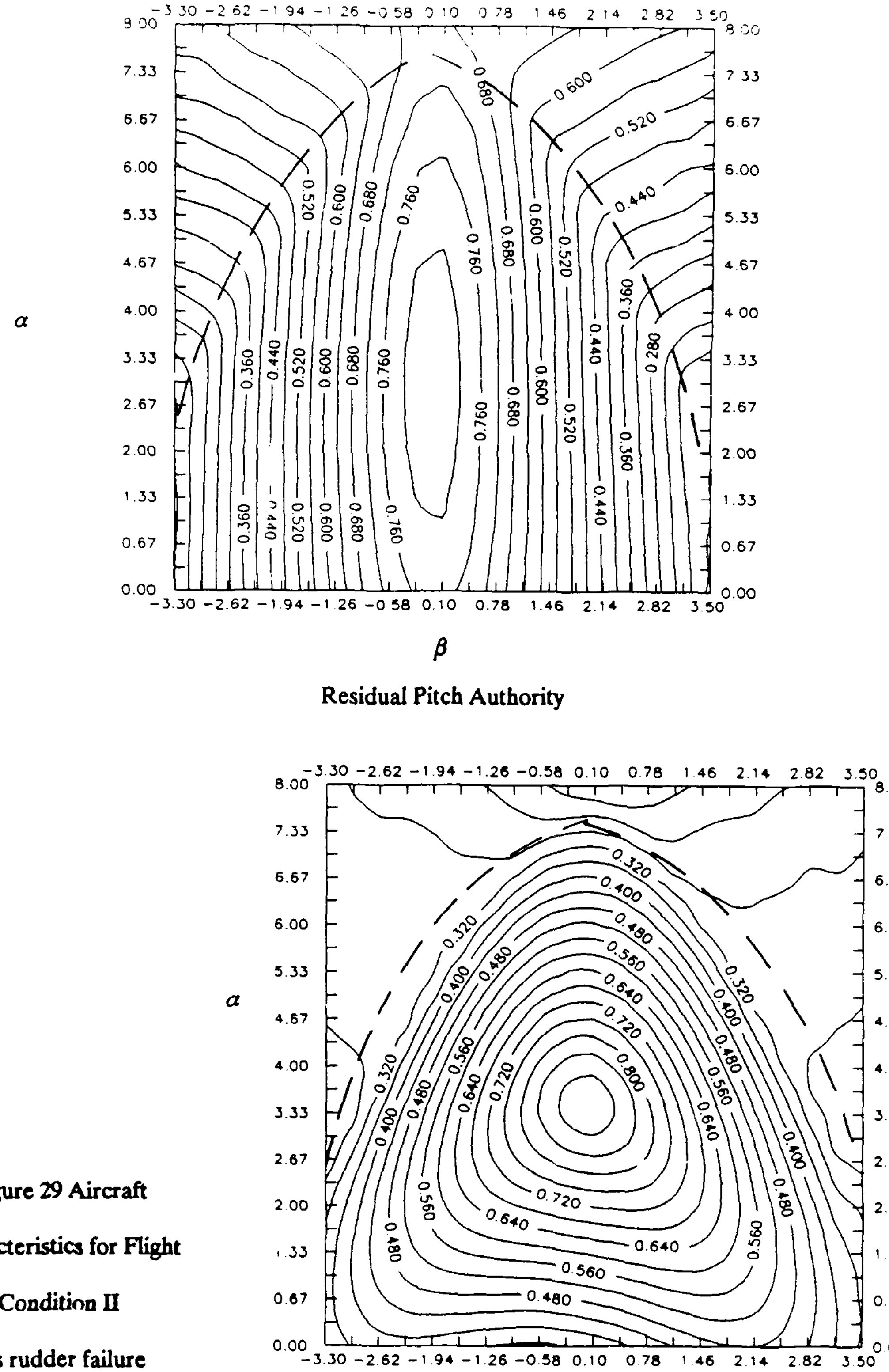
Roll Angle



Drag Coefficient

Figure 28 Aircraft Characteristics for Flight Condition II

0 Degrees rudder failure



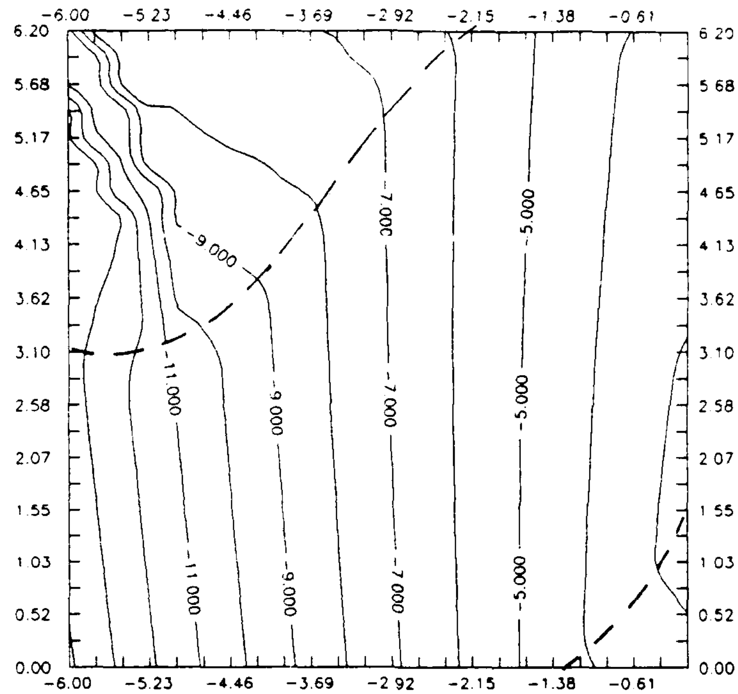
Residual Pitch Authority

Residual Roll Authority

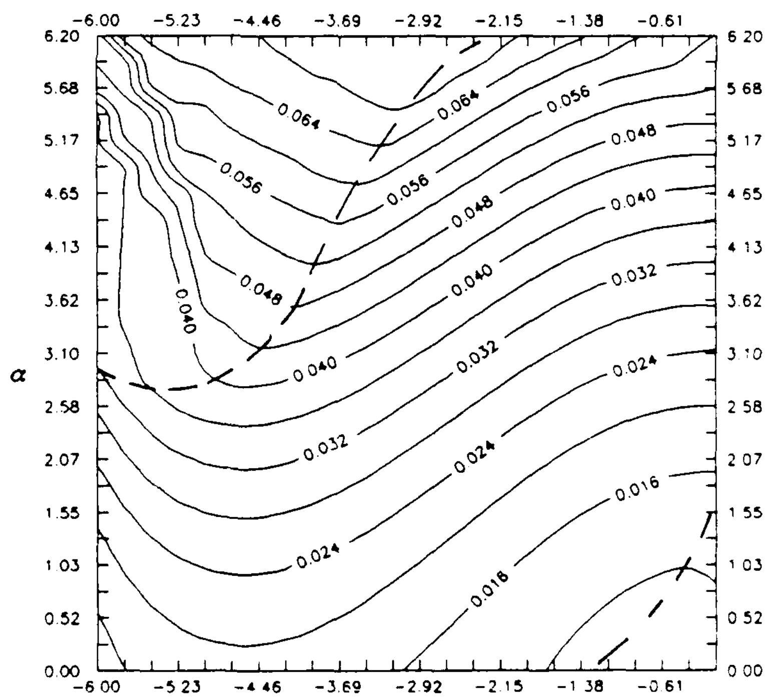
Figure 29 Aircraft Characteristics for Flight Condition II

0 degrees rudder failure

Residual Control Authorities

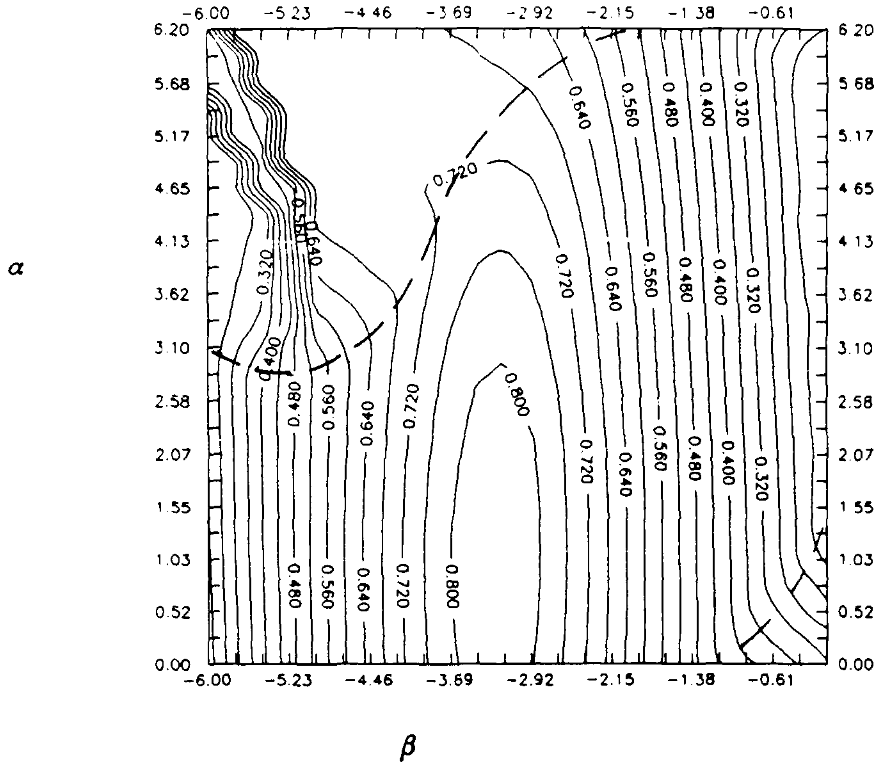


Roll Angle



Drag Coefficient

Figure 30 Aircraft Characteristics for Flight Condition II
-10 Degrees Rudder Failure



Residual Pitch Authority

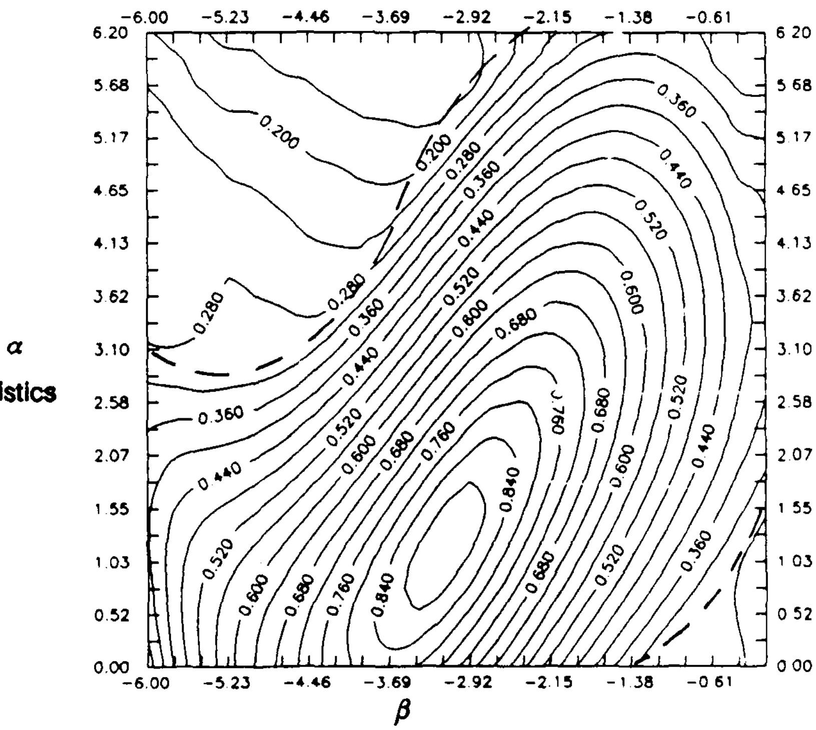


Figure 31 Aircraft Characteristics for Flight Condition II

-10 Degrees Rudder Failure

Residual Control authorities

Residual Roll Authority

Further Insight into the Trim Problem

Two matrix decomposition techniques were used to provide additional insight into the characteristics of the aircraft within a trim region. Through the use of the Singular Value Decomposition of the matrix containing the control derivatives of the various controls used in achieving trim it was possible to define the vectors of control deflection which span the null space. By manipulating the Row Reduced Echelon Form of the problem in the manner discussed in Chapter IV it was possible to define what the allowable control deflections at a particular point in α/β space are. Further insight into the interrelationship of the control surfaces in achieving trim was also obtained via this decomposition. Although not used in this manner here, this technique also defines the range of failures that can be sustained by any two additional surfaces.

Four points from the equilibrium region of Case C at Flight Condition I, Figure , were selected for study and these points are listed in Table 11. Essentially, they represent the extremes in α and β at which trim could be effected. Performing the row reduction of the augmented matrix for points 1 and 2

Table 11 Investigation Points

	q	α	β
Point 1	75	12	0.0
Point 2	75	18	-0.8
Point 3	75	17	-1.8
Point 4	75	17	1.8

and manipulating as was discussed in Chapter IV yields equations (5.1) and (5.2).

$$\begin{bmatrix} LFL \\ RFL \\ LHT \\ RHT \end{bmatrix} = \begin{bmatrix} 15.27 \\ 16.76 \\ 21.66 \\ -23.20 \end{bmatrix} + LLEF \begin{bmatrix} 0.12 \\ -0.02 \\ -0.34 \\ 0.24 \end{bmatrix} + RLEF \begin{bmatrix} -0.13 \\ 0.26 \\ 0.37 \\ -0.50 \end{bmatrix} \tag{5.1}$$

$$\begin{bmatrix} LFL \\ RFL \\ LHT \\ RHT \end{bmatrix} = \begin{bmatrix} -19.08 \\ -12.21 \\ -18.73 \\ -25.64 \end{bmatrix} + LLEF \begin{bmatrix} 0.13 \\ -0.05 \\ -0.41 \\ 0.30 \end{bmatrix} + RLEF \begin{bmatrix} -0.19 \\ 0.34 \\ 0.50 \\ -0.62 \end{bmatrix} \tag{5.2}$$

Here, the first column of numbers represents the control deflections that the control surfaces listed on the left would have to take on to achieve trim for the LEFs set at zero. The range of allowable solutions then contains any combination of deflections of the LEFs that does not lead to a violation of the deflection constraints of the other control surfaces. Points 1 and 2 represent the minimum and maximum AOA at which trim may be achieved for Case C. If the LEFs are set at their scheduled values for the respective AOAs these two equations would represent the Case B solution at these two points. Note that without an asymmetric deflection capability, equation (5.2) would not represent a solution due to the violation of the deflection limit on the RHT. Another point of interest is that the total elevator deflection changes very little between the low AOA to the higher AOA at point two; -22.43 and -22.18 degrees, respectively. What does change dramatically is the employment of the flaperons, which experience a complete change of sign indicating that at some intermediate AOA the flap deflection is approximately zero. A final note is that the sign combinations on the LEF terms remain consistent between equation (5.1) and (5.2) illustrating that there is not some fundamental change in the interaction of the control surfaces from point one to two. The null spaces are spanned by the two vectors obtained in the singular value decomposition. Any combination of the control surfaces within the span of these vectors, at each point, will result in a zero input to the force and moments contained in the b

Table 12 Null Vectors at Points 1 and 2

Point 1	Point 2
$\begin{bmatrix} 0.139 \\ -0.142 \\ -0.401 \\ 0.397 \\ 0.631 \end{bmatrix} \begin{bmatrix} 0.000 \\ 0.167 \\ 0.018 \\ -0.175 \\ 0.692 \end{bmatrix}$	$\begin{bmatrix} 0.163 \\ -0.211 \\ -0.404 \\ 0.442 \\ 0.444 \end{bmatrix} \begin{bmatrix} 0000 \\ 0.145 \\ -0.056 \\ -0.104 \\ 0.810 \end{bmatrix}$

vector of the linear problem: Normal force, Pitching moment, Rolling moment, and Yawing moment.

Points 3 and 4, see Table 11, represent the aircraft at the AOA at which the largest latitude in β exists for this flight condition. The appropriate augmented matrices and manipulations lead to equations (5.3) and (5.4).

$$\begin{bmatrix} LFL \\ RFl \\ LHT \\ RHT \end{bmatrix} = \begin{bmatrix} -20.73 \\ -5.17 \\ -12.88 \\ -31.26 \end{bmatrix} + \text{LLEF} \begin{bmatrix} 0.11 \\ -0.03 \\ -0.36 \\ 0.24 \end{bmatrix} + \text{RLEF} \begin{bmatrix} -0.17 \\ 0.31 \\ 0.45 \\ -0.56 \end{bmatrix} \tag{5.3}$$

$$\begin{bmatrix} LFL \\ RFL \\ LHT \\ RHT \end{bmatrix} = \begin{bmatrix} -9.06 \\ -22.34 \\ -30.95 \\ -13.16 \end{bmatrix} + \text{LLEF} \begin{bmatrix} 0.16 \\ -0.05 \\ -.038 \\ 0.30 \end{bmatrix} + \text{RLEF} \begin{bmatrix} -0.13 \\ 0.28 \\ -0.39 \\ -0.54 \end{bmatrix} \tag{5.4}$$

At these points it can be observed that both the flaperons and the horizontal tails are taking on large asymmetric deflections to generate the lateral forces and moments required to hold the aircraft in equilibrium. Note the reversals in the magnitudes of the control deflections which occur as the aircraft traverses from negative β, point 3, to positive β at point 4. Again it is evident that with out the aid of the LEFs deployed in an asymmetric fashion, the constraints on the deflections of the control surfaces cannot be met. The sign combinations observed in equations (5.1) and (5.2) for the LEFs are maintained in equations (5.3) and (5.4) indicating that a fundamental change in the relationship of the control surfaces has not occurred in the β range traversed. The null vectors associated with points 3 and 4 are listed below.

Table 13 Null Vectors at Points 3 and 4

Point 3

$$\begin{bmatrix} 0.148 \\ 0.200 \\ -0.432 \\ 0.445 \\ 0.433 \\ -0.605 \end{bmatrix} \quad \begin{bmatrix} 0.000 \\ 0.143 \\ -0.065 \\ -0.094 \\ 0.831 \\ 0.525 \end{bmatrix}$$

Point 4

$$\begin{bmatrix} 0.158 \\ -0.141 \\ -0.415 \\ 0.415 \\ 0.671 \\ -0.401 \end{bmatrix} \quad \begin{bmatrix} 0.000 \\ 0.171 \\ 0.045 \\ -0.210 \\ 0.628 \\ 0.728 \end{bmatrix}$$

Summary

In this chapter the results of the investigations into the availability of a trim solution for an aircraft which has sustained a failure of the rudder were discussed. It was demonstrated that even when the aircraft sustained a "hardover" failure of the rudder, trim was achievable at realistic flight conditions. Further, all three of the proposed control schemes were capable of achieving this condition. It was also shown, however, that a return to wings level, zero sideslip flight may not be possible. Even the allowance for complete independence of the remaining control surfaces did not significantly alter this finding. Though the necessity for flight in an unsymmetric orientation might not be desirable, it should not obscure the finding that the aircraft can still maintain constant altitude, rectilinear flight, even when it has sustained the most severe failure of the rudder.

Through the use of plots illustrating the regions in α/β space at which trim could be achieved for the three different control schemes, the advantages offered by each scheme were demonstrated. The most dramatic expansion of the trim region was observed at Flight Condition I when the six control surfaces were allowed to operate with complete independence. This augmentation results from employing the LEFs in an independent manner in a region of α/β space where they have gained effectiveness relative to the other surfaces.

The existence of preferred locations within the regions was demonstrated by the use of contour plots of the aircraft roll angle, drag coefficient, and residual pitch and roll authorities. For partial failures of the rudder it may be possible to orient the aircraft close to symmetric flight but it was shown that there are resulting penalties to be paid in the form of reduction of residual control authorities. By decomposing the problem with row reduction of the augmented matrix of the linear problem formulated in Chapter IV it was possible to gain a better "feel" for how the controls deflected at different points in α/β space.

VI CONCLUSIONS AND RECOMMENDATIONS

In the introduction chapter of this thesis it was stated that this research would encompass a thorough investigation of the stability characteristics of an aircraft which had sustained damage to a primary control surface. This analysis was carried out by formulating functional representations of wind tunnel data for an F-16. The polynomials developed from this data were examined to identify coupling which might be significant. This data was then used to perform a nonlinear analysis which defined the regions in α/β space in which equilibrium could be maintained when the aircraft sustained a failure of the rudder. The following paragraphs provide a summary of the observations and conclusions of this research.

Coupling Effects

The contour plots that were constructed to observe the variation of the aerodynamic coefficients indicated that there was a significant variation in the longitudinal coefficients as a function of β. This variation was symmetric about β equals zero. Not only was this variation observed in these plots, but the trim evaluations performed later also were effected. A slightly unsymmetrical orientation resulted in trim being achieved at a lower angle of attack. A coupling of α and β was also noticed in the lateral coefficients at the higher angles of attack. Plots of the normalized control derivatives provided several key insights. The most significant of these was the indication that the rudder is the only control surface which is effective in generating side force on the aircraft. The flaperons and horizontal tails proved to be of the same order of magnitude for most of the forces, leading to the conclusion that a failure of one of these surfaces can be effectively addressed with the remaining surfaces. The leading edge flaps, which at low angles of attack were not particularly significant relative to the other surfaces, became effective with respect to the other controls as the aircraft AOA was increased

A final coupling effect, which was not actually aerodynamic in nature but was of importance, involved one of the angular relationships used to describe the aircraft orientation. In Appendix D it was demonstrated that the expression usually employed to relate aircraft pitch angle to flight path angle is not satisfactory for analysis that will occur in asymmetric orientations. The appropriate relationship was derived in Appendix D and used in the analysis performed in this thesis.

Equilibrium Evaluations

The equilibrium analysis performed in this thesis indicated that, with the control surfaces currently on the F-16, it is possible to place the aircraft in state of constant altitude, rectilinear flight when the aircraft has sustained a failure of the rudder. In fact, all three of the control schemes investigated in this thesis, trimmed the aircraft even when a maximum deflection of the rudder was the indicated failure. While trim could not be achieved at all flight conditions with this failure, and the resulting orientation was unsymmetrical, the fact remains that a hardover failure of the rudder need not imply a departure of the aircraft. It was also demonstrated, that although the rudder is the dominant control surface; employing the remaining control surfaces with complete independence gave the aircraft a limited ability to affect its lateral characteristics. This finding is particularly significant for failures of the rudder which leave it free floating or remove it entirely. For these failures, the rudder does not generate unwanted forces and moments which must be overcome by the remaining surfaces.

The characteristics of the aircraft, within the regions of α/β space where trim could be achieved, were examined to gain a better understanding of the implications of a failed rudder. It was observed, that there were both benefits and penalties associated with being located at a particular position in the trim region. For instance, the equilibrium location at which the aircraft retained the maximum amount of residual control authority might result in the aircraft oriented with significant sideslip and roll angles. Conversely, if the pilot desires an orientation of the aircraft which is nearly symmetric, there is a corresponding reduction in residual control authority.

The advantages to be gained by employing the control surfaces with greater independence, were most evident at the high AOA associated with the lower dynamic pressure of Flight Condition I. At this higher AOA, allowing the leading edge flaps to deflect independently provided a significant augmentation of the trim region. Most notably, the region was expanded in β; demonstrating an improved capability to affect the lateral orientation of the aircraft. These observations, as well as those discussed above, indicate that employing the control surfaces currently on the F-16 with greater independence, provides an effective means of compensating for a failure of the rudder. A fully satisfactory solution, however, will require an additional control surface which is effective in generating side force and yawing moment. Thrust vectoring might also be a means of imparting the forces and moments needed to offset the negative effects of the failed rudder.

Recommendations

It would seem that most investigations generate more questions then they ever answer. Relative to the work performed in this thesis four recommendations for follow on work are proposed. First, the failure of control surfaces other then the rudder should be investigated using the methods used in this thesis. While information about other failures can be deduced from the investigations performed here, a more thorough study would provide clearer insight. Further, it is possible that the advantages to be gained from allowing greater independence among the control surfaces are more significant then observed in this study. Investigating another failure mode might highlight a clear advantage of one control scheme over another.

Second, there are two entire sets of data taken by Turhal [12] that were not subjected to complete analysis in this research. The wind tunnel data for the floating left flaperon and missing left flaperon cases should be curve fit and subjected to the same analysis performed here. The curve fitting routines and the methodology developed for performing the analysis are either currently set up to perform this investigation or could easily be modified to do so. This analysis would provide important information regarding the implications of a dual failure mode.

Third, a dynamic analysis should be performed of the aircraft, where the model has been formulated to account for the aircraft trimmed in the unsymmetrical orientation. How will the aircraft respond if it is trimmed in an unsymmetrical orientation? How has the aircraft response been limited if the aircraft has been located at the preferred orientation of wings level with the attending penalties in residual control authorities? These are important questions; which are very pertinent to fully describing the dynamic characteristics of the aircraft which has sustained a rudder failure.

Fourth, a similar study should be performed using an aircraft that has some means, other than the rudder, for effectively generating side force and yawing moment.

APPENDIX A

For the experimental data recorded in Turhal's research, [12] each of the six force or moment coefficients was a function of three variables; Angle of Attack, Sideslip angle, and the deflection of a single control surface.

$$C_f = C_f(\alpha, \beta, \delta) \tag{A.1}$$

Three model configurations were investigated; all control surfaces fixed at zero and one surface varying, the left flaperon floating free and one other surface varying, and the left flaperon missing with one surface varying. A detailed discussion of the experimental procedure may be found in [12].

Obviously it is not practical to investigate every point in the $\alpha/\beta/\delta$ space. Therefore, experiment data for a representative sampling of discrete data points was recorded. For the investigation performed in this theses, however, some form of functional representation of the data was required. A least squares curve fitting technique was chosen as a method for creating a function which approximates the behavior of the experimental data. The following is a general discussion of the technique used to curvefit the force and moment coefficients and follows the development of [12].

Given a dependent variable C_f and a vector of independent variable $\overline{X}$ the behavior of C_f can be approximated by a predictor equation of the following form

$$C_{f(\bar{x})} = \sum_{i=0}^{I} a_i \Phi_i(\bar{x}) \tag{A.2}$$

Where $\Phi_i(\overline{X})$ is an arbitrary function. At a particular value of $\overline{X}$, the error between the observed value of Cf and the predicted value will be

$$E_j = C_{f_j} - \sum_{i=0}^{I} a_i \Phi_i (\bar{x}_j) \tag{A.3}$$

In determining the sum of the errors it is important to recognize that not only are negative errors as significant as positive ones but also that the subsequent cancelling that occurs in summing the errors is undesirable. For these reasons, the error at each value of $\overline{X}$ is squared prior to the summation operation.

The total square error is then written as

$$E_T^2 = \sum_{j=1}^{J} E_j^2 = \sum_{j=1}^{J} \left\{ C_{f_j} - \sum_{i=0}^{I} a_i \Phi_i (\bar{x}_j) \right\}^2 \tag{A.4}$$

To find the coefficients which will result in a curve fit with the minimum sum of the square errors *the expression for* t^2 *is differentiated partially with respect to each coefficient* Ai. *The corresponding* equations are then set equal to zero.

$$\frac{\delta E_T^2}{\delta a} = 2 E_T \frac{\delta E_T}{\delta a} \tag{A.5}$$

$$\frac{\delta E_T^2}{\delta a_0} = \sum_{j=1}^{J} \left[C_{f_j} - \sum_{i=0}^{I} a_i \Phi_i (\bar{x}_j) \right] [-2\Phi_0(\bar{x}_j)] = 0 \tag{A.6}$$

$$\frac{\delta E_T^2}{\delta a_1} = \sum_{j=1}^{J} \left[C_{f_j} - \sum_{i=0}^{I} a_i \Phi_i (\bar{x}_j) \right] [-2 \Phi_1 (\bar{x}_j)] = 0 \tag{A.7}$$

$$\vdots \qquad\qquad \vdots \qquad\qquad \vdots$$

$$\frac{\delta E_T^2}{\delta a_I} = \sum_{j=1}^{J} \left[C_{f_j} - \sum_{i=0}^{I} a_i \Phi_i (\bar{x}_j) \right] \left[-2 \Phi_I (\bar{x}_j)\right] = 0 \tag{A.8}$$

The I + 1 equations can then be places into a Matrix equation

$$[A]\{a\} = \{b\} \tag{A.9}$$

where [A] is a symmetric matrix of the following form

$$\begin{bmatrix} \sum_{j=1}^{J} \Phi_0^2(\bar{x}_j) & & & & \\ \sum_{j=1}^{J} \Phi_0(\bar{x}_j)\,\phi_1(\bar{x}_j) & \sum_{j=1}^{J} \Phi_1^2(\bar{x}_j) & & & \\ \sum_{j=1}^{J} \Phi_0(\bar{x}_j)\,\Phi_2(x_j) & \sum_{j=1}^{J} \Phi_1(\bar{x}_j)\Phi_2(\bar{x}_j) & \sum_{j=1}^{J} \Phi_2^2(\bar{x}_j) & & \\ \vdots & \vdots & \vdots & \ddots & \\ \sum_{j=1}^{J} \Phi_0(\bar{x}_j)\Phi_I(\bar{x}_j) & \sum_{j=1}^{J} \Phi_1\,\Phi_I & \cdots & \cdots & \sum_{j=1}^{J} \Phi_I^2(\bar{x}_j) \end{bmatrix} \tag{A.10}$$

Since [A] is square and nonsingular, the coefficients of Ai may be found using

$$\{a\} = [A]^{-1}\{b\} \tag{A.11}$$

For the curve fitting accomplished in this thesis the function Φ was chosen to be a polynomial in three variables. Therefore, C_f took on the following form

$$C_f = \sum_{\ell=0}^{L} \sum_{m=0}^{M} \sum_{n=0}^{N} a_{\ell mn} \, \alpha^{\ell} \beta^{m} \delta^{n} \tag{A.12}$$

Where the values of 'A' determined from solving the linear problem become the coefficients which multiply the respective polynomial terms. For the work done in this thesis, it was helpful to separate those polynomial terms not associated with a control surface deflection from those which were. The zero defection terms were written as

$$C_{f_0} = C_f(\alpha,\beta) = \sum_{\ell=0}^{L} \sum_{m=0}^{M} a_{\ell m} \, \alpha^{\ell} \beta^{m} \tag{A.13}$$

In effect, this set of terms represents the variation of the force or moment coefficients with all the control surfaces set equal to zero. The total value of the force or moment coefficient including the effects of each control surface is then written as

$$C_{f_T}(\alpha,\beta,\delta) = C_{f_0} + \sum_{k=1}^{7} C_{f_k}(\alpha,\beta,\delta) \tag{A.14}$$

$$C_{f_t} = \sum_{\ell=0}^{L} \sum_{m=0}^{M} a_{\ell m} \, \alpha^{\ell} \beta^{m} + \sum_{k=1}^{7} \sum_{\ell=0}^{L} \sum_{m=0}^{M} \sum_{n=1}^{N} B_{\ell mn} \, a_{\beta}^{\ell m} \, \delta_k^{n} \tag{A.15}$$

The FORTRAN computer codes used to accomplish the assembly and curve fitting of the wind tunnel data may be found in Appendix B. There codes are currently configured for and will compile on a UNIX operating system.

It should be noted that the final summation is over the seven independent control surfaces. It was assumed for the curve fitting that the contribution of each controi surface may be summed together via the superposition principle, and that each control surface was of the form

$$C_{f\delta i} = \left\{ C_{\alpha_\delta} \alpha + C_{\beta_\delta} \beta + C_\delta \right\} \delta_i \tag{A.16}$$

APPENDIX B

Contained in this appendix are the two FORTRAN codes which were used to perform the curvefits of Turhal's wind tunnel data. Polyfitb.for is actually just a variation of Polyfita.for which was used to develop the predictor equations for the control derivatives. Its primary difference from Polyfita.for is that it calls in the polynomial fits that were derived from the "zero" case data and then uses these polynomials as the basis upon which the curve fits of the other data sets is built up from. A more detailed discussion of this procedure is given in Chapter II. The programs are formatted to operate with a UNIX operating system.

Polyfita.for

```
c  ********************************************************
c           POLYFITA.FOR
c  ********************************************************
c
c  Maj L. Hudson
c  Capt S. Zaiser
c
c  This program will perform a liest squares curve fit on
c  experimental data which is read into the program from existing
c  guide and data files.  The required forms for these files may be
c  found in comments in the program.  The program is currently configured
c  to attempt to fit the experimental data with a predictor equation which
c  has a polynomial form and is a function of three variables; alpha, beta,
c  and delta.  A detailed discussion of the theory of this program may be
c  found in Appendix A of the thesis.  For ease of use the program is menu
c  driven.
c
c    See Numerical Recipees for subroutines SVDCMP AND SVBKSB
c  version  01 Aug 89  SMZ
c        16 Sep 89
c   The guiding file should have the following format:
c
c   line 1: title
c   line 2: output file name. The curve fit coef end up in this file
c        in the form of data statements.
c   line 3: surface designation, appended to the results  6 characters.
c   line 4: nx,ny,nz the highest powers desired in the curve fits.
c   line 5: number of data files to be read.
c
      implicit real*8 (a-h,o-z)
c
      parameter (max_data=5000)
c
      real*8 limits(3,2),error
      real*8 x_data(3,max_data),y_data(max_data),choice
      real*8 coef(100,6),x(3,1000),w(1000,6),s(13)
c
      character*50 filename,outfile,guide,sfile,gfile
      character*80 title,author,data,facility,date
      character*5  surface
      character*2  force
      character*1 answer
c
      integer cons,var1,var2
      integer ia(100,6),ib(100,6),id(100,6),no_fcns(6),i_task
      common /pwrs / ia,ib,id,no_fcns,i_fcn
c
      external f3
      external sqr_err
c
 10   print *, '********************************************'
      print *, '       MENU                    '
      print *, '********************************************'
      print *, ' '
      print *, '************************************************'
      print *, '1.  Read in data files'
      print *, '2.  Curve fit data'
      print *, '3.  Graph data and curve fit'
      print *, '4.  Evaluate the square error'
      print *, '5.  Write results to file'
      print *, '6.  Create graphing files'
      print *, '7.  Exit program'
      print *, '************************************************'
      print *, ' '
      print *, 'Enter selection:'
      read *, choice
c
c
      if (choice.eq.1) then
        goto 20
       else if (choice.eq.2) then
        goto 200
       else if (choice.eq.3) then
        write(6,*) 'Disabled return to menu'
        goto 10
       else if (choice.eq.4) then
        goto 675
       else if (choice.eq.5) then
        goto 725
       else if (choice.eq.6) then
        goto 625
       else if (choice.eq.7) then
        goto 750
       else
        goto 10
       endif
c
c The data files are in the form
c column no.   item description
c
c   1        item number for the data file.
c   2        angle of attack.
c   3        dynamic pressure
```

```
c  4      yawing moment coefficient
c  5      rolling moment coefficient
c  6      pressure coefficient
c  7      lift coefficient
c  8      drag coefficient
c  9      pitching moment coefficient
c  10     side force coefficient
c  11     yaw angle
c  12     not used ---- 0.0
c  13     not used ---- 0.0
c
c
c   the column identifiers for respective values
 20  ibeta=11
     ialpha=2
     ilift=7
     idrag=8
     iside=10
     iroll=5
     ipitch=9
     iyaw=4
c
c Enter the name of the file containing the information to guide the program
c through the data input process.
c
     write(6,*)'Enter the guide file name.'
     read(5,31000) guide
     write(6,31000) guide
     open(15,file=guide,status='old')
c
c  The guide file contians:
c   1. Title of the guide file.
c   2. Outfile name.(not used)
c   3. The control surface which is being varied.
c   4. Not used.
c   5. The number of data files listed in the guide file.
c
     read(15,10000) title
     write(6,*) title
     read(15,82000) outfile
     write(6,*) outfile
     read(15,83000) surface
     write(6,*) surface
     read(15,*) nx,ny,nz
     read(15,*) nfiles
     write(6,*) 'The number of files = ',nfiles
c
c  Initialize maximum values to zero.
c
     npts=0
     k=0
c
     a_min=1.0e15
     b_min=1.0e15
     d_min=1.0e15
     a_max=-1.0e15
     b_max=-1.0e15
     d_max=-1.0e15
c
c    Open Guide files and assign the wind tunnel data
c    to the appropriate arrays for curve fitting.
c
     do 100 i=1,nfiles
     read(15,90000,end=75,err=85) (filename(i1:i1),i1=1,19),delta
       write(6,30000) filename
       open(14,file=filename,status='old')
c
c
       do 50 j=1,60
         read(14,*,end=75,err=85) (s(k1),k1=1,13)
         k=k+1
c
c        The x matrix contains the values of alpha,
c        beta, and surface deflection respectively
c
         x(1,k)=s(ialpha)
c
c        The sign on the beta readings is negated to conform
c        to standard convention (wind from the right).
c
         x(2,k)= (-1) * s(ibeta)
         x(3,k)=delta
c
         if(x(1,k).lt.a_min) a_min=x(1,k)
         if(x(2,k).lt.b_min) b_min=x(2,k)
         if(x(3,k).lt.d_min) d_min=x(3,k)
         if(x(1,k).gt.a_max) a_max=x(1,k)
         if(x(2,k).gt.b_max) b_max=x(2,k)
         if(x(3,k).gt.d_max) d_max=x(3,k)
c
c
         w(k,1)=s(ilift)
         w(k,2)=s(idrag)
         w(k,3)=s(iside)
         w(k,4)=s(ipitch)
         w(k,5)=s(iroll)
         w(k,6)=s(iyaw)
c
50     continue
75     close(14)
      goto 100
85     write(6,*)'Have had an error in reading ',filename
      close(14)
100  continue
    close(15)
    npts=k
c
c
    write(6,*) 'The data varied as follows:'
    write(6,*) a_min,' alpha ',a_max
    write(6,*) b_min,' beta ',b_max
    write(6,*) d_min,' delta ',d_max
c
c
c
c   Return to menu
c
    goto 10
200  write(6,*)'Which force coefficient do you wish to curve fit?'
    write(6,*) '(Enter the corresponding number.)'
    write(6,*) '1 lift'
    write(6,*) '2 drag'
    write(6,*) '3 side force'
    write(6,*) '4 pitching moment'
    write(6,*) '5 rolling moment '
    write(6,*) '6 yawing moment'
    write(6,*) '7 or greater for listing the data'
    read(5,*) ifit
    if(ifit.gt.6) then
c
      write(6,*) 'Which data do you wish to examine?'
      write(6,*) '(Enter the corresponding number.)'
      write(6,*) '1 lift'
      write(6,*) '2 drag'
      write(6,*) '3 side force'
      write(6,*) '4 pitching moment'
      write(6,*) '5 rolling moment '
      write(6,*) '6 yawing moment'
      write(6,*) '7 alpha'
      write(6,*) '8 beta'
      write(6,*) '9 delta'
      read(5,*) iexam
      if(iexam.lt.6) then
        do 400 ijk=1,npts
          write(6,*) w(ijk,iexam)
 400      continue
        else
        write(6,*) 'iexam-6=',iexam-6
        do 500 ijk=1,npts
          write(6,*) x(iexam-6,ijk)
 500      continue
        end if
      else
c
c       Define the form of the polynomial to be curve fit.
c
      call bldpwr(ifit)
 550    continue
c
c       Perform the lstsqrs curve fit.
c
      call lstsqr (f3,no_fcns(ifit),x,w(1,ifit),npts,coef(1,ifit))
      do 600 ijk=1,no_fcns(ifit)
c
c         Write the results of the curve fit to screen.
c
        write(6,60000) ijk,ia(ijk,ifit),ib(ijk,ifit),id(ijk,ifit),
   x                   coef(ijk,ifit)
 600    continue
      end if
c
c   Return to menu
c
    goto 10
```

```
c
c   GRAPHER AND SURFER FILE CREATION
c
 625  write(6,*)'Which do you want to hold constant?'
      write(6,*)' 1. alpha'
      write(6,*)' 2. beta'
      write(6,*)' 3. delta'
      read(5,*) cons
c
      write(6,*)'What is the first variable?'
      write(6,*)' 1. alpha'
      write(6,*)' 2. beta'
      write(6,*)' 3. delta'
      read(5,*) var1
c
      write(6,*)'What is the second variable?'
      write(6,*)' 1. alpha'
      write(6,*)' 2. beta'
      write(6,*)' 3. delta'
      read(5,*) var2
c
c    initialize limits matrix with variable ranges
c
      limits(1,1)=a_min
      limits(1,2)=a_max
      limits(2,1)=b_min
      limits(2,2)=b_max
      limits(3,1)=d_min
      limits(3,2)=d_max
c
c
 650  write(6,*)'The range on the constant is:'
      write(6,*)limits(cons,1),' constant ',limits(cons,2)
      write(6,*) 'Do you want to change it?'
      read(5,80000) answer
      if(answer.eq.'y' .or. answer.eq.'Y') then
        write(6,*)'What should the lower value be?'
        read(5,*) limits(cons,1)
        write(6,*)'What should the upper value be?'
        read(5,*) limits(cons,2)
        goto 650
      end if
c
c   Search for those experimental data points which fall within
c   the variable ranges defined above.
c
      call find_pts(x,w(1,ifit),npts,x_data,y_data,ndata,limits,
     x          cons,var1,var2)
c
c   write the variable values, force or moment values, and
c   evaluated values to a data file for evaluation in 'grapher'
c
      write(6,*) 'Do you want to create a Grapher  file?'
      read(5,80000) answer
c
      if(answer.eq.'y' .or. answer.eq.'Y') then
      write(6,*) 'Enter the file name with .dat:'
      read(5,31000) gfile
c
      open(10,file=gfile,status='unknown')
c
      do 665 i=1,ndata
      val1 = evlsqr(f3,coef(1,ifit),no_fcns(ifit),x_data(1,i))
c
c The Grapher file will have the following form
c column no.   item description
c
c   1        Value of the first variable.
c   2        Value of the second variable.
c   3        The experimental data value at that point.
c   4        The curve fits value at that point.
c   5        Value of the variable held constant
c
      write(10,32000) x_data(var1,i),x_data(var2,i),y_data(i),
     x       val1,x_data(cons,i)
 665  continue
      close(10)
      end if
c
c   write data to a file for evaluation of contour plots in the
c   'surfer' software.  These data files contain predicted values
c   of the force or moment as a function of the two specified variables
c
      write(6,*) 'Do you want to create a Surfer file?'
      read(5,80000) answer
c
      if(answer.eq.'y' .or. answer.eq.'Y') then
      call surf(coef(1,ifit),limits,ifit,cons,var1,var2)
      end if
c
c    Return to menu
c
      goto 10
c
c    Define the r squared value of the curve fit and provide
c    the opportunity to modify the form of the polynomial.
c
 675  error = sqr_err(x,w(1,ifit),npts,coef(1,ifit),ifit)
       write(6,*) '————————————————————————'
       write(6,*) 'The value of r squared is ',error
       write(6,*) '————————————————————————'
      write(6,*)'What do you want next?'
      write(6,*)' 1. Remove powers from the approximating functions'
      write(6,*)'    and refit.'
      write(6,*)' 2. Add powers'
      write(6,*)' 3. Quit'
      read(5,*) i_task
c
      if( i_task.eq.1) then
        write(6,*)' What is the smallest magnitude you wish to keep'
        write(6,*)' in the current fit?'
        read(5,*) bound
        call rm_coef(coef(1,ifit),bound,ifit)
        goto 550
      else if (i_task.eq.2) then
        call addcoef((coef(1,ifit),ifit)
        goto 550
      else
c
c    Return to menu
c
        goto 10
      end if
 725  call output(coef(1,ifit),ifit,error)
c
c   Return to menu
c
      goto 10
c
c
10000 format(a80)
20000 format(2x,e15.8,3(2x,i3),' coef, ia, ib, id ', i3)
30000 format(5x,'reading from ',a40)
31000 format(a50)
32000 format(5(1x,e15.7))
60000 format(' fcn no=',i3,' ia=',i3,' ib=',i3,' id=',i3,'   coef=',
     x    e12.5)
80000 format(a1)
82000 format(a40)
83000 format(a5)
89000 format(i4,' alpha=',f10.5,' beta=',f10.5,' delta=',f5.2,
     x        ' coef=',2f10.5)
90000 format(2x,19a1,3x,f5.2)
 750  STOP
      END
c
c     ********************************************************
c                 LSTSQR
c     ********************************************************
c
      subroutine lstsqr(functn,nf,x,w,npts,coef)
      implicit real*8 (a-h,o-z)
      parameter (msize=100)
c
c functn is an explicit function giving the set of fitting functions
c     to be used in the least squares curve fitting process.
c     It has the following parameter list:
c       functn( nfnc,x,k )
c     The arguments are:
c       nfnc = the identification number of the function to be used.
c       x    = an array of arguments of the function.
c       k    = the index to the argument to use in the evaluation
c              of the function.
c
c nf is the number of functions contained in the family of functions
c     provided by functn.
c
c x    is the array of values at which the known values are given.
c      It may be one dimensional or multi-dimensional.
c
c w    is the array of known values to be curve fitted.
c
c npts  is the number of points to be curve fitted.
```

```
c
c coef  is the array of coefficients weighting the functions
c
      real*8 x(1),w(1),coef(1)
      real*8 a(msize,msize),rhs(msize)
      external functn
c
c   Assemble the matricies to be used in determining the
c   coeficients of the polynomial predictor equation.  A
c   detailed discusion of the composition of these matricies
c   may be found by referencing the thesis.
c
      do 400 i=1,nf
        write(6,*)'Setting up equation no',i
        do 200 j=1,nf
          a(i,j)=0.0
          do 100 k=1,npts
            a(i,j)=a(i,j) + functn(i,x,k)*functn(j,x,k)
100       continue
200     continue
        rhs(i)=0.0
        do 300 k=1,npts
          rhs(i)=rhs(i) + functn(i,x,k)*w(k)
300     continue
400   continue
      write(6,*)'Solving the linear equations in lstsqr.'
c
c     Solve the linear problem which has been setup.
c
        call svd_solve(a,rhs,coef,nf,nf,msize,msize)
c       do     500 i=1,nf
c         write(6,*) i, coef(i)
c 500   continue
      write(6,*)'Finished in lstsqr.'
      return
      end
c
c   *******************************************
c              SQR_ERR
c   *******************************************
c
      real*8 function sqr_err(x,w,npts,coef,i_fn)
c
c
c   The purpose of this subprogram is to calcu..te the value of
c   r squared as a measure of the 'goodness' of the curve fit.
c

      implicit real*8 (a-h,o-z)
      real*8 x(3,1000),coef(1),w(1),error,sum1,nerr,esqr
      real*8 sum2,sum3,rsqr,mean,wmin,wmax
      integer ia(100,6),ib(100,6),id(100,6),no_fcns(6)
      common /pwrs /ia,ib,id,no_fcns,i_fcn
      external f3
      sum1 = 0.0
      sum2 = 0.0
      sum3 = 0.0
      wmin = 1.0e15
      wmax = -1.0e15
      do 100 i = 1,npts
        if(w(i).gt.wmax) wmax = w(i)
        sum1 = sum1 + w(i)
 100 continue
      mean = sum1 / npts
      do 200 i=1,npts
      esqr = evlsqr(f3,coef,no_fcns(i_fn),x(1,i))
      error1 = w(i) - evlsqr(f3,coef,no_fcns(i_fn),x(1,i))
      sum2 = sum2 + (error1)**2
      sum3 = sum3 + (w(i) - mean)**2
 200 continue
      rsqr = 1 - (sum2 /sum3)
      sqr_err = rsqr
      return
      end
c
c
c   *********************************************
c             F3
c   *********************************************
c
      real*8 function f3(i,x,k)
      implicit real*8 (a-h,o-z)
      real*8 x(3,1)
      integer ia(100,6),ib(100,6),id(100,6),no_fcns(6)
      common /pwrs /ia,ib,id,no_fcns,i_fcn
c
      if (i.gt.100) write(6,*) '*** ERR - undeclared function for i=',i
      alpha = x(1,k)
      beta = x(2,k)
      delta = x(3,k)
      f3=poly(ia(i,i_fcn),alpha)
     x        *poly(ib(i,i_fcn),beta)
     x           *poly(id(i,i_fcn),delta)
      return
      end
c
c   ********************************************
c          BLDPWR
c   ********************************************
c
      subroutine bldpwr(i_fn)
c
c The purpose of the subprogram is to systematically create a common stat-
ment
c which defines the polynomial terms to be used for accomplishing the curve
c fit. In general the polynomial will involve combinations of three variables
c See Ref in appendix A of thesis.
c
      integer ia(100,6),ib(100,6),id(100,6),no_fcns(6)
      common /pwrs /ia,ib,id,no_fcns,i_fcn
      integer alp, bet, del, comb
c
      i_fcn = i_fn
 50   write(6,*)'Do you want to'
      write(6,*)' 1. Generate all combinationsof powers'
      write(6,*)' 2. Enter specific combinations of powers'
      read(5,*) ij
c
      if(ij.eq.1) then
        write(6,*) 'What order do you want alpha fit to be?'
        read(5,*) na
        write(6,*) 'What order do you want the beta fit to be?'
        read(5,*) nb
        write(6,*) 'What order do you want the delta fit to be?'
        read(5,*) nd
        k=1
c
c     This routine generates all permutations of the powers specified
c
        do 300 i1=0,na
          do 200 i2=0,nb
            do 100 i3=0,nd
              ia(k,i_fcn)=i1
              ib(k,i_fcn)=i2
              id(k,i_fcn)=i3
              k=k+1
 100        continue
 200      continue
 300    continue
        comb=na*nb*nd
      else if (ij.eq.2)th.. 
c
c     This routine allows t.. specification of a specific set
c     of terms.
c
        k=1
        print *, 'enter the number of combinations desired'
        read *, comb
        do 800 j=1,comb
          print *, 'building combination', k
          print *, 'enter the power on alpha'
          read *, alp
          print *, 'enter the power on beta'
          read *, bet
          print *, 'enter the power on delta'
          read *, del
          ia(k,i_fcn)=alp
          ib(k,i_fcn)=bet
          id(k,i_fcn)=del
          k=k+1
 800  continue
      else
        goto 50
      end if
      do 900 j=1,comb
        print *, ia(j,i_fcn), ib(j,i_fcn), id(j,i_fcn)
 900  continue
      no_fcns(i_fcn) = k - 1
      write(6,*) no_fcns(i_fcn),' functions initialized in bldpwr.'
      return
      end
c
c   ***********************************************
c          EVLSQR
```

```
c  ************************************************
c
      real*8 function evlsqr(functn,coef,nf,x)
      implicit real*8 (a-h,o-z)
c
c This functions evaluates the curve fit at the first point in x.
c
      real*8 coef(1),x(1)
      external functn
c
      evlsqr=0.0
      do 100 i=1,nf
        evlsqr=evlsqr + coef(i)*functn(i,x,1)
100   continue
      return
      end
c
c  ***************************************************
c          POLY
c  ***************************************************
c
c
      real*8 function poly(nfnc,x)
      implicit real*8 (a-h,o-z)
c
c This function returns values of the family of polynomials.
c
c  nfnc gives the power to raise x to.
c
      if(nfnc.eq.0) then
        poly=1.0
      else
       if (x.eq.0.0)  then
         poly=0.0
       else
         poly=x**nfnc
       end if
      end if
      return
      end
c
c  ***************************************************
c          RM_COEF
c  ***************************************************
c
      subroutine rm_coef(coef,bound,i_fn)
      implicit real*8(a-h,o-z)
      real*8 coef(1)
      integer ia(100,6),ib(100,6),id(100,6),no_fcns(6)
      common /pwrs /ia,ib,id,no_fcns,i_fcn
c
c  The purpose of this program is to remove those
c  polynomial terms whose coeficients are smaller than
c  a specified value.  This routine needs work.
c
      write(6,*)'In rm_coef with bound=',bound
      write(6,*)'no_fcns =',no_fcns(i_fn)
      write(6,*)'i_fn=',i_fn
      i = 1
100   continue
        if((dabs(coef(i)).lt.bound) then
         if(i.ne.no_fcns(i_fn)) then
          do 200 j=i,no_fcns(i_fn)-1
           ia(j,i_fn) = ia(j+1,i_fn)
           ib(j,i_fn) = ib(j+1,i_fn)
           id(j,i_fn) = id(j+1,i_fn)
           coef(j) = coef(j+1)
200        continue
         end if
         ia(no_fcns(i_fn),i_fn) = 0
         ib(no_fcns(i_fn),i_fn) = 0
         id(no_fcns(i_fn),i_fn) = 0
         coef(no_fcns(i_fn)) = 0.0
         no_fcns(i_fn) = no_fcns(i_fn) - 1
        end if
        i = i + 1
      if((i.le.no_fcns(i_fn)) goto 100
      write(6,10000) no_fcns(i_fn)
      do 300 i=1,no_fcns(i_fn)
        write(6,20000) i,ia(i,i_fn),ib(i,i_fn),id(i,i_fn)
300   continue
      return
10000 format('  The number of functions is ',i3)
20000 format( i4,' ia=',i3,' ib=',i3,' id=',i3)
      end
c
c  ******************************************
```

```
c       ADDCOEF
c  ******************************************
c
c
c
      subroutine addcoef(coef,i_fn)
      implicit real*8(a-h,o-z)
c
c  The purpose of this subroutine is to provide a
c  means for adding polynomial terms to an existing
c  predictor equation.
c
      real*8 coef(1)
      integer ia(100,6),ib(100,6),id(100,6),no_fcns(6)
      integer comb,newcm
      common /pwrs /ia,ib,id,no_fcns,i_fcn
       print *, 'enter the number of additional combinations desired'
       read *, comb
       k=1 + no_fcns(i_fn)
c
c   Assemble the additional polynomial terms and
c   append them to the existing polynomial.
c
       do 800 j=1,comb
            print *, 'building combination', k
            print *, 'enter the power on alpha'
            read *, alp
            print *, 'enter the power on beta'
            read *, bet
            print *, 'enter the power on delta'
            read *, del
            ia(k,i_fcn)=alp
            ib(k,i_fcn)=bet
            id(k,i_fcn)=del
            k=k+1
 800  continue
      no_fcns(i_fn) = no_fcns(i_fn) + comb
      return
      end
c
c  ******************************************
c       OUTPUT
c  ******************************************
c
      subroutine output(coef,ifit,rsqr)
c
c  The purpose of this subprogram is to write the values of the polynomial
c  coeficients and respective powers to an output file.
c
c
      integer ia(100,6),ib(100,6),id(100,6),no_fcns(6)
      common /pwrs /ia,ib,id,no_fcns,i_fcn
      real*8 coef(1)
      real * 8 rsqr
      character *40 data
      character *14 control, force
      print *, 'Enter the file name'
      read(5,10000) data
      open(16,file=data,status='unknown')
      print *, 'Enter the control surface:'
      read(5,15000) control
      print *, 'Enter the force being fit:'
      read(5,15000) force
      write(16,*) control
      write(16,*) force
      write(16,*) rsqr
      write(16,*) no_fcns(ifit)
        do 100 ijk=1,no_fcns(ifit)
c
c The output file will have the following form
c column no.  item description
c
c  1       Counter of term number.
c  2       Power on alpha for that term.
c  3       Power on beta for that term.
c  4       Power on delta for that term.
c  5       Coeficient associated with that term.
c
          write(16,80000) ijk,ia(ijk,ifit),ib(ijk,ifit),id(ijk,ifit),
     x              coef(ijk)
 100    continue
c
      close(16)
      write(6,*) 'output file complete'
10000 format(a40)
15000 format(a14)
80000 format( 4(1x,i4.2),1x,f10.8)
      return
```

```
      end
c
c     *****************************************************
c              FIND_PTS
c     *****************************************************
c
      subroutine find_pts(x,y,no_data,x_pts,y_pts,no_pts,limits,
     x                   cons,var1,var2)
c
c The purpose of this routine is to search through a set of data points, x,
c and collect the points which fall within the limits specified.
c
c  x     == points in the domain.
c  y     == functional values associated with x.
c  no_data == number of data points total.
c  x_pts  == points which are found within the limits
c  y_pts  == functional values associated with x_pts
c  no_pts == number of points found.
c  limits == the bounds of acceptability on the data points.
c    limit(i,j), j=1 for lower limit
c              j=2 for upper limit
c  constant == the variable which is held constant.
c
      integer cons,var1,var2
      real*8 x(3,1),y(1)
      real*8 x_pts(3,1),y_pts(1)
      real*8 limits(3,2)
c
c
      no_pts = 0
      do 300 i=1,no_data
        if((x(cons,i).lt.limits(cons,1) .or.
     x      x(cons,i).gt.limits(cons,2)) goto 300
c
c If it gets here, then it is within limits.
c
        no_pts = no_pts+1
        x_pts(var1,no_pts)=x(var1,i)
        x_pts(var2,no_pts)=x(var2,i)
        x_pts(cons,no_pts)=x(cons,i)
        y_pts(no_pts)=y(i)
 300  continue
      return
      end
c
c     *****************************************************
c              SURF
c     *****************************************************
c
      subroutine surf(coef,limits,ifit,cons,var1,var2)
c
c  The purpose of this subprogram is to create an array of data for
c  use in 'SURFER'. The first column of data is the first variable, the
c  second the second, the third is the evaluated force or moment value
c  and the fourth column is the value of the variable held constant.
c
      implicit real*8 (a-h,o-z)
      parameter ( no_divs=25)
      character * 40 sfile
      real*8 x(3),dx(3),coef(1),limits(3,2)
      real*8 s(no_divs+1),t(no_divs+1),y(no_divs+1)
      integer ia(100,6),ib(100,6),id(100,6),no_fcns,cons,var1,var2
      common /pwrs /ia,ib,id,no_fcns(6),i_fcn
      external f3
c
c   Generate data array to be plotted.
c
       d_x = (limits(var1,2) - limits(var1,1))/no_divs
       d_y = (limits(var2,2) - limits(var2,1))/no_divs
c
c   Initialize the variable in the x array which is constant
c
       x(1)= 0.5*(limits(1,1) + limits(1,2))
       x(2)= 0.5*(limits(2,1) + limits(2,2))
       x(3)= 0.5*(limits(3,1) + limits(3,2))
c
c
      write(6,*) 'Enter the file name with .dat:'
      read(5,31000) sfile
c
      open(11,file=sfile,status='unknown')
       do 150 j=1,no_divs
        t(j) = limits(var2,1) + (j-1)*d_y
         do 100 i=1,no_divs
          s(i) = limits(var1,1) + (i-1)*d_x
          x(var1)=s(i)
          x(var2)=t(j)
          y(i) = evlsqr(f3,coef,no_fcns(ifit),x)
          write(11,32000) x(var1),x(var2),y(i),x(cons)
 100      continue
 150   continue
       close(11)
       return
31000 format(a40)
32000 format(4(1x,e15.7))
       end
c
c
c    ******************************************
c
SVD_SOLVE
c    ******************************************

      subroutine svd_solve(a,b,x,n,m,np,mp)
      implicit real*8(a-h,o-z)
      Parameter (nmax=100)
      real*8 A(mp,np),W(nmax),V(nmax**2)
c
      call svdcmp(a,n,m,np,mp,w,v)
      wmax = 0.0d0
      do 100 j=1,n
        if (w(j).gt.wmax) wmax = w(j)
 100  continue
      wmin = wmax*1.0d-12
      do 200 j=1,n
        if(w(j).lt.wmin) w(j) = 0.0d0
200   continue
      call svbksb(a,w,v,n,m,np,mp,b,x)
      return
      end
      include svdcmp.for
      include svbksb.for
Polyfitb.for



c   POLYFITB.FOR
c
c   version  01 Aug 89  SMZ
c
c   The guiding file should have the following format:
c
c   line 1: title
c   line 2: output file name. The curve fit coef end up in this file
c          in the form of data statements.
c   line 3: surface designation, appended to the results  6 characters.
c   line 4: nx,ny,nz the highest powers desired in the curve fits.
c   line 5: number of data files to be read.
c
      implicit real*8 (a-h,o-z)
      parameter (max_data=5000)
      real*8 limits(3,2),error
      real*8 x_data(3,max_data),y_data(max_data),choice
      real*8 coef((100,6),x(3,1000),w(1000,6),s(13)
c
      character*50 filename,outfile,guide,sfile,gfile
      character*80 title,author,data,facility,date
      character*5  surface
      character*2  force
      character*1  answer
c
      integer cons,var1,var2
      integer ia(100,6),ib(100,6),id(100,6),no_fcns(6),i_task
      common /pwrs / ia,ib,id,no_fcns,i_fcn
      external f3
      external sqr_err
c
 10  print *, '*****************************************'
     print *, '         MENU                           '
     print *, '*****************************************'
     print *, ' '
     print *, '*********************************************'
     print *, '1.  Read in data files'
     print *, '2.  Curve fit data'
     print *, '3.  Graph data and curve fit'
     print *, '4.  Evaluate the square error'
     print *, '5.  Write results to file'
     print *, '6.  Create graphing files'
     print *, '7.  Exit program'
     print *, '*********************************************'
     print *, ' '
     print *, 'Enter selection:'
     read *, choice
```

```
c
c
   if (choice.eq.1) then
     goto 20
   else if (choice.eq.2) then
     goto 200
   else if (choice.eq.3) then
     goto 10
   else if (choice.eq.4) then
     goto 675
   else if (choice.eq.5) then
     goto 725
   else if (choice.eq.6) then
     goto 625
   else if (choice.eq.7) then
     goto 750
   else
     goto 10
   endif
c
c The data files are in the form
c column no.  item description
c
c  1       item number for the data file.
c  2       angle of attack.
c  3       dynamic pressure
c  4       yawing moment coefficient
c  5       rolling moment coefficient
c  6       pressure coefficient
c  7       lift coefficient
c  8       drag coefficient
c  9       pitching moment coefficient
c  10      side force coefficient
c  11      yaw angle
c  12      not used  ---- 0.0
c  13      not used  ---- 0.0
c
c
c   the column identifiers for respective values
 20  ibeta=11
   ialpha=2
   ilift=7
   idrag=8
   iside=10
   iroll=5
   ipitch=9
   iyaw=4
c
c Enter the name of the file containing the information to guide the program
c through the dat input process.
c
   write(6,*)'Enter the guide file name.'
   read(5,31000) guide
   write(6,31000) guide
   open(15,file=guide,status='old')
c
c the main file contians:
c   1. title card to be included as a comment line in the data statements
c   2. nx,ny, and nz the orders of alpha,beta and delta fits. NOT USED
c   3. nfiles the number of file names to follow.
c   4. a list of file names containing the data for individual alpha sweeps.
c     one file name per line.
c
c The output is in the form of a data statement for each coefficient.
c
   read(15,10000) title
   write(6,*) title
   read(15,82000) outfile
   write(6,*) outfile
   read(15,83000) surface
   write(6,*) surface
   read(15,*) nx,ny,nz
   read(15,*) nfiles
   write(6,*) 'The number of files = ',nfiles
c
c  Initialize maximum values to zero.
c
c    open(12,file='data.all',status='unknown')
   npts=0
   k=0
c
   a_min=1.0e15
   b_min=1.0e15
   d_min=1.0e15
   a_max=-1.0e15
   b_max=-1.0e15
   d_max=-1.0e15
     do 100 i=1,nfiles
     read(15,90000,end=75,err=85) (filename(i1:i1),i1=1,19),delta
       write(6,30000) filename
       open(14,file=filename,status='old')
c
c  must add a control setting value to the beginning of each data file.
c
       do 50 j=1,60
         read(14,*,end=75,err=85) (s(k1),k1=1,13)
         k=k+1
         x(1,k)=s(ialpha)
         x(2,k)= (-1) * s(ibeta)
         x(3,k)=delta
         if(x(1,k).lt.a_min) a_min=x(1,k)
         if(x(2,k).lt.b_min) b_min=x(2,k)
         if(x(3,k).lt.d_min) d_min=x(3,k)
         if(x(1,k).gt.a_max) a_max=x(1,k)
         if(x(2,k).gt.b_max) b_max=x(2,k)
         if(x(3,k).gt.d_max) d_max=x(3,k)
         w(k,1)=s(ilift)
         w(k,2)=s(idrag)
         w(k,3)=s(iside)
         w(k,4)=s(ipitch)
         w(k,5)=s(iroll)
         w(k,6)=s(iyaw)
50      continue
75      close(14)
       goto 100
85      write(6,*)'Have had an error in reading ',filename
       close(14)
100   continue
     close(15)
     npts=k
c
c
     write(6,*) 'The data varied as follows:'
     write(6,*) a_min,' alpha ',a_max
     write(6,*) b_min,' beta ',b_max
     write(6,*) d_min,' delta ',d_max

c
     continue
     goto 10
200   write(6,*) 'Which case do you wish to work with?'
     write(6,*) '       '
     write(6,*) '1. Fixed'
     write(6,*) '2. Float'
     write(6,*) '3. Missing'
     read(5,*) icase
     if (icase.eq.1) then
       call fixzer (icase)
     else if (icase.eq.2) then
       call flozer (icase)
     else if (icase.eq.3) then
       call miszer (icase)
     else
       goto 200
     endif
c
c
c
     write(6,*)'Which force coefficient do you wish to curve fit?'
     write(6,*) '(Enter the corresponding number.)'
     write(6,*) '1 lift'
     write(6,*) '2 drag'
     write(6,*) '3 side force'
     write(6,*) '4 pitching moment'
     write(6,*) '5 rolling moment '
     write(6,*) '6 yawing moment'
     write(6,*) '7 or greater for listing the data'
     read(5,*) ifit
     if(ifit.gt.6) then
       write(6,*) 'Which data do you wish to examine?'
       write(6,*) '(Enter the corresponding number.)'
       write(6,*) '1 lift'
       write(6,*) '2 drag'
       write(6,*) '3 side force'
       write(6,*) '4 pitching moment'
       write(6,*) '5 rolling moment '
       write(6,*) '6 yawing moment'
       write(6,*) '7 alpha'
       write(6,*) '8 beta'
       write(6,*) '9 delta'
       read(5,*) iexam
       if(iexam.lt.6) then
         do 400 ijk=1,npts
           write(6,*) w(ijk,iexam)
```

```
400     continue
      else
        write(6,*) 'iexam-6=',iexam-6
        do 500 ijk=1,npts
          write(6,*) x(iexam-6,ijk)
 500    continue
       end if
     else
       i_fcn = ifit
c      call bldpwr(ifit)
 550   continue
       call lstsqr (f3,no_fcns(ifit),x,w(1,ifit),npts,coef(1,ifit))
       do 600 ijk=1,no_fcns(ifit)
         write(6,60000) ijk,ia(ijk,ifit),ib(ijk,ifit),id(ijk,ifit),
   x                coef(ijk,ifit)
 6?0   continue
     end if
     goto 10
c
c    GRAPHER AND SURFER FILE CREATION
c
c
c
c
 625 write(6,*)'Which do you want to hold constant?'
     write(6,*)' 1. alpha'
     write(6,*)' 2. beta'
     write(6,*)' 3. delta'
     read(5,*) cons
c
     write(6,*)'What is the first variable?'
     write(6,*)' 1. alpha'
     write(6,*)' 2. beta'
     write(6,*)' 3. delta'
     read(5,*) var1
c
     write(6,*)'What is the second variable?'
     write(6,*)' 1. alpha'
     write(6,*)' 2. beta'
     write(6,*)' 3. delta'
     read(5,*) var2
c
c    initialize limits matrix with variable ranges
c
     limits(1,1)=a_min
     limits(1,2)=a_max
     limits(2,1)=b_min
     limits(2,2)=b_max
     limits(3,1)=d_min
     limits(3,2)=d_max
c
c
 650 write(6,*)'The range on the constant is:'
     write(6,*)limits(cons,1),' constant ',limits(cons,2)
     write(6,*) 'Do you want to change it?'
     read(5,80000) answer
     if(answer.eq.'y' .or. answer.eq.'Y') then
       write(6,*)'What should the lower value be?'
       read(5,*) limits(cons,1)
       write(6,*)'What should the upper value be?'
       read(5,*) limits(cons,2)
       goto 650
     end if
     call find_pts(x,w(1,ifit),npts,x_data,y_data,ndata,limits,
   x          cons,var1,var2)
c
c    write the variable values, force or moment values, and
c    evaluated values to a data file for evaluation in 'grapher'
c
     write(6,*) 'Do you want to create a Grapher file?'
     read(5,80000) answer
     if(answer.eq.'y' .or. answer.eq.'Y') then
     write(6,*) 'Enter the file name with .dat:'
     read(5,31000) gfile
     open(10,file=gfile,status='unknown')
     do 665 i=1,ndata
     val1 = evl_pr(f3,coef(1,if...   _fcns(ifit),x_data(1,i))
     write(10,32000) x_data(var1,i),x_data(var2,i),y_data(i),
   x        val1,x_data(cons,i)
 665 continue
     close(10)
     end if
c
c    write data to a file for evaluation of contour plots in the
c    'surfer' software
c
     write(6,*) 'Do you want to create a Surfer file?'
     read(5,80000) answer
     if(answer.eq.'y' .or. answer.eq.'Y') then
     call gen_pctr(coef(1,ifit),limits,ifit,cons,var1,var2)
     end if
c
       goto 10
c
c
c
 675  error = sqr_err(x,w(1,ifit),npts,coef(1,ifit),ifit)
       write(6,*) '------------------------------------'
       write(6,*) 'The value of r squared is ',error
       write(6,*) '------------------------------------'
     write(6,*)'What do you want next?'
     write(6,*)' 1. Remove powers from the approximating functions'
     write(6,*)'    and refit.'
     write(6,*)' 2. Add powers'
     write(6,*)' 3. Quit'
     read(5,*) i_task
     if( i_task.eq.1) then
       write(6,*)' What is the smallest magnitude you wish to keep'
       write(6,*)' in the current fit?'
       read(5,*) bound
       call rm_coef(coef(1,ifit),bound,ifit)
       goto 550
     else if (i_task.eq.2) then
       call addcoef(coef(1,ifit),ifit)
       goto 550
     else
       goto 10
     end if
 725 call output(coef(1,ifit),ifit,error)
     goto 10
c
c
10000 format(a80)
20000 format(2x,e15.8,3(2x,i3),' coef, ia, ib, id ', i3)
30000 format(5x,'reading from ',a40)
31000 format(a50)
32000 format(5(1x,e15.7))
60000 format(' fcn no=',i3,' ia=',i3,' ib=',i3,' id=',i3,'   coef=',
   x     e12.5)
80000 format(a1)
82000 format(a40)
83000 format(a5)
89000 format(i4,' alpha=',f10.5,' beta=',f10.5,' delta=',f5.2,
   x      ' coef=',2f10.5)
90000 format(2x,19a1,3x,f5.2)
 750 STOP
     END
c
c  ****************************************************
c            FIXZER
c  ****************************************************
c
     subroutine fixzer(icase)
     integer ia(100,6),ib(100,6),id(100,6),no_fcns(6)
     common /pwrs /ia,ib,id,no_fcns,i_fcn
     integer alp, bet, del, comb, nofn
     real*8 zcoef(100,6),a(5), rsqr
     integer lift,drag,side,pitch,roll,yaw
     character * 10 force,control
c
c
     lift = 1
     drag = 2
     side = 3
     pitch = 4
     roll = 5
     yaw = 6
c
c    column identifiers
c
     ifcns = 1
     ialpha = 2
     ibeta = 3
     idelta = 4
     izcf = 5
c
c
      open(14,file='fixz1.dat',status='old')
      read(14,*) control
      write(6,*) control
      read(14,*) force
      write(6,*) force
      read(14,*) rsqr
      write(6,*) rsqr
```

```
      read(14,*) nofn
c
c
       do 15 j=1,60
         read(14,*,end=25,err=35) (s(k1),k1=1,5)
         ia(j,lift) = s(ialpha)
         ib(j,lift) = s(ibeta)
         id(j,lift) = s(idelta)
         zcoef(j,lift) = s(izcf)
         no_fcns(lift) = s(ifcns)
15     continue
25     close(14)
35     write(6,*)'Reading fizz1.dat'
      close(14)
c     do 40 j = 1,no_fcns(lift)
c     print *, ia(j,lift),ib(j,lift),id(j,lift),zcoef(j,lift)
40     continue

c
c
      open(14,file='fizz2.dat',status='old')
      read(14,*) control
      write(6,*) control
      read(14,*) force
      write(6,*) force
      read(14,*) rsqr
      write(6,*) rsqr
      read(14,*) nofn
c
c

       do 45 j=1,60
         read(14,*,end=50,err=51) (s(k1),k1=1,5)
         ia(j,drag) = s(ialpha)
         ib(j,drag) = s(ibeta)
         id(j,drag) = s(idelta)
         zcoef(j,drag) = s(izcf)
         no_fcns(drag) = s(ifcns)
45     continue
50     close(14)
51     write(6,*)'Reading fizz2.dat'
      close(14)
c     do 55 j = 1,no_fcns(drag)
c     print *, ia(j,drag),ib(j,drag),id(j,drag),zcoef(j,drag)
55     continue
c
c
      open(14,file='fizz3.dat',status='old')
      read(14,*) control
      write(6,*) control
      read(14,*) force
      write(6,*) force
      read(14,*) rsqr
      write(6,*) rsqr
      read(14,*) nofn
c
c

       do 65 j=1,60
         read(14,*,end=75,err=77) (s(k1),k1=1,5)
         ia(j,side) = s(ialpha)
         ib(j,side) = s(ibeta)
         id(j,side) = s(idelta)
         zcoef(j,side) = s(izcf)
         no_fcns(side) = s(ifcns)
65     continue
75     close(14)
77     write(6,*)'Reading fizz3.dat'
      close(14)
c     do 80 j = 1,no_fcns(side)
c     print *, ia(j,side),ib(j,side),id(j,side),zcoef(j,side)
80     continue
c
c
      open(14,file='fizz4.dat',status='old')
      read(14,*) control
      write(6,*) control
      read(14,*) force
      write(6,*) force
      read(14,*) rsqr
      write(6,*) rsqr
      read(14,*) nofn
c
c

       do 95 j=1,60
         read(14,*,end=100,err=105) (s(k1),k1=1,5)
         ia(j,pitch) = s(ialpha)
         ib(j,pitch) = s(ibeta)
         id(j,pitch) = s(idelta)
         zcoef(j,pitch) = s(izcf)
         no_fcns(pitch) = s(ifcns)
95     continue
100    close(14)
105    write(6,*)'Reading fizz4.dat'
      close(14)
c     do 110 j = 1,no_fcns(pitch)
c     print *, ia(j,pitch),ib(j,pitch),id(j,pitch),zcoef(j,pitch)
110    continue
c
c
      open(14,file='fizz5.dat',status='old')
      read(14,*) control
      write(6,*) control
      read(14,*) force
      write(6,*) force
      read(14,*) rsqr
      write(6,*) rsqr
      read(14,*) nofn
c
c
       do 115 j=1,60
         read(14,*,end=125,err=127) (s(k1),k1=1,5)
         ia(j,roll) = s(ialpha)
         ib(j,roll) = s(ibeta)
         id(j,roll) = s(idelta)
         zcoef(j,roll) = s(izcf)
         no_fcns(roll) = s(ifcns)
115    continue
125    close(14)
127    write(6,*)'Reading fizz5.dat'
      close(14)
c     do 130 j = 1,no_fcns(roll)
c     print *, ia(j,roll),ib(j,roll),id(j,roll),zcoef(j,roll)
130    continue
c
c
      open(14,file='fizz6.dat',status='old')
      read(14,*) control
      write(6,*) control
      read(14,*) force
      write(6,*) force
      read(14,*) rsqr
      write(6,*) rsqr
      read(14,*) nofn
c
c
       do 145 j=1,60
         read(14,*,end=150,err=155) (s(k1),k1=1,5)
         ia(j,yaw) = s(ialpha)
         ib(j,yaw) = s(ibeta)
         id(j,yaw) = s(idelta)
         zcoef(j,yaw) = s(izcf)
         no_fcns(yaw) = s(ifcns)
145    continue
150    close(14)
155    write(6,*)'Reading fizz6.dat'
      close(14)
c     do 160 j = 1,no_fcns(yaw)
c     print *, ia(j,yaw),ib(j,yaw),id(j,yaw),zcoef(j,yaw)
160    continue
      return
      end

c
c
c     ********************************************************
c              FLOZER
c     ********************************************************
c
      include flozer
c
c     ********************************************************
c              MISZER
c     ********************************************************
c
      include miszer
c
c
c     **********************************************************
c              LSTSQR
c     **********************************************************
c
      include lstsqr.for
```

```
c  *********************************************
c          SQR_ERR
c  *********************************************
c
   include sqr_err.for
c
c  *********************************************
c          F3
c  *********************************************
c
   include f3.for
c  *********************************************
c       BLDPWR
c  *********************************************
c
   include bldpwr
c
c  ****************************************************
c        EVLSQR
c  ****************************************************
   include evlsqr.for
c
c  ******************************************************
c        POLY
c  ******************************************************
c
   include poly.for
c
c  *****************************************************
c         RM_COEF
c  *****************************************************
   include rm_coef.for
c
c  *******************************************
c       ADDCOEF
c  *******************************************
c
   include addcoef.for
c
c  *********************************************
c       OUTPUT
c  *********************************************
c
   include output.for
c
c  *******************************************************
c          FIND_PTS
c  *******************************************************
c
   include find_pts.for
c
c  *****************************************************
c        GEN_PCTR
c  *****************************************************
c
   include gen_pctr
c
c  *********************************************
c     SVD_SOLVE
c  *********************************************
```

APPENDIX C

Introduction

Included in this appendix are the "polynomial equations" used to predict the aircraft control and stability derivatives for use in the trim analysis. Each set contains the following information. The data set on which the liest squares curve fit was accomplished to obtain the polynomial coefficients; i.e. zero, right horizontal tail etc.. The force or moment coefficient represented, the r squared value calculated in fitting the experimental data, and the number of terms in the polynomial. The columns of the data file contain the following values:

1. Number of the polynomial term
2. Power on the alpha term
3. Power on the beta term
4. Power on the delta term
5. Coefficient Associated with that term

Aircraft Stability Derivatives

```
zero
lift
0.998957786134833
8
 01  00  00  00 0.02425990
 02  00  01  00 0.01176668
 03  00  02  00 0.02273735
 04  00  03  00 -.00062970
 05  00  04  00 -.00059456
 06  01  00  00 0.07041496
 07  01  02  00 -.00001098
 08  02  00  00 -.00029655
zero
drag
0.999223969997176
12
 01  00  00  00 0.00989113
 02  00  01  00 0.00030617
 03  00  02  00 0.00082931
 04  00  03  00 -.00002382
 05  00  04  00 -.00002320
 06  01  00  00 -.00090749
 07  01  01  00 0.00017652
 08  01  02  00 0.00036501
 09  01  03  00 -.00001032
 10  01  04  00 -.00000963
 11  02  00  00 0.00114791
 12  02  01  00 0.00000162
zero
side
0.986993089975102
8
 01  00  00  00 .00000000
 02  00  01  00 -.01817564
 03  00  02  00 0.00011201
 04  00  03  00 -.00003593
 05  01  00  00 -.00007302
 06  02  00  00 0.00001572
 07  02  01  00 0.00000599
 08  03  00  00 -.00000196
zero
pitch
0.999419588860318
9
 01  00  00  00 0.00912623
 02  00  01  00 -.00372458
 03  00  02  00 -.00697840
 04  00  03  00 0.00019974
 05  00  04  00 0.00018126
 06  01  00  00 -.01944657
 07  01  01  00 -.00003131
 08  01  02  00 -.00001042
 09  02  00  00 -.00011202
zero
roll
0.971483734626692
12
```

```
01  00  00  00 0.00000000
02  00  01  00 -.00206500
03  00  02  00 0.00002188
04  00  03  00 0.00000592
05  01  00  00 0.00003762
06  01  01  00 -.00001006
07  01  02  00 0.00000007
08  01  03  00 0.00000037
09  02  00  00 0.00000083
10  02  01  00 0.00000072
11  02  02  00 -.00000005
12  02  03  00 0.00000003
zero
yaw
0.99450857443165
9
01  00  00  00 0.00000000
02  00  01  00 0.00598800
03  00  02  00 -.00005049
04  01  00  00 -.00008376
05  01  01  00 0.00006041
06  01  02  00 0.00000241
07  02  00  00 -.00000379
08  02  01  00 -.00000559
09  03  00  00 0.00000044
```

Aircraft Control Derivatives

```
lfl
lift
0.99710493885523
3
01  00  00  01 0.00808747
02  01  00  01 -.00007270
03  00  01  01 0.00016236
lfl
drag
0.99231474275174
3
01  00  00  01 0.00004459
02  01  00  01 0.00010485
03  00  01  01 0.00002147
lfl
side
0.99121786830928
3
01  00  00  01 0.00005379
02  01  00  01 0.00002327
03  00  01  01 -.00001908
lfl
pitch
0.99732533446253
3
01  00  00  01 -.00220590
02  01  00  01 -.00000686
03  00  01  01 -.00014414
lfl
roll
0.94671363142610
3
01  00  00  01 0.00124298
02  01  00  01 -.00001534
03  00  01  01 -.00000691
lfl
yaw
0.99067236168187
3
01  00  00  01 0.00011910
02  01  00  01 -.00002654
03  00  01  01 0.00000960
lht
lift
0.99875023115023
3
01  00  00  01 0.00524917
02  01  00  01 -.00001946
03  00  01  01 0.00007021
lht
drag
0.98700590428333
3
01  00  00  01 0.00023247
02  01  00  01 0.00014775
03  00  01  01 0.00001676
lht
side
0.99372908655258
3
01  00  00  01 -.00098652
02  01  00  01 0.00001164
03  00  01  01 -.00001998
lht
pitch
0.99720293330100
3
01  00  00  01 -.00712409
02  01  00  01 0.00000701
03  00  01  01 -.00009962
lht
roll
0.94875038542011
3
01  00  00  01 0.00052168
02  01  00  01 0.00000432
03  00  01  01 0.00000226
lht
yaw
0.98483418189834
3
01  00  00  01 0.00055888
02  01  00  01 -.00002129
03  00  01  01 0.00001092
lle
lift
0.99911018902596
3
01  00  00  01 -.00080409
02  01  00  01 0.00000181
03  00  01  01 -.00009354
lle
drag
0.99558264659623
3
01  00  00  01 0.00023239
02  01  00  01 0.00011567
03  00  01  01 0.00001087
lle
side
0.99268840957863
3
01  00  00  01 0.00003387
02  01  00  01 -.00004212
03  00  01  01 -.00003871
lle
pitch
0.99928094340029
3
01  00  00  01 -.00017172
02  01  00  01 -.00002018
03  00  01  01 0.00004164
lle
roll
0.94950568502171
3
01  00  00  01 -.00006718
02  01  00  01 0.00001683
03  00  01  01 -.00000630
lle
yaw
0.98693954647448
3
01  00  00  01 -.00002596
02  01  00  01 0.00001220
03  00  01  01 0.00001063
rfl
lift
0.99710493885523
3
01  00  00  01 0.00808747
02  01  00  01 -.00007270
03  00  01  01 -.00016236
rfl
drag
0.99231474275174
3
01  00  00  01 0.00004459
02  01  00  01 0.00010485
03  00  01  01 -.00002147
rfl
side
0.99121786830928
```

```
3
 01  00  00  01 -.00005379
 02  01  00  01 -.00002327
 03  00  01  01 -.00001908
rfl
pitch
0.99732533446253
3
 01  00  00  01 -.00220590
 02  01  00  01 -.00000686
 03  00  01  01 0.00014414
rfl
roll
0.94671363142610
3
 01  00  00  01 -.00124298
 02  01  00  01 0.00001534
 03  00  01  01 -.00000591
rfl
yaw
0.99067236168187
3
 01  00  00  01 -.00011910
 02  01  00  01 0.00002654
 03  00  01  01 0.00000969
rbt
lift
0.99875023115023
3
 01  00  00  01 0.00524917
 02  01  00  01 -.00001946
 03  00  01  01 -.0000[illegible]21
rbt
drag
0.98700590428333
3
 01  00  00  01 0.00023247
 02  01  00  01 0.00014775
 03  00  01  01 -.00001676
rbt
side
0.99372908655258
3
 01  00  00  01 0.00098652
 02  01  00  01 -.00001164
 03  00  01  01 -.00001998
rbt
pitch
0.99720293330100
3
 01  00  00  01 -.00712409
 02  01  00  01 0.00000701
 03  00  01  01 0.00009962
rbt
roll
0.94875038542011
3
 01  00  00  01 -.00052168
 02  01  00  01 -.00000432
 03  00  01  01 0.00000226
rbt
yaw
0.98483418189834
3
 01  00  00  01 -.00065888
 02  01  00  01 0.00002129
 03  00  01  01 0.00001092
rle
lift
0.99942437477478
3
 01  00  00  01 -.00092112
 02  01  00  01 0.00085398
 03  00  01  01 0.00006637
rle
drag
0.99902917949565
3
 01  00  00  01 0.00066338
 02  01  00  01 -.00009976
 03  00  01  01 0.00002005
rle
side
0.99129740157611
3
 01  00  00  01 -.00030801
 02  01  00  01 0.00007528
 03  00  01  01 -.00002878
rle
pitch
0.99947325104706
3
 01  00  00  01 -.00029839
 02  01  00  01 -.00001594
 03  00  01  01 -.00004731
rle
roll
0.92814744695374
3
 01  00  00  01 0.00012548
 02  01  00  01 -.00001585
 03  00  01  01 -.00000511
rle
yaw
0.98461215072914
3
 01  00  00  01 0.00017089
 02  01  00  01 -.00003[illegible]02
 03  00  01  01 0.00001108
rud
lift
0.99903585130353
3
 01  00  00  01 -.00003361
 02  01  00  01 0.00000247
 03  00  01  01 0.00000177
rud
drag
0.97920031383825
3
 01  00  00  01 0.00022245
 02  01  00  01 -.00001533
 03  00  01  01 0.00002796
rud
side
0.99435063831380
3
 01  00  00  01 0.00334111
 02  01  00  01 0.00000440
 03  00  01  01 0.00000386
rud
pitch
0.99715538778147
3
 01  00  00  01 0.00000413
 02  01  00  01 -.00000019
 03  00  01  01 0.00003014
rud
roll
0.98420339634860
3
 01  00  00  01 0.00053189
 02  01  00  01 -.00004126
 03  00  01  01 0.00000001
rud
yaw
0.99464111708521
3
 01  00  00  01 -.00204024
 02  01  00  01 -.00000473
 03  00  01  01 -.00000257
```

APPENDIX D
DEVELOPMENT OF EQUATIONS FOR TRIM SURVEYS

Introduction

In the quest to gain insight into the nature of the stability characteristics of an impaired aircraft it is necessary to derive the equations which will describe a state of equilibrium for the aircraft in flight. The derivation of these equilibrium, or trim, equations will follow the more detailed discussion found in [6:203-233]. In this chapter the nonlinear equilibrium equations for an aircraft in rectilinear flight will be derived. A functional relationship for describing the aircraft pitch angle in terms of angle of attack, roll angle and side-slip angle is also derived for use in Chapter IV of the thesis.

Derivation of Equilibrium Equations

The following assumptions are stated at the beginning of the derivation of the aircraft equations of motion and will be re-referenced at appropriate points in the derivation.

1. The aircraft is assumed to be a rigid airframe.
2. The earth is assumed to be an inertial frame of reference.
3. The aircraft mass and mass distribution are assumed to be constant.
4. The X-Z plane of the aircraft is assumed to be a plane of inertial symmetry.

Four orthogonal right handed coordinate systems are defined so that the location, orientation, and motion of the aircraft may be conveniently described. The aerodynamic forces and moments will also be referenced in these axis systems.

Earth Fixed: The earth fixed frame is rigidly attached to the earth and is oriented so that the Z axis is collinear with the gravitational acceleration vector. In light of assumption number 2 this frame is considered to be an inertial coordinate system.

Body: The Body frame is one of three body fixed frames which are defined such that their origins are rigidly attached to the center of gravity of the aircraft. The Body frame is oriented so that the X axis proceeds positively out the nose of the aircraft. The Y axis is defined to be positive out the right wing of the aircraft and the Z axis is located normal to X-Y plane.

Stability : The Stability axis system is also a body axis system which is rigidly attached to the aircraft center of gravity. The Stability axis system is defined by rotating the Body axis system about the Body Y axis until the stability x axis, Xs, is collinear with the projection of the velocity vector on the X-Z plane of symmetry. The Stability axes are denoted with capital letters subscripted with a small s. A pictorial representation of the Body and Stability axis systems is shown in Figure 32

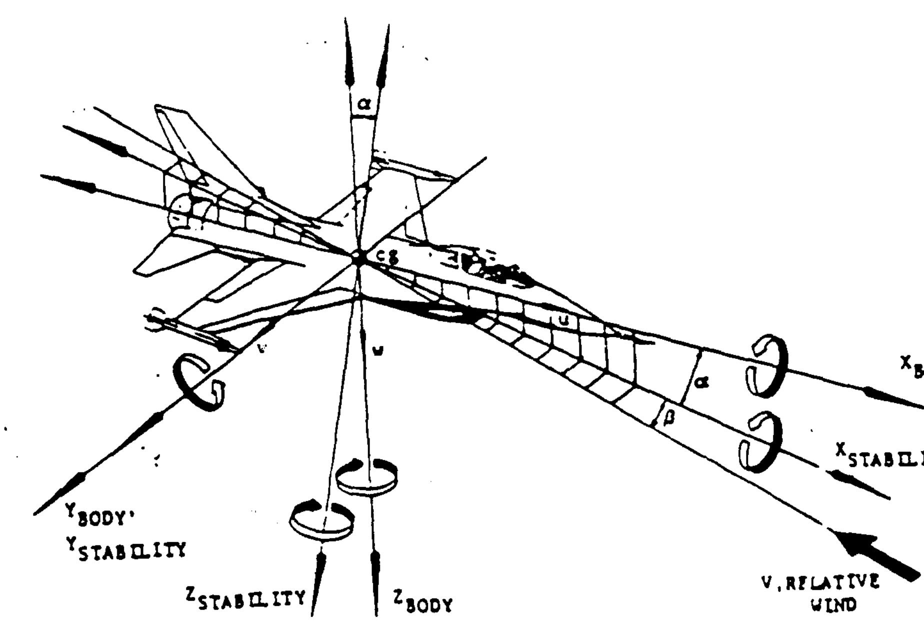


Figure 32 Body and Stability Axis Systems

Wind: The last body axis system defined is the Wind axis system. It is defined by rotating the Stability axis system about the Zs axis until the x axis, Xw, is collinear with the free stream velocity vector V. The Wind axis are denoted with capital letters subscripted with a small w.

Given a rigid body, Assumption 1, its position and orientation in space can be completely described with six coordinates. For this reason, aircraft are often referred to as six degree of freedom systems. For aircraft motion studies it is usually most desirable to work with a reference frame which is rigidly attached to the aircraft. The Body axis system is therefore selected as the coordinate frame in which the derivation of the aircraft equations of motion will be accomplished. The aircraft rectilinear velocity vector $\overline{V}$ and angular velocity vector $\overline{\Omega}$ are defined in the Body axis system as:

$$\overline{V} = U\hat{i} + V\hat{j} + W\hat{k} \tag{D.1}$$

$$\overline{\Omega} = P\hat{i} + Q\hat{j} + R\hat{k} \tag{D.2}$$

With these quantities defined the linear and angular momentum vectors of the aircraft are defined as:

$$\overline{P} = m\overline{V} \tag{D.3}$$

$$\overline{H} = \underline{I} \cdot \overline{\Omega} \tag{D.4}$$

$\underline{I}$ is the inertia dyad and for most aircraft it is a symmetric matrix of the following form:

$$\underline{I} = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix} \tag{D.5}$$

The definition of the individual elements of this matrix may be found in [6:209-215]. Assumption 3 implies that the mass in the linear momentum equation and the inertia dyad will not vary with time and may therefore be regarded as constants.

Application of Newton's Second Law to the aircraft indicates that the time rate of change of linear momentum is proportional to the sum of the externally applied forces.

$$\Sigma \; \overline{F}_{ext} = \frac{d\overline{P}}{dt} \tag{D.6}$$

In an analogous fashion, the inertial time rate of change of the angular momentum is proportional to the sum of the applied moments about the center of mass.

$$\Sigma \; \overline{M}_{ext} = \frac{d\overline{H}}{dt} \tag{D.7}$$

Note that Newton's Laws must be applied in an inertial reference frame. The aircraft Body axis system in general will be rotating and accelerating relative to the earth and therefore does not qualify as an inertial frame of reference. For this reason, it is necessary to form the stated time derivatives in an equation which relates the aircraft frame of reference to one which is inertial. As stated in Assumption 2 the earth will be considered to be an inertial frame. The time rate of change of the linear momentum in the Body axis system is then [6:211]:

$$\frac{d\overline{P}}{dt} = \dot{\overline{P}} + \overline{\Omega} \times \overline{P} \tag{D.8}$$

$$\frac{d\overline{P}}{dt} = m \left\{ \dot{u}\hat{i} + \dot{v}\hat{j} + \dot{w}\hat{k} + (QW-RV)\hat{i} + (RU-PW)\hat{j} + (PV-QU)\hat{k} \right\} \tag{D.9}$$

And equating equation (D.9) to the sum of the externally applied forces yields:

$$\Sigma F_{ext} = m \left\{ (\dot{u}+QW-RV)\hat{i} + (\dot{v}+RU-PW)\hat{j} + (\dot{w}+PV-QU)\hat{k} \right\} \tag{D.10}$$

The time rate of change of the angular momentum in the Body axis system is given as

$$\frac{d\bar{H}}{dt} = \dot{\bar{H}} + \bar{\Omega} \times \bar{H} \tag{D.11}$$

$$= \frac{d(\underline{I} \cdot \bar{\Omega})}{dt} + \bar{\Omega} \times \underline{I} \cdot \bar{\Omega} \tag{D.12}$$

The expression for the dot product of the inertia dyad can be expanded to give:

$$\begin{aligned} \underline{I} \cdot \bar{\Omega} &= (PI_{xx} - QI_{xy} - RI_{xz})\hat{i} \\ &+ (-PI_{yx} + QI_{yy} - RI_{yz})\hat{j} \\ &+ (-PI_{zx} - QI_{zy} + RI_{zz})\hat{k} \end{aligned} \tag{D.13}$$

Applying Assumption 4 implies that $I_{yz} = 0$ and that $I_{xz} = 0$. Making these simplifying substitutions, taking the time derivative and substituting back into equation (D.12) yields:

$$\begin{aligned} \frac{d\bar{H}}{dt} &= (\dot{P}I_{xx} - \dot{R}I_{xx})\hat{i} + \dot{Q}I_{yy}\,\hat{j} + (-\dot{P}I_{xz} + \dot{R}I_{zz})\hat{k} \\ &+ \bar{\Omega} \times \left\{ (PI_{xx} - RI_{xz})\hat{i} + QI_{yy}\,\hat{j} + (PI_{xz} + RI_{zz})\hat{k} \right\} \end{aligned} \tag{D.14}$$

When the cross product is performed, the equation may be split into three separate scalar equations; one equation for each of the coordinate directions.

$$\frac{dHx}{dt} = \dot{P}I_{xx} - \dot{R}I_{xz} - Q\,PI_{xz} + QR\,(I_{zz} - I_{yy}) \tag{D.15}$$

$$\frac{dHy}{dt} = \dot{Q}I_{yy} + P^2I_{xz} + RP(I_{xx} - I_{zz}) - R^2I_{xz} \tag{D.16}$$

$$\frac{dHz}{dt} = \dot{R}I_{zz} - \dot{P}I_{xz} + PQ(I_{yy} - I_{xx}) + QR\,I_{xz} \tag{D.17}$$

Equations (D.6) and (D.7) may now be expressed in their component form to issue the six aircraft equations of motion in the aircraft Body axis.

$$\Sigma F_x = m\ (\dot{U} + QW - RV) \tag{D.18}$$

$$\Sigma F_y = m\ (\dot{V} + RU - PW) \tag{D.19}$$

$$\Sigma F_z = m\ (\dot{W} + PV - QU) \tag{D.20}$$

$$\Sigma M_x = \dot{P}I_{xx} - \dot{R}I_{xz} - QPI_{xz} + QR\,(I_{zz} - I_{yy}) \tag{D.21}$$

$$\Sigma M_y = \dot{Q}I_{yy} + P^2I_{xz} + RP\,(I_{xx} - I_{zz}) - R^2I_{xz} \tag{D.22}$$

$$\sum M_z = \dot{R} I_{zz} - \dot{P} I_{xz} + PQ\,(I_{yy} - I_{xx}) + QR I_{xz} \qquad \text{(D.23)}$$

The forces and moments which are applied to the aircraft and are represented on the left hand side of the above equations will be developed by the aerodynamic characteristics of the aircraft and the thrust of the engine. Also included in the force equations will be the force exerted on the aircraft by gravity.Since the gravitational vector is defined in the Earth Fixed reference frame it is necessary to define a method by which the gravitational vector may be expressed in the Body frame. A transformation matrix may be defined in terms of the three Euler angles; Ψ, θ, and ϕ. Ψ is defined as the aircraft heading angle, θ the pitch angle and ϕ the roll angle. The transformation matrix between the Earth frame and the Body frame, called [BV], is rather cumbersome but since it will be needed at a later point in the derivation it is defined now.

$$[BV] = \begin{bmatrix} \cos\psi \cos\theta & \sin\psi \cos\theta & -\sin\theta \\ \cos\psi \sin\theta \sin\phi - \sin\psi \cos\phi & \sin\psi \sin\theta \sin\phi + \cos\psi \cos\phi & \cos\theta \sin\phi \\ \cos\psi \sin\theta \cos\phi + \sin\psi \sin\phi & \sin\psi \sin\theta \cos\phi - \cos\psi \sin\phi & \cos\theta \cos\phi \end{bmatrix} \qquad \text{(D.24)}$$

Transforming the gravity vector into the Body axis system by premultiplying by [BV] provides the gravity force to be applied in each of the aircraft force equations:

$$m\bar{g} = mg\,(-\sin\theta\ \hat{i} + \cos\theta\ \sin\phi\ \hat{j} + \cos\theta\ \cos\phi\ \hat{k}) \tag{D.25}$$

Since the investigations conducted in this thesis will be concerned with the aircraft in an equilibrium state, the equations of motion are further simplified by setting all of the acceleration terms to zero. The resulting equations are the equations which describe an aircraft in a state of equilibrium or trim.

$$F_{A_x} + F_{T_x} - mg\ \sin\theta = m\ (QW - RV) \tag{D.26}$$

$$F_{A_Y} + F_{T_Y} + mg\ \cos\theta\ \sin\phi = m\ (RU - PW) \tag{D.27}$$

$$F_{A_z} + F_{T_Z} + mg\ \cos\theta\ \cos\phi = m\ (PV - QU) \tag{D.28}$$

$$M_{A_X} = QR\ (I_{zz} - I_{yy}) - QPI_{xz} \tag{D.29}$$

$$M_{A_Y} = P^2 I_{xz} + PR\ (I_{xx} - I_{zz}) - R^2 I_{xz} \tag{D.30}$$

$$M_{A_Z} = PQ\ (I_{yy} - I_{xx}) + QRI_{xz} \tag{D.31}$$

The A subscripts indicate an aerodynamic force or moment. A_T represents a force component generated by the aircraft engine. It is assumed from this point forward that the thrust vector of the engine is aligned with the Body X axis and that therefore the Z and Y components due to thrust are zero.

Several steady state flight conditions can be described with these equations, [10,37-39]. For rectilinear flight all of the angular rates are zero. In steady turning flight the heading angle changes at a constant rate. The third steady condition is that of a steady, symmetrical pull-up which is characterized by:

$$V = P = R = 0$$

and the wings level or ϕ equal to zero. The studies conducted in this thesis are concerned with rectilinear flight and so equations (D.26) - (D.31) may be further simplified into the form in which they are applied in Chapter IV.

$$F_{A_X} + F_{T_X} - mg \sin\theta = 0 \tag{D.32}$$

$$F_{A_Y} + mg \cos\theta \cos\phi = 0 \tag{D.33}$$

$$F_{A_Z} + mg \cos\theta \cos\phi = 0 \tag{D.34}$$

$$M_{A_X} = 0 \tag{D.35}$$

$$M_{A_Y} = 0 \tag{D.36}$$

$$M_{A_Z} = 0 \tag{D.37}$$

Derivation of Flight Path / Pitch Angle Relationship

Because of the form chosen to model the aerodynamic forces and moments, equations (D.32)-(D.34) contain not only trigonometric functions but are also nonlinear in α and β. For this reason these equations may not be solved with conventional linear analysis techniques and will require some other method of solution. This technique will be developed in chapter IV. The technique will require, at one point, a functional description of the aircraft pitch angle which holds the aircraft flight path angle at zero. This function will now be derived

The flight path angle, γ, will be defined as the angle, in a vertical plane, that the aircraft velocity vector forms with the local horizontal. For many flight analyses where small angles are assumed the relationship between the flight path angle and the pitch angle may be expressed as

$$\gamma = \theta - \alpha \tag{D.38}$$

In general however this relationship does not hold since the aircraft is allowed to take on significant values of roll angle. For this reason it is necessary to derive an expression for the pitch angle in terms of α, β, and ϕ for γ equal to zero. To begin the derivation, two sets of Euler angles are defined. The first set

$$\psi \quad \theta \quad \phi \tag{D.39}$$

locate the aircraft Body axis with respect to the inertial Earth fixed frame. The second set

$$\psi_\omega = 0 \tag{D.40}$$

$$\theta_\omega = 0 \tag{D.41}$$

$$\phi_\omega = \phi_\omega \tag{D.42}$$

are used to specify the Wind axis relative to the Earth frame. Equation (D.40) indicates that a specified heading has been selected and equation (D. 41) represents the flight path angle equal to zero condition. An arbitrary rotation of the aircraft about its velocity vector is indicated by equation (D.42).

Equation (D.24) represented the transformation matrix between the Earth fixed frame and the Body fixed frame. Since this matrix is an orthonormal matrix, [2,116] the following relationships apply:

$$[L]^{-1} = [L]^{T} \tag{D.43}$$

$$[VB] = [BV]^{T} \tag{D.44}$$

$$[VB] = \begin{bmatrix} \cos\theta\ \cos\phi & \begin{matrix}\sin\phi\ \sin\theta\ \cos\psi \\ -\cos\phi\ \sin\psi\end{matrix} & \begin{matrix}\cos\phi\ \sin\theta\ \cos\psi \\ +\sin\phi\ \sin\psi\end{matrix} \\ \cos\theta\ \sin\psi & \begin{matrix}\sin\phi\ \sin\theta\ \sin\psi \\ +\cos\phi\ \cos\psi\end{matrix} & \begin{matrix}\cos\phi\ \sin\theta\ \sin\psi \\ -\sin\phi\ \cos\psi\end{matrix} \\ -\sin\theta & \sin\phi\ \cos\theta & \cos\phi\ \cos\theta \end{bmatrix} \tag{D.45}$$

Recognizing that the wind axis system is also a body fixed system and substituting the defined Euler angles to the Wind axis into equation (D.45) yields

$$[VW] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi_\omega & -\sin\phi_\omega \\ 0 & \sin\phi_\omega & \cos\phi_\omega \end{bmatrix} \tag{D.46}$$

A transformation matrix may then be obtained from the Wind axis to the Body axis system in terms of the defined set of six Euler angles. This matrix is obtained by postmultiplying equation (D.24) by equation (D.46).

$$[BW] = [BV] * [VW] \tag{D.47}$$

(D.48)

$$[BW] = \begin{bmatrix} \cos\theta \cos\psi & \cos\phi_\omega \cos\theta \sin\psi - \sin\theta \cos\phi_\omega & -\sin\phi_\omega \cos\theta \sin\psi - \sin\theta \cos\phi_\omega \\ \sin\phi \sin\theta \cos\psi - \cos\phi \sin\psi & \cos\phi_\omega (\sin\phi \sin\theta \sin\psi + \cos\phi \cos\psi) + \sin\phi_\omega \sin\phi \cos\theta & -\sin\phi_\omega (\sin\phi \sin\theta \sin\psi + \cos\phi \cos\psi) + \cos\phi_\omega \sin\phi \cos\theta \\ \cos\phi \sin\theta \cos\psi + \sin\phi \sin\psi & \cos\phi_\omega (\cos\phi \sin\theta \sin\psi - \sin\phi \cos\psi) + \sin\phi_\omega \cos\phi \cos\theta & -\sin\phi_\omega (\cos\phi \sin\theta \sin\psi - \sin\phi \cos\psi) + \cos\phi_\omega \cos\phi \cos\theta \end{bmatrix}$$

The transformation matrix between the Wind and Body axis systems may also be expressed in terms of α and β as:

$$[BW] = \begin{bmatrix} \cos\alpha\ \cos\beta & -\cos\alpha\ \sin\beta & -\sin\alpha \\ \sin\beta & \cos\beta & 0 \\ \sin\alpha\ \cos\beta & -\sin\alpha\ \sin\beta & \cos\alpha \end{bmatrix} \tag{D.49}$$

Equating equation (D.48) with equation (D.49) provides the equations needed to obtain the desired expression for θ. Setting the first column of each matrix equal to one another yields the three equations

$$\cos\theta\ \cos\psi = \cos\alpha\ \cos\beta \tag{D.50}$$

$$\cos\phi\ \sin\theta\ \cos\psi + \sin\phi\ \sin\psi = \sin\alpha\ \cos\beta \tag{D.51}$$

$$\sin\phi\ \cos\psi\ \sin\theta - \cos\phi\ \sin\psi = \sin\beta \tag{D.52}$$

Equation (D.51) is divided by $\sin(\phi)$ and equation (D.52) by $\cos(\phi)$ to produce

$$\frac{\cos\phi}{\sin\phi} \sin\theta\ \cos\psi + \sin\psi = \frac{\sin\alpha\ \cos\beta}{\sin\phi} \tag{D.53}$$

$$\frac{\sin\phi}{\cos\phi} \sin\theta\ \cos\psi - \sin\psi = \frac{\sin\beta}{\cos\phi} \tag{D.54}$$

Adding equations (D.53) and (D.54) gives:

$$\frac{\cos\phi \ \sin\theta \ \cos\psi}{\sin\phi} + \frac{\sin\phi \ \sin\theta \ \cos\psi}{\cos\phi} = \frac{\sin\alpha \ \cos\beta}{\sin\phi} + \frac{\sin\beta}{\cos\phi} \tag{D.55}$$

$$\sin\theta \ \cos\psi \left\{ \frac{\cos^2\phi}{\sin\phi \ \cos\phi} + \frac{\sin^2\phi}{\sin\phi \ \cos\phi} \right\} = \frac{\sin\alpha \ \cos\beta}{\sin\phi} + \frac{\sin\beta}{\cos\phi} \tag{D.56}$$

$$\frac{\sin\theta \ \cos\psi}{\sin\phi \ \cos\phi} = \frac{\sin\alpha \ \cos\beta}{\sin\phi} + \frac{\sin\beta}{\cos\phi} \tag{D.57}$$

$$\sin\theta \ \cos\psi = \sin\alpha \ \cos\beta \ \cos\phi + \sin\beta \ \sin\phi \tag{D.58}$$

Equation (D.50) provides the relationship that

$$\cos\psi = \frac{\cos\alpha \ \cos\beta}{\cos\theta} \tag{D.59}$$

which can then be substituted into equation (D.58) to provide

$$\frac{\sin\theta}{\cos\theta} \cos\alpha \ \cos\beta = \sin\alpha \ \cos\beta \ \cos\phi + \sin\beta \ \sin\phi \tag{D.60}$$

The desired pitch angle, to hold the flight path angle equal to zero, in terms of α, β, and ϕ is then

$$\theta = \mathrm{Tan}^{-1}\left\{ \mathrm{Tan}\alpha \; \cos\phi + \frac{\mathrm{Tan}\beta}{\cos\alpha} \; \sin\phi \right\} \tag{D.61}$$

APPENDIX E AERODYNAMIC COEFFICIENTS

Normalized Control Derivatives

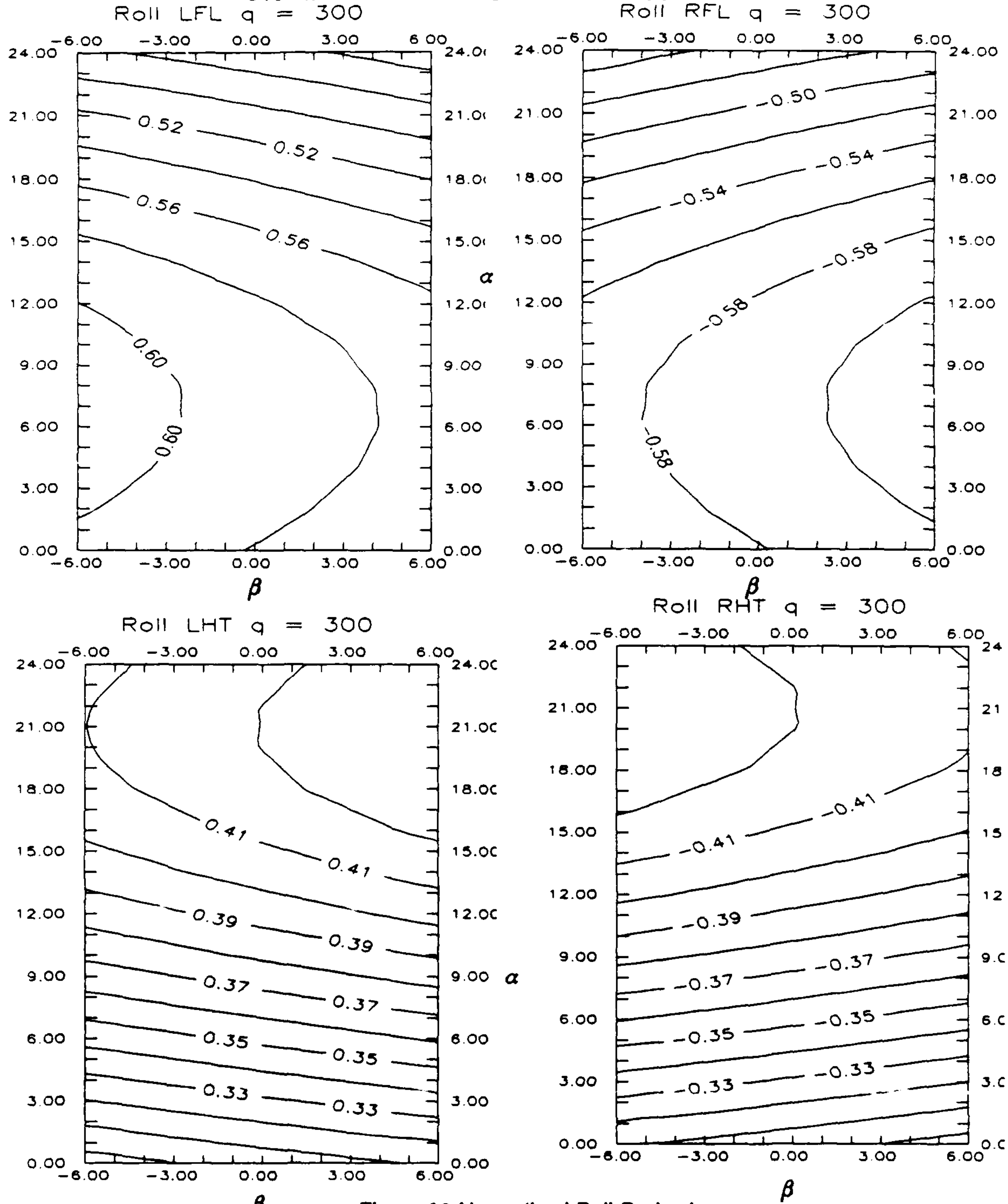


Figure 33 Normalized Roll Derivatives

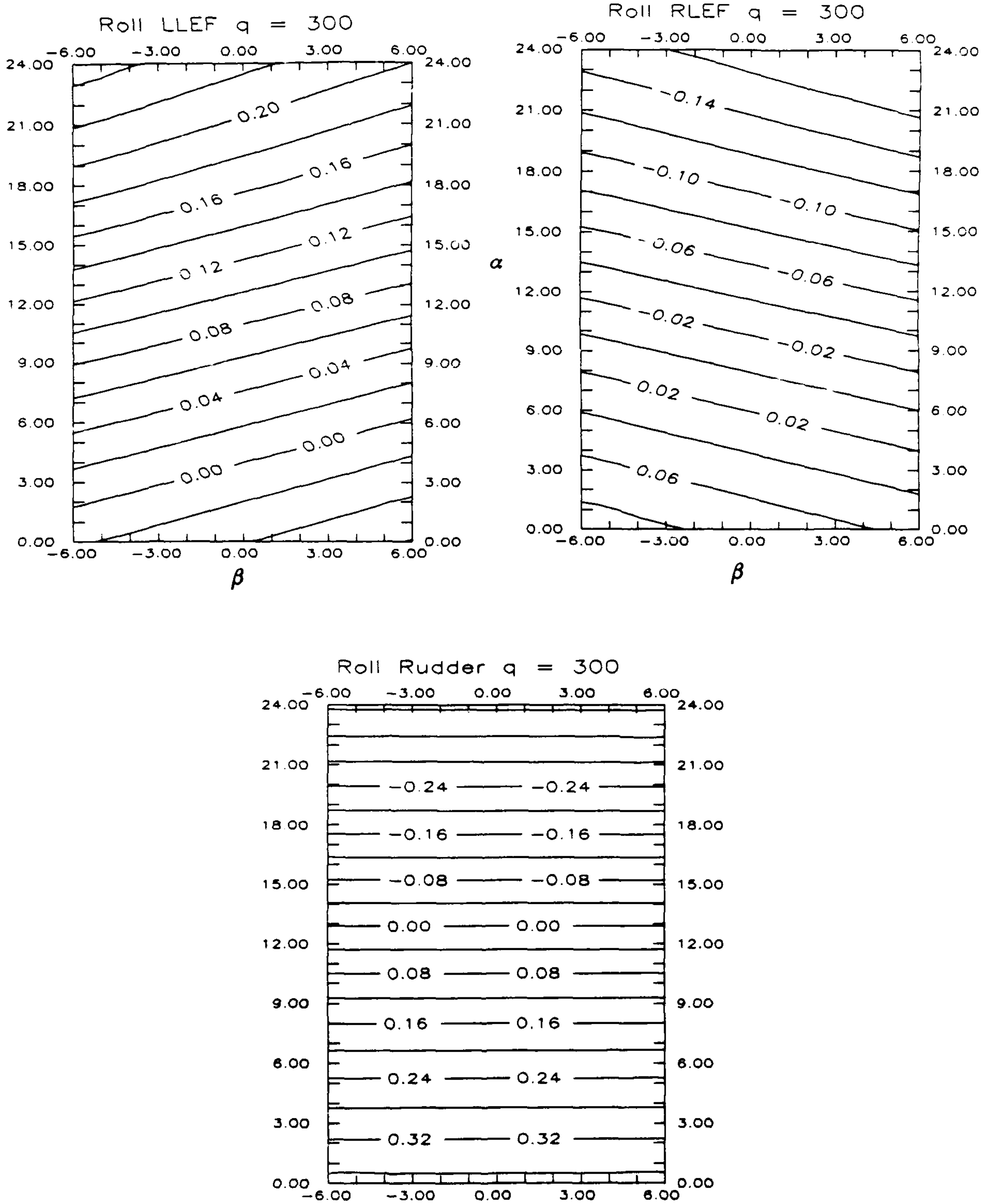


Figure 34 Normalized Roll Derivatives

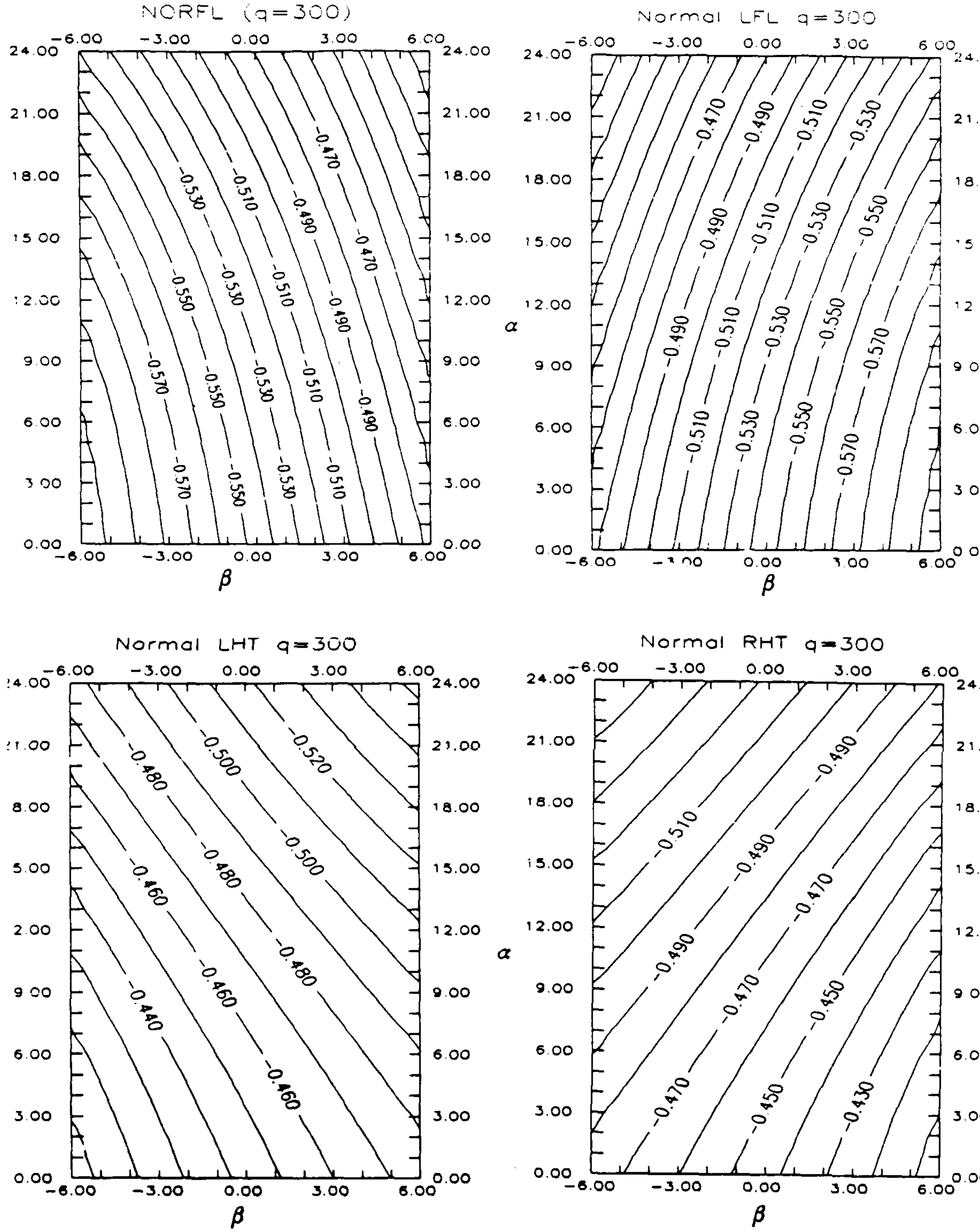


Figure 35 Normalized Control Derivatives

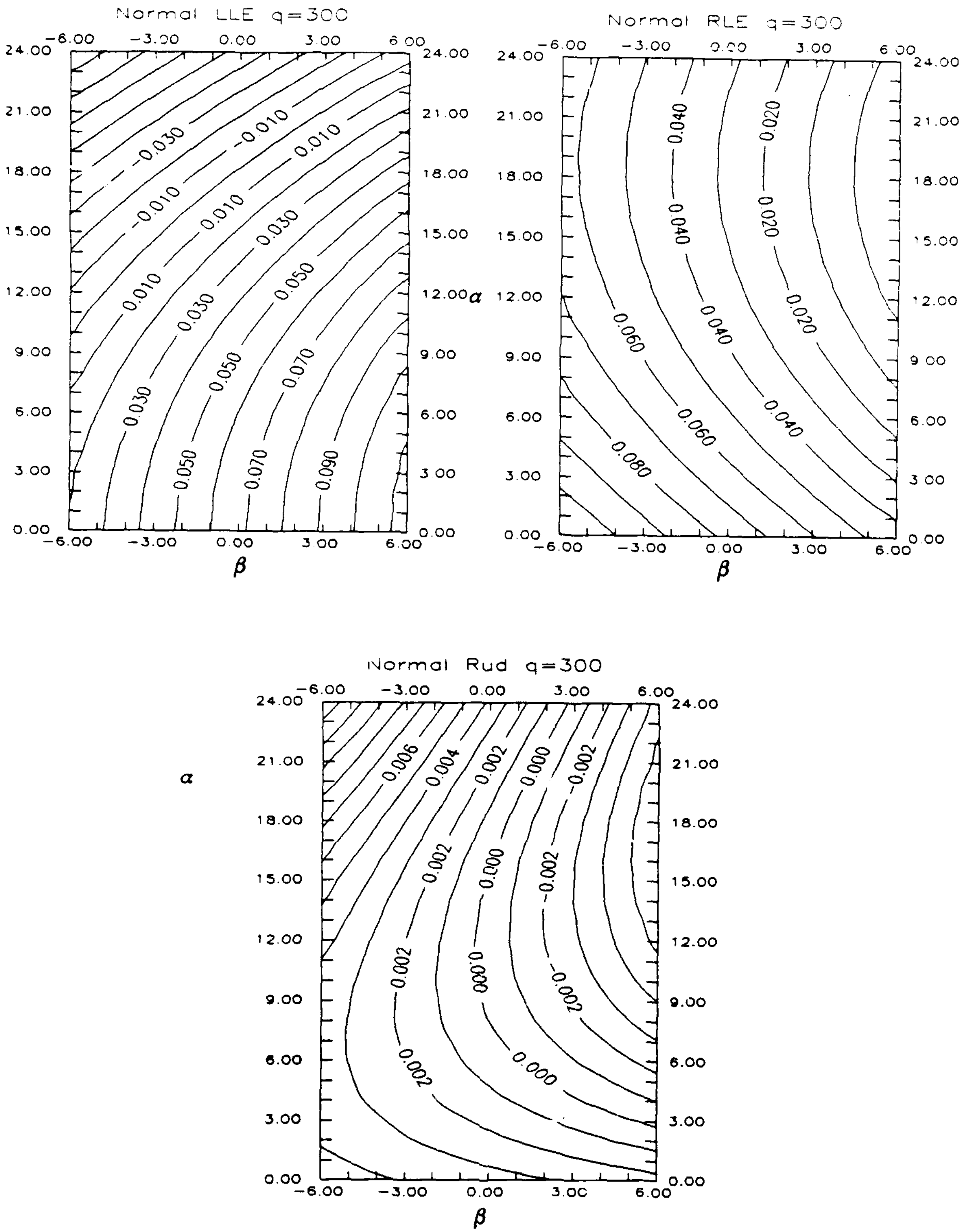


Figure 36 Normalized Control Derivatives

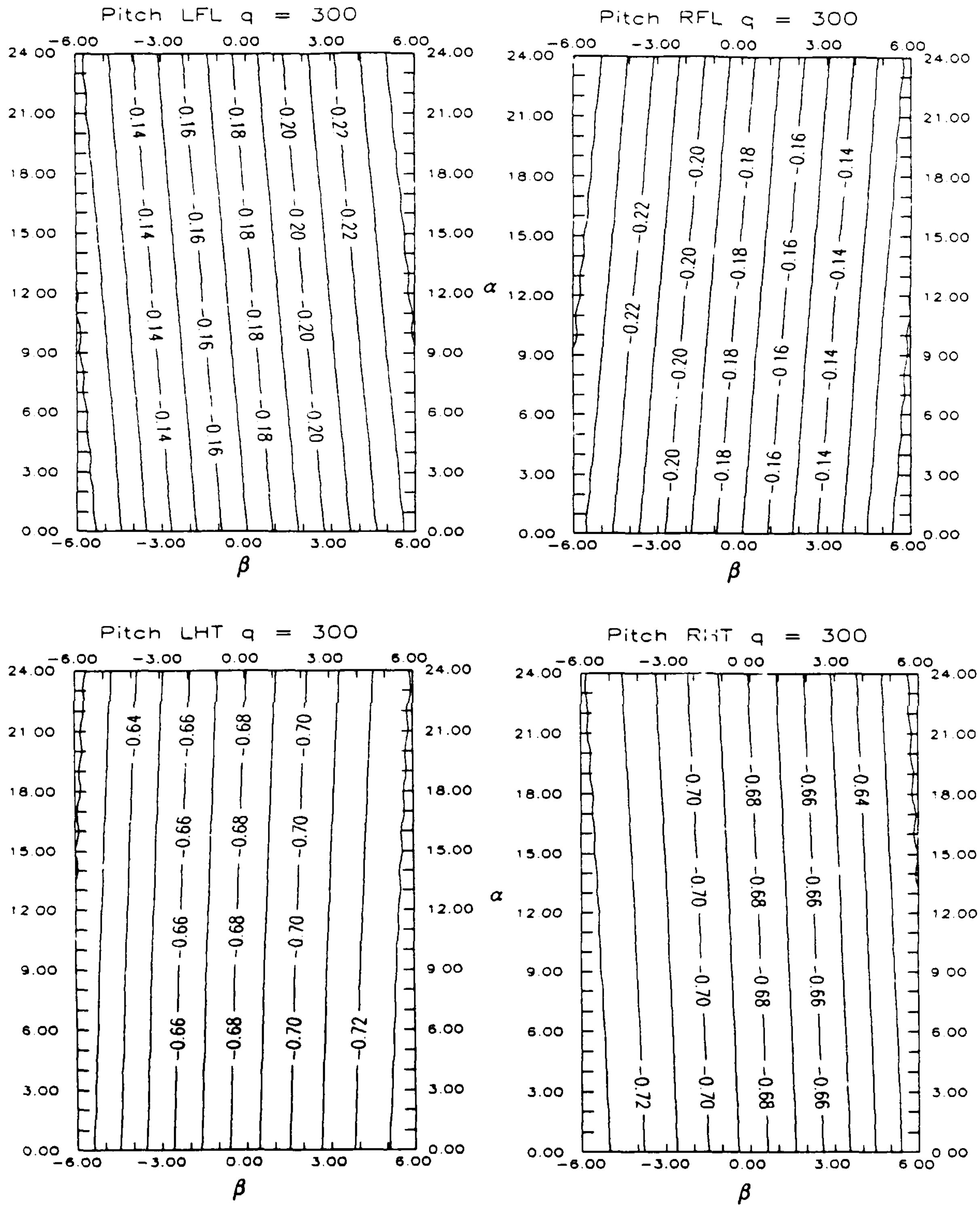


Figure 37 Normalized Pitch Derivatives

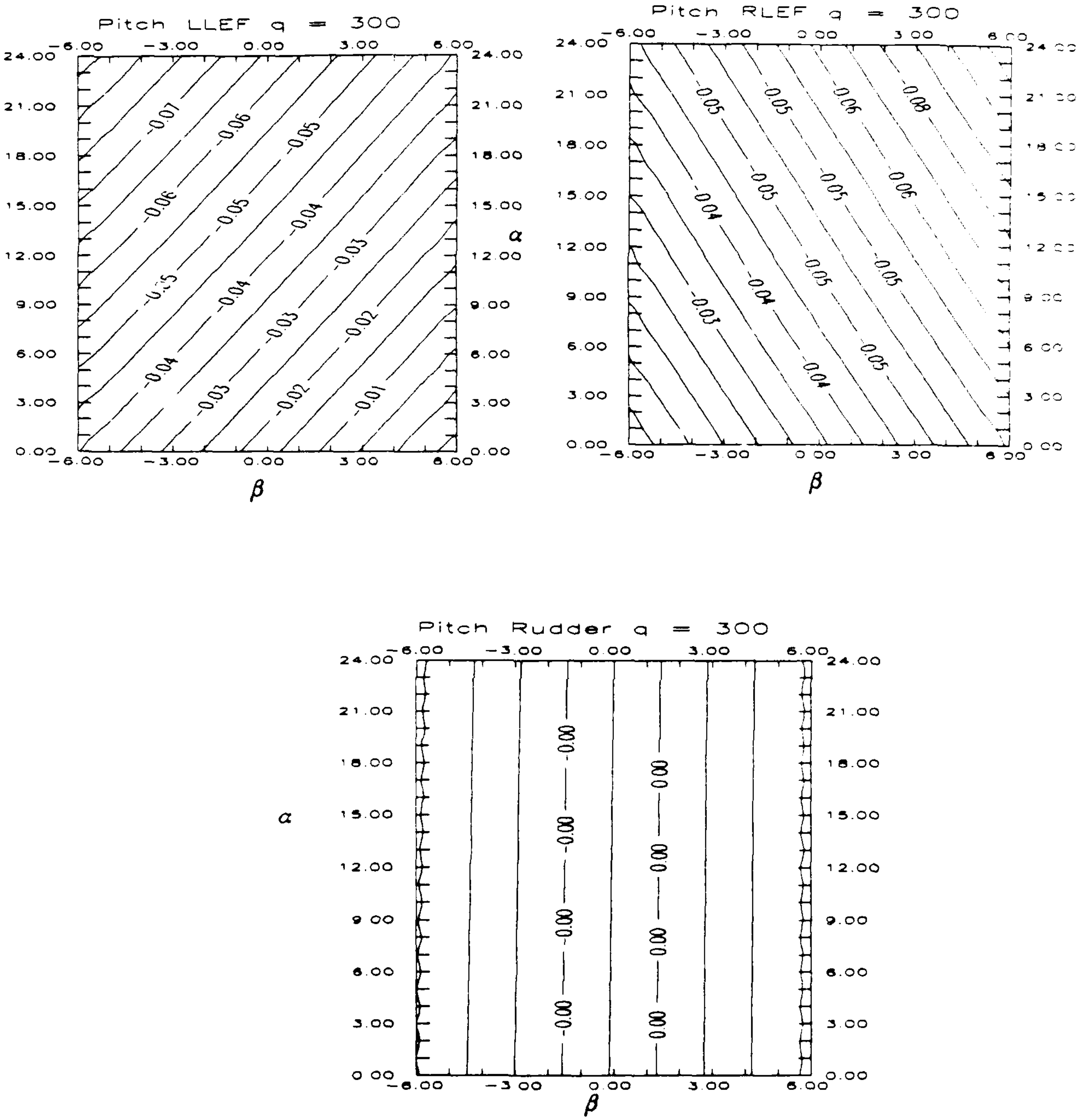


Figure 38 Normalized Pitch Derivatives

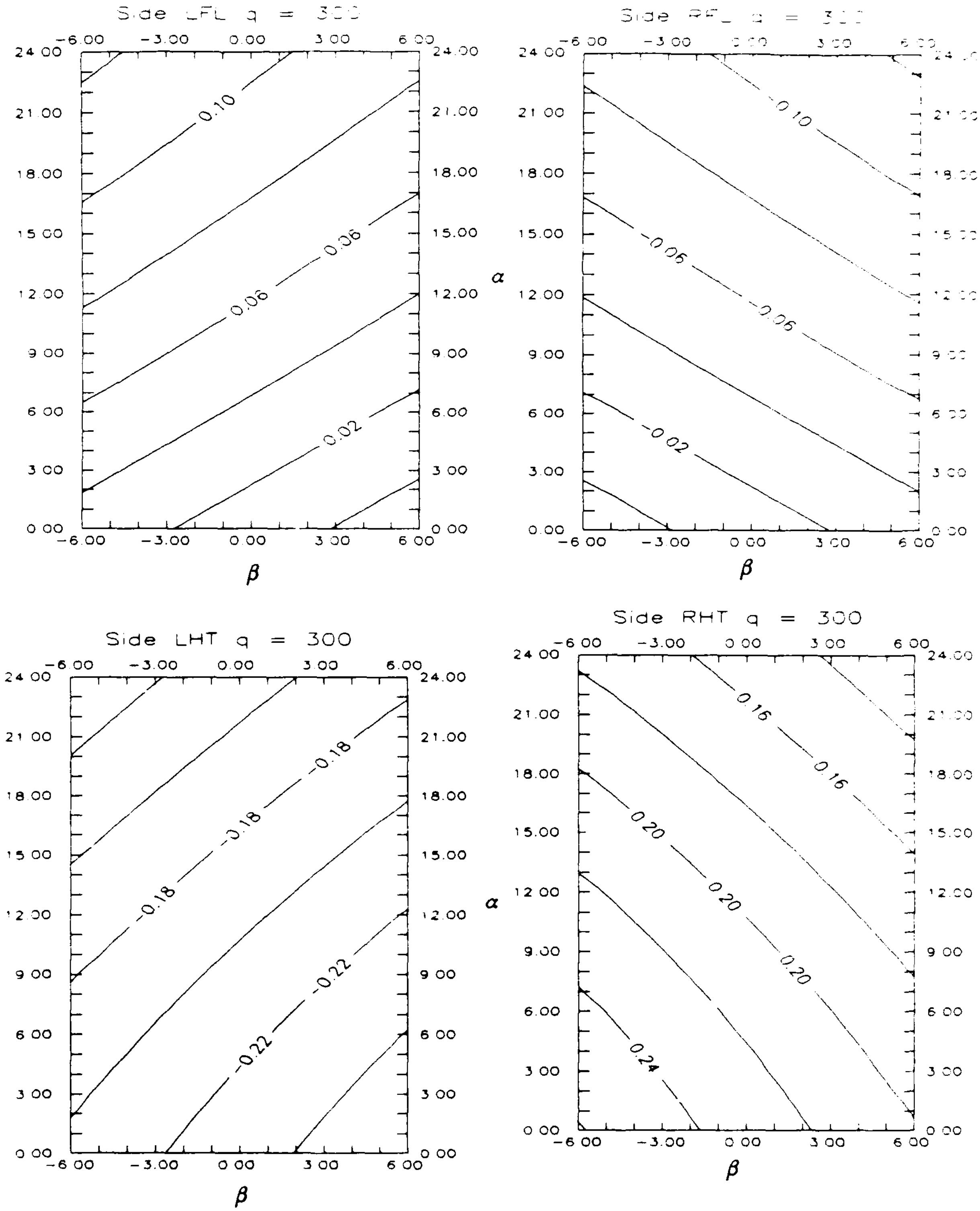


Figure 39 Normalized Side Derivatives

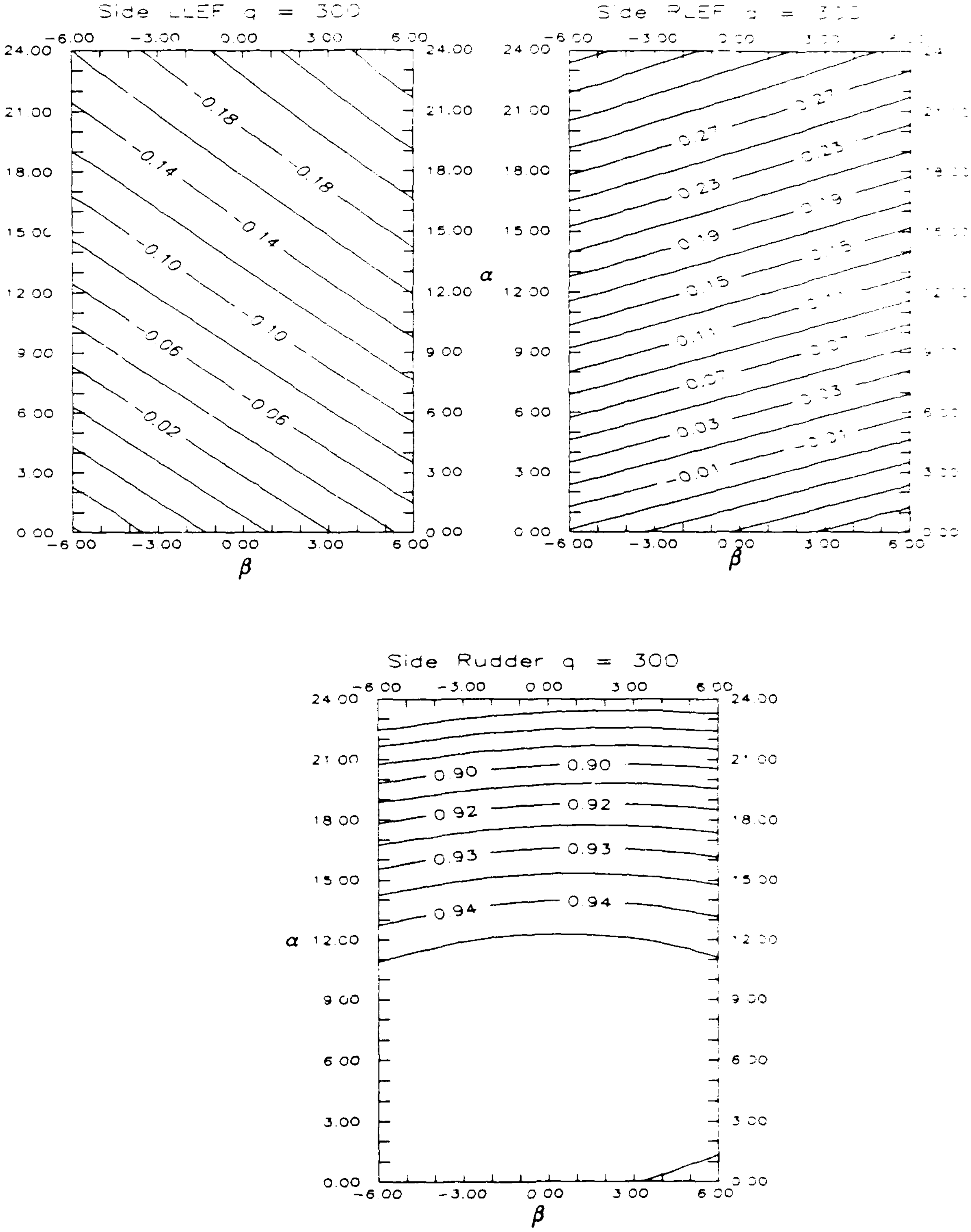


Figure 40 Normalized Side Derivatives

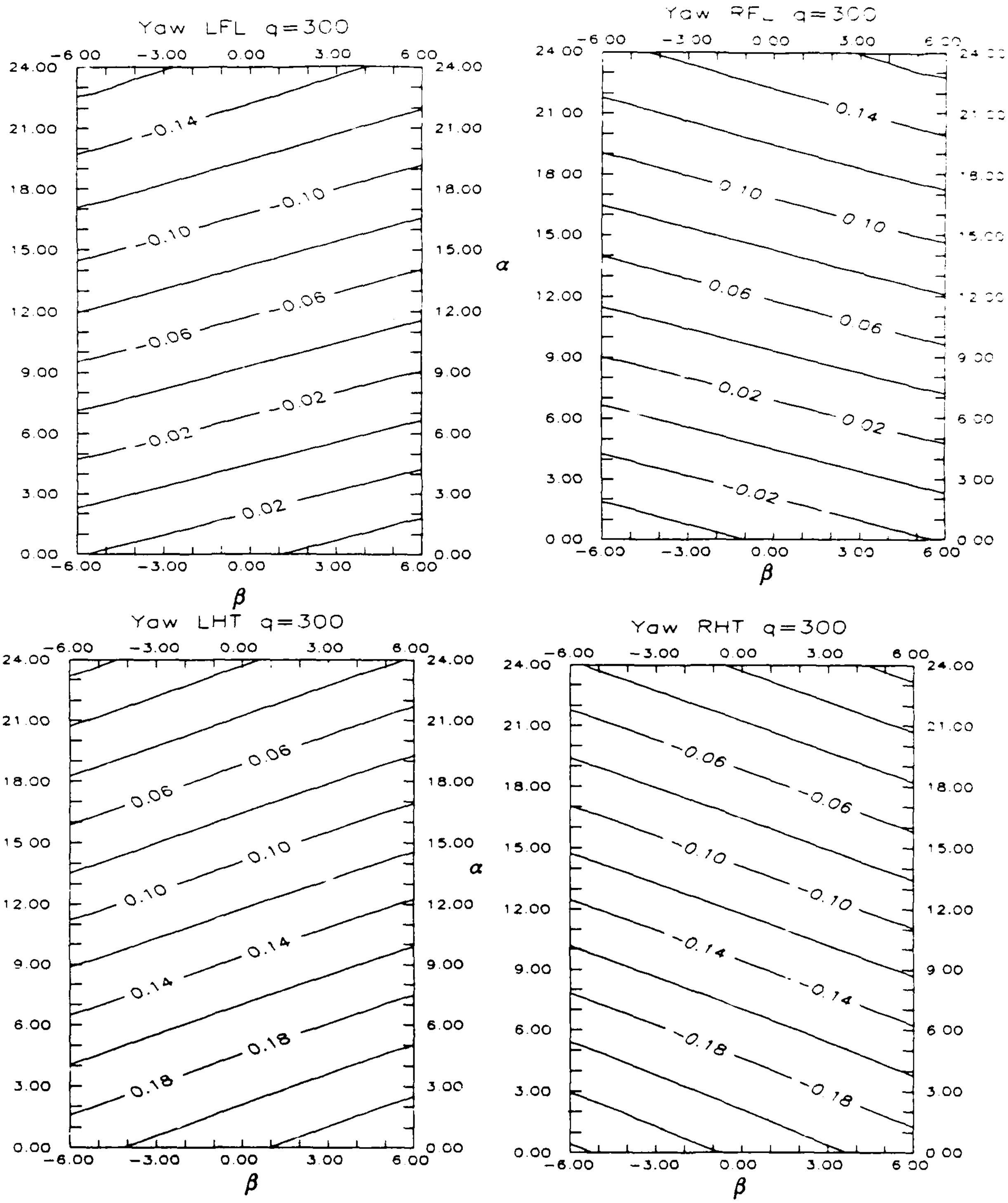


Figure 41 Normalized Yaw Derivatives

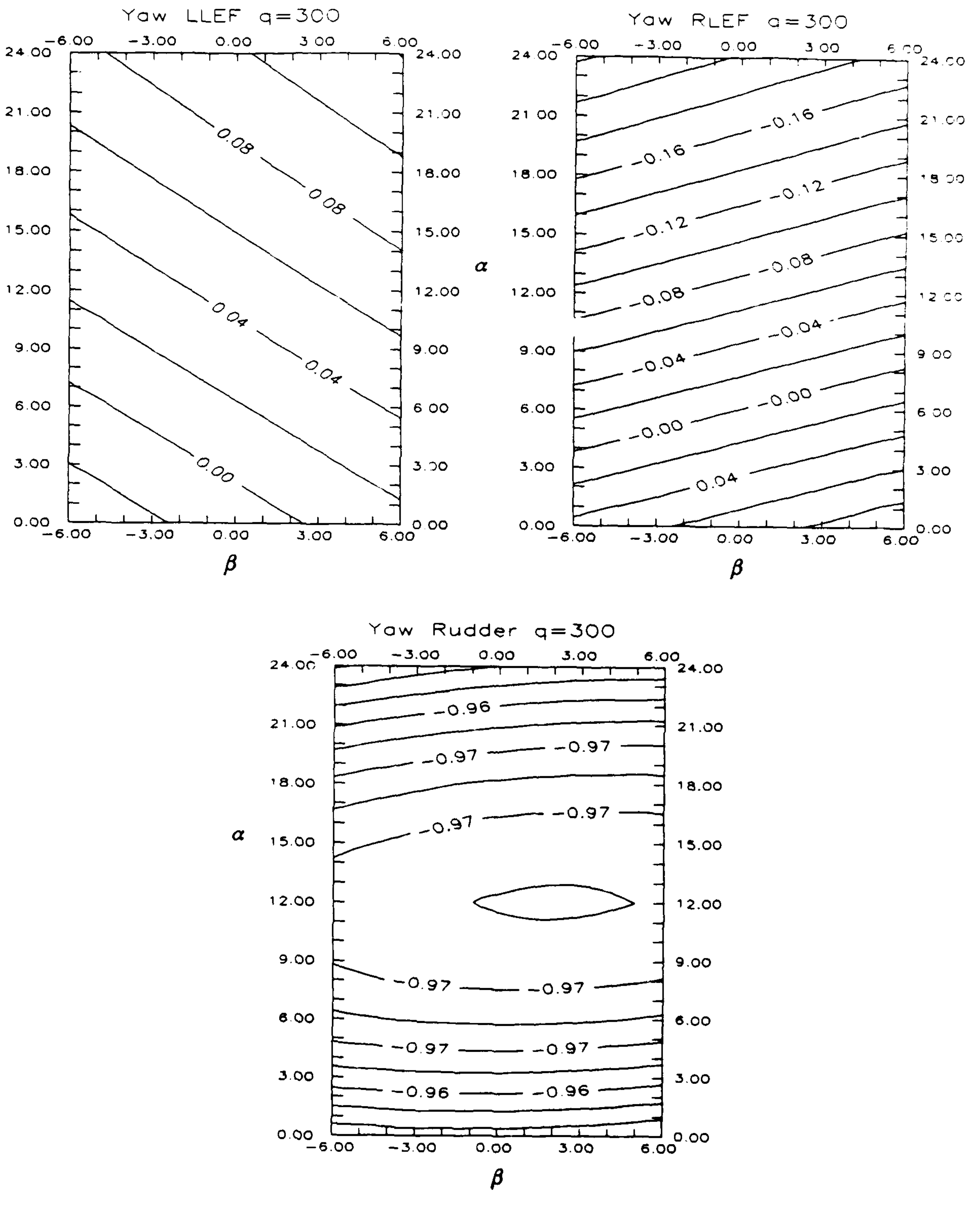


Figure 42 Normalized Yaw Derivatives

Coutours of Aerodynamic Coefficients

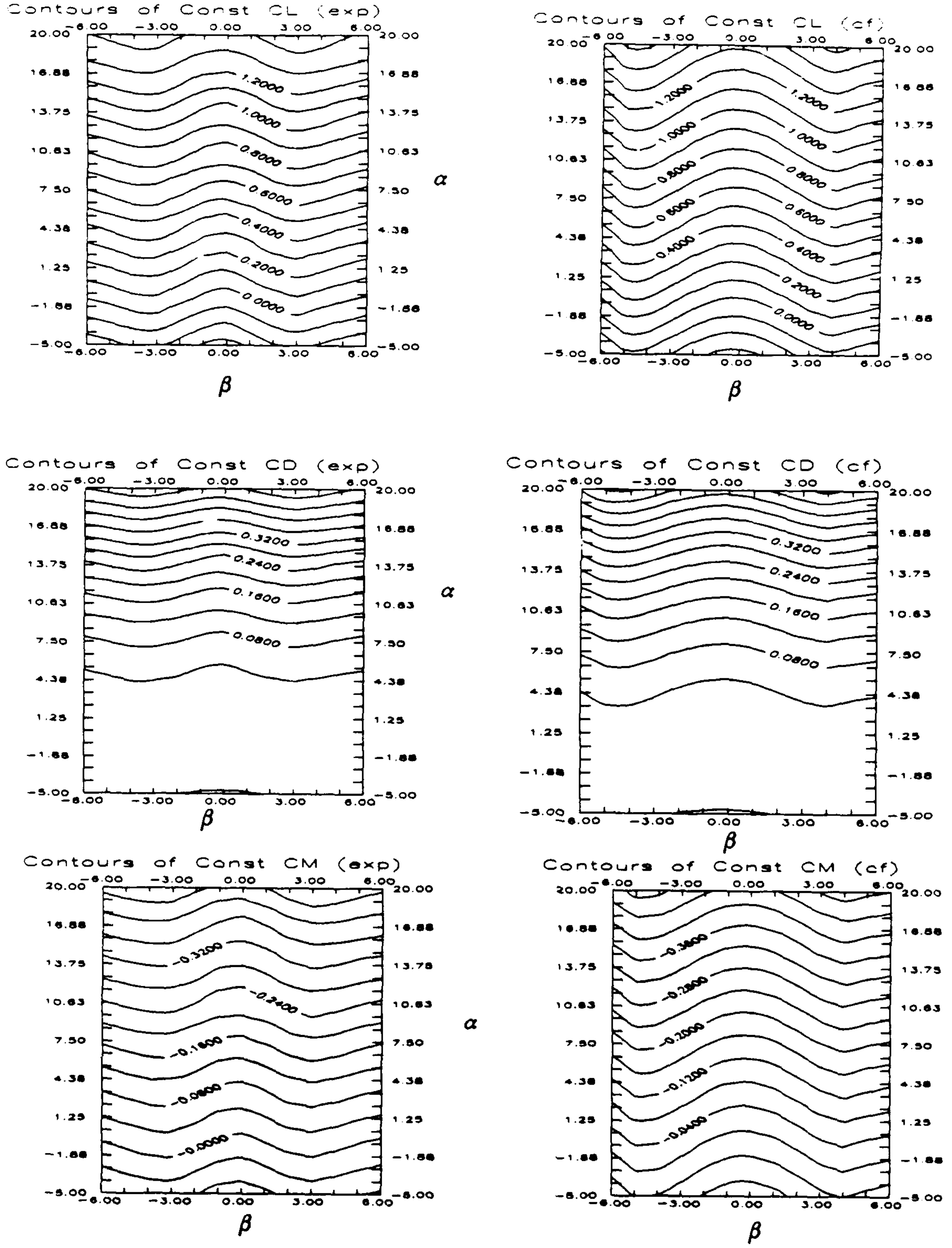


Figure 43 Longitudinal Coefficients

Contours of Aerodynamic Coefficients (continued)

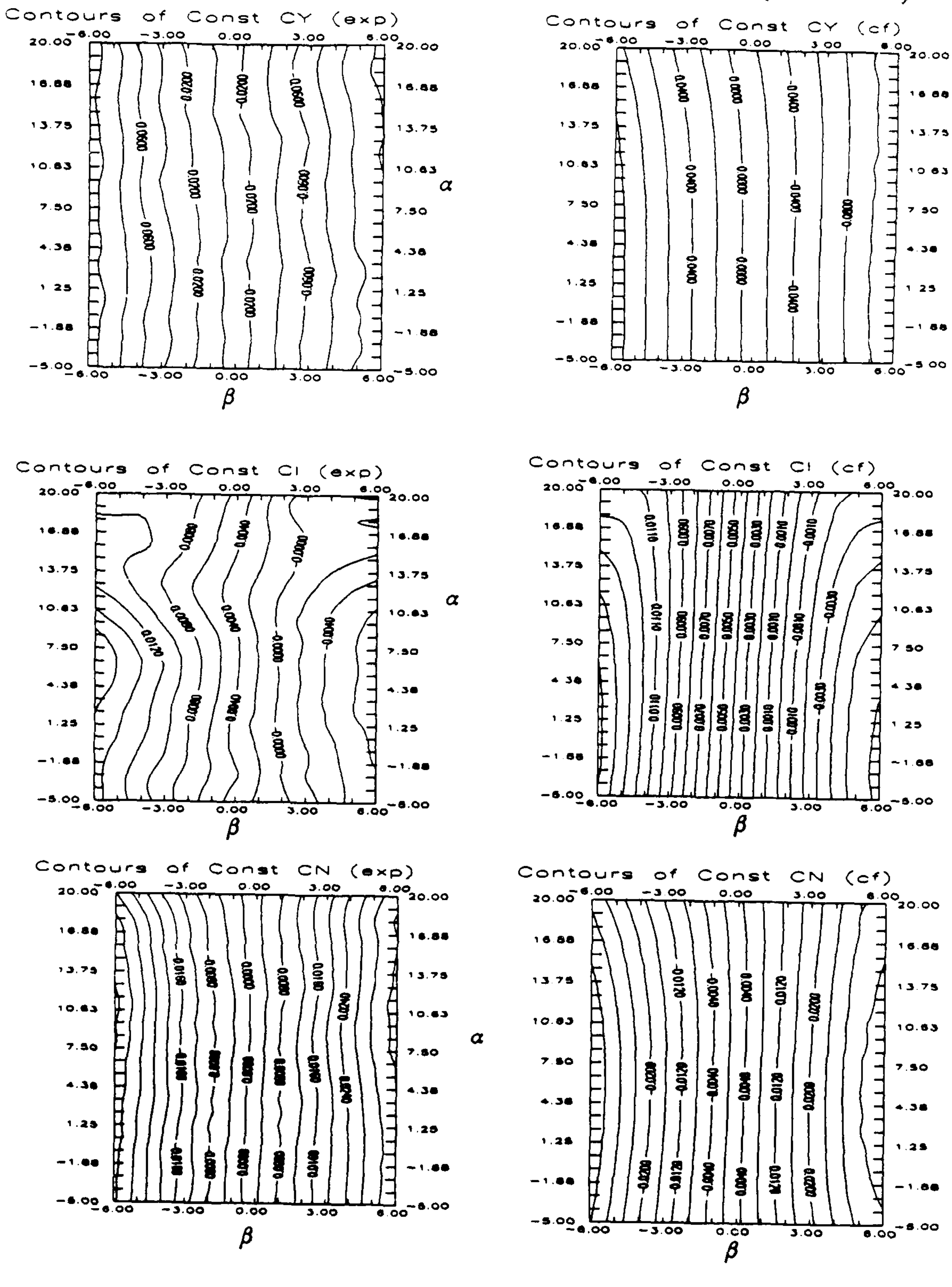


Figure 44 Lateral Coefficients

APPENDIX F

The three codes used to perform the investigations of equilibrium regions are contained in this appendix. Autrima.for represents Case A, Autrimb.for represents Case B and Autrimc.for represents Case C. A flow chart is presented in Chapter IV which provides a schematic description of the operations of the codes.

Autrimb.for

```
c
c    ****************************************************
c              TRIMHb.FOR
c    ****************************************************
c    17 Oct 89 SMZ
c
     implicit real*8 (a-h,o-z)
c
c
     parameter ( gw = 19000, cg =27.208, btail=63.7)
     parameter (span =29.0,chord=10.937,wing=300,vtail=54.75)
     parameter (alt = 10000)
     parameter (llemin=-2,llemax=25,lflmin=-20,lflmax=20)
     parameter (rflmin=-20,rflmax=20,lhtmin=-25,lhtmax=25)
     parameter (rhtmin=-25,rhtmax=25)
     parameter (rlemin=-2.5,rlemax=25,pi=3.1415927)
     parameter (nf = 4,msize = 4)
c
c
     real*8  a(4,4),delta(4),b(4),zer(6)
     real*8  ranmin,ranmax,rud,lle,rle,rfl,lfl,lht,rht
     real*8 alpha,beta,ra,rb,rd,min(4),max(4),delmn
     real*8 iaz(20,6),ibz(20,6),idz(20,6),coefz(20,6),nofncz(6)
     real*8 ialle(20,6),iblle(20,6),idlle(20,6),coeflle(20,6)
     real*8 iarle(20,6),ibrle(20,6),idrle(20,6),coefrle(20,6)
     real*8 iarud(20,6),ibrud(20,6),idrud(20,6),coefrud(20,6)
     real*8 iarfl(20,6),ibrfl(20,6),idrfl(20,6),coefrfl(20,6)
     real*8 iarht(20,6),ibrht(20,6),idrht(20,6),coefrht(20,6)
     real*8 ialfl(20,6),iblfl(20,6),idlfl(20,6),coeflfl(20,6)
     real*8 ialht(20,6),iblht(20,6),idlht(20,6),coeflht(20,6)
     real*8 nflle(6),nfrle(6),nfrud(6),nfrfl(6),nflfl(6),cflle(6)
     real*8 nfrht(6),nflht(6),cfrle(6),cfz(6),cfrud(6),cfrfl(6)
     real*8 cflfl(6),cfrht(6),cflht(6),cft(6),ralp,phi1,phi1r
     real*8 crle(6),cz(6),crud(6),crfl(6),clle(6),sbet,tbet,cbet
     real*8 clfl(6),crht(6),clht(6),ct(6),cthr,stht,thtadj
     real*8 fay,alpmin,alpmax,betmin,betmax,sinarea,pauth,rauth
     real*8 inda,indb,indd,phi2r,err
c
c
     character*20  trim,notrim,phicon,drag,detail,auth
c
c
     external f3
     external poly


c
c    The purpose of this program is to search for trim solutions for
c    the F-16 given a rudder failure and the angle of deflection at which
c    this surface is locked into a "hardover" failure. Coeficients for the
c    computation of aerodynamic forces must be supplied as data files
c    which are called into subroutines in this program. This program assumes
c    a steady state condition of straight flight and that linear superposition
c    holds. The LEFS are scheduled in this program as a function of A.O.A.
C     Wings level flight is not enforced and so in general the roll angle will
c    have a non zero value. The flight path angle is specified at zero.
c    This program uses the two flaperons and the two horizontal tails
c    independently to achieve a trim solution.
c
c    The control surfaces in the delta vector are numbered as
c    follows:
c          1. Port Flaperon
c          2. Starboard Flaperon
c          3. Port Horizontal Tail
c          4. Starboard Horizontal Tail
c
c
c
c    This version of the program will write the specified information to data
c    files which can be evaluated in either SURFER or GRAPHER.
c
     write(6,*) '*************************************'
     write(6,*) '          AUTRIMB'
     write(6,*) '*************************************'
     write(6,*) ''
     write(6,*) 'Please enter the specified rudder defl in degs:'
     read(5,*) rud
     write(6,*) ''
     write(6,*) 'Please enter the min alpha in degs: '
     read(5,*) alpmin
     write(6,*) ''
     write(6,*) 'Please enter the max alpha in degs: '
     read(5,*) alpmax
     write(6,*) ''
     write(6,*) 'Please enter the index for alpha: '
     read(5,*) inda
     write(6,*) ''
     write(6,*) 'Please enter the min beta in degs: '
     read(5,*) betmin
     write(6,*) ''
     write(6,*) 'Please enter the max beta in degs: '
     read(5,*) betmax
     write(6,*) ''
     write(6,*) 'Please enter the index for beta: '
     read(5,*) indb
     write(6,*) ''
     write(6,*) ''
     write(6,*) ' The currently selected ranges for trim investigation
    x are as follows:'
     write(6,*) ''
     write(6,*) 'Failed surface:  Rudder'
     write(6,*) ''
     write(6,*) '     Min alpha:', alpmin
     write(6,*) '     Max alpha:', alpmax
     write(6,*) ''
     write(6,*) '      Min beta:', betmin
     write(6,*) '      Max beta:', betmax
c
c
     write(6,*) 'Enter a filename for trim solutions:'
     read(5,5000) trim
     open(12,file=trim,status='new')
c
```

```
c
      write(6,*) 'Enter a filename for Phi contours:'
      read(5,5000) phicon
      open(11,file=phicon,status='new')
c
c
      write(6,*) 'Enter a filename for Drag coef contours:'
      read(5,5000) drag
      open(10,file=drag,status='new')
c
c
c
      write(6,*) 'Enter a filename for Mean aileron contours:'
      read(5,5000) detail
      open(9,file=detail,status='new')
c
c
      write(6,*) 'Enter a filename for control authority contours:'
      read(5,5000) auth
      open(8,file=auth,status='new')
c
      write(6,*) ' '
      write(6,*) 'Opening file',trim
c
c     Initialize the min and max comparison vectors
c
      min(1) = lflmin
      min(2) = rflmin
      min(3) = lbtmin
      min(4) = rbtmin
c
      max(1) = lflmax
      max(2) = rflmax
      max(3) = lbtmax
      max(4) = rbtmax
c
c     initialize the ranges
c
c
      ralp =((alpmax-alpmin)/inda) + 1
      rb =((betmax-betmin)/indb) + 1
      z = 0
      t = 0
c
c
      call dynpres(mach,qbar)
      write(6,*) 'The value of the dynamic pressure is:', qbar
c
c
c     Call in the polynomial predictor equations for the forces
c     and moments.
c

      call fixzer(iaz,ibz,idz,coefz,nofncz)
      call fixlle(ialle,iblle,idlle,coeflle,nflle)
      call fixrle(iarle,ibrle,idrle,coefrle,nfrle)
      call fixrud(iarud,ibrud,idrud,coefrud,nfrud)
      call fixrfl(iarfl,ibrfl,idrfl,coefrfl,nfrfl)
      call fixrbt(iarbt,ibrbt,idrbt,coefrbt,nfrbt)
      call fixlfl(ialfl,iblfl,idlfl,coeflfl,nflfl)
      call fixlbt(ialbt,iblbt,idlbt,coeflbt,nflbt)
      write(6,*) 'finished reading files'
c
c
       beta = betmin
       do 200 j=1,rb
         alpha = alpmin
         do 300 k=1,ralp
           call lef(qbar,alpha,lle,rle,cflle,cfrle,ialle,iblle,idlle,
     x     beta,coeflle,nflle,iarle,ibrle,idrle,coefrle,nfrle,clle,
     x     crle)
           call zero(qbar,alpha,beta,iaz,ibz,idz,coefz,nofncz,cfz,cz)

           call failed(qbar,alpha,beta,rud,iarud,ibrud,idrud,coefrud,
     x           nfrud,cfrud,crud)
c
c
      alpr = alpha*(pi/180)
      betr = beta*(pi/180)
c
c     Specify the Flight Path angle equal to zero
c     which implies first estimate of theta is alpha
c
      thtr = alpr
      salpha = sin(alpr)
      calpha = cos(alpr)
      sbeta = sin(betr)
      cbeta = cos(betr)
      ctht = cos(thtr)
      talp = tan(alpr)
      tbet = tan(betr)
c
c     Calculate the aero forces in the body z and x axis respectively
c
      faz = salpha*(-1*(cfz(2)+cflle(2)+cfrle(2)))
     x     + calpha*(-1*(cfz(1)+cflle(1)+cfrle(1)))
      fax = calpha*(-1*(cfz(2)+cflle(2)+cfrle(2)))
     x     - salpha*(-1*(cfz(1)+cflle(1)+cfrle(1)))
      fay = (cfz(3) + cflle(3) + cfrle(3) + cfrud(3))
c
c     Calculate first estimate of Phi from side force eq
c
      fgw = fay/(-1*gw*ctht)
      if ((fgw.gt.1.0) then
        goto 600
      else if ((fgw.lt.-1.0) then
        goto 600
      else
        phi1r = asin(fgw)
      endif
c
c
50    cphi = cos(phi1r)
      ctht = cos(thtr1)
      stht = sin(thtr1)
      ftx = gw*stht - fax
      phi1 = phi1r*(180/pi)
c
c     Construct the lefthand side of the linear problem with known
c     force and moment data.
c
c     The b vector contains the following forces and moments by row
c
c     1. Normal
c     2. Pitch
c     3. Roll
c     4. Yaw
c
      b(1) = -1*gw*cphi*ctht - faz
      do 700 l = 1,3
        m = l + 3
        n = l+1
        b(n) = -1 * (cfz(m) + cflle(m) + cfrle(m) + cfrud(m))
700   continue
c
c     Assemble the A matrix to be used in the linear problem.
c     This matrix is composed of the control derivatives of the
c     controls that will be used to effect a trim solution.
c
      call flaper(qbar,alpha,beta,iarfl,ibrfl,idrfl,coefrfl,nrfl,
     x        cfrfl,crfl)
      call flaper(qbar,alpha,beta,ialfl,iblfl,idlfl,coeflfl,nlfl,
     x        cflfl,clfl)
      call brztail(qbar,alpha,beta,iarbt,ibrbt,idrbt,coefrbt,nrbt,
     x        cfrbt,crbt)
      call brztail(qbar,alpha,beta,ialbt,iblbt,idlbt,coeflbt,nlbt,
     x        cflbt,clbt)
c
c
      a(1,1) = -1*cflfl(2)*salpha -1*cflfl(1)*calpha
      a(1,2) = a(1,1)
      a(1,3) = -1*cflbt(2)*salpha -1*cflbt(1)*calpha
      a(1,4) = a(1,3)
c
c
      do 800 l = 1,3
        m = l + 3
        n = l + 1
        a(n,1) = cflfl(m)
        a(n,2) = cfrfl(m)
        a(n,3) = cflbt(m)
        a(n,4) = cfrbt(m)
800   continue
c
c
c     Solve the linear problem which has been set up.
c
        call svd_solve(a,b,delta,nf,nf,msize,msize)

c
c     Sum up side forces due to control deflections
c
      fay = (cfz(3) + cflle(3) + cfrle(3) + cfrud(3))
      fay = fay + (delta(1) * cflfl(3))
```

```
      fay = fay + (delta(2) * cfrfl(3))
      fay = fay + (delta(3) * cflht(3))
      fay = fay + (delta(4) * cfrht(3))
c
c     Sum up Normal forces due to control deflections
c
      fazt = salpha*(-1*(cflz(2)+cflle(2)+cfrle(2)))
     x      + calpha*(-1*(cflz(1)+cflle(1)+cfrle(1)))
c
      a(1,1) = -1*cflfl(2)*salpha -1*cflfl(1)*calpha
      a(1,2) = -1*cfrfl(2)*salpha -1*cfrfl(1)*calpha
      a(1,3) = -1*cflht(2)*salpha -1*cflht(1)*calpha
      a(1,4) = -1*cfrht(2)*salpha -1*cfrht(1)*calpha
c
      fazt = fazt + (delta(1) * a(1,1))
      fazt = fazt + (delta(2) * a(1,2))
      fazt = fazt + (delta(3) * a(1,3))
      fazt = fazt + (delta(4) * a(1,4))
c
c     Adjust Pitch angle for the new roll angle
c
      thtadj =talp*(-1*fazt/gw)+(thet/calpha)*(-1*fay/gw)
      if (thtadj.gt.1.0) then
       goto 600
      else if (thtadj.lt.-1.0) then
       goto 600
      else
      thtr2=asin(thtadj)
      endif
      ctht = cos(thtr2)


c
c     Adjust Roll angle for new theta and control deflections
c
      fgw = fay/(-1*gw*ctht)
      if (fgw.gt.1.0) then
       goto 600
      else if (fgw.lt.-1.0) then
       goto 600
      else
      phi2r = asin(fgw)
      endif
c
c
      err1 = sqrt((phi1r - phi2r)**2)
      err2 = sqrt((thtr1 - thtr2)**2)
c     write(6,*) 'The error is:',err
c
c     Determine if new phi angle is within
c     .0001 radians of first approximation
c
      if (z.gt.21) then
        goto 525
      else if(err1.gt..0001) then
        phi1r = phi2r
        thtr1 = thtr2
        z = z + 1
        goto 50
      else if(err2.gt..0001) then
        phi1r = phi2r
        thtr1 = thtr2
        z = z + 1
        goto 50
      endif
c
c     Determine if the computed solution violates constraints on
c     control surface deflection limits and write the data to the
c     apropriate file.
c
         do 400 l=1,4
          if(delta(l).lt.min(l) .or. delta(l).gt.max(l)) then
           goto 510
          else
          endif
400       continue
c
450      continue
       phi2 = phi2r*(180/pi)
       theta = thtr2*(180/pi)
c
c
c       write(6,*) ''
c       write(6,*) 'The value of alpha is:    ',alpha
c       write(6,*) 'The value of beta is:     ',beta
c       write(6,*) 'The rudder deflection is: ',rud
c       write(6,*) ''
c       WRITE (6,*) 'The value of the Roll angle is:',phi2
c       write(6,*) ''
c       WRITE (6,*) 'The value of the Pitch Angle is:',theta
c       write (6,*) ''
c       write(6,*) ' A solution exists at this point:'
c       write(6,*) '$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$'
c       write(6,*) ''
c       write(6,*) 'LFL',delta(1)
c       write(6,*) ''
c       write(6,*) 'RFL',delta(2)
c       write(6,*) ''
c       write(6,*) 'LHT',delta(3)
c       write(6,*) ''
c       write(6,*) 'RHT',delta(4)
c       write(6,*) ''
c       write(6,*) '$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$'
c
c
       delmn = (delta(1) + delta(2))/2
c
c
       cdt = crle(2)+cz(2)+crud(2)+crfl(2)+clle(2)+
     x      clfl(2)+crht(2)+clht(2)
c
c       Compute the solution area as of this pass.
c
          t = t+1
       slnarea = t*(inda*indb)
c
c     Calculate remaining pitch and roll authority
c
     call author(cflfl,cfrfl,cflht,cfrht,delta,pauth,rauth)

c
c       Write output to file for plotting in Grapher or Surfer
c
       write(12,60000) beta,alpha,rud,slnarea
       write(11,60000) beta,alpha,phi2,rud
       write(10,60000) beta,alpha,cdt,rud
       write(9,60000) beta,alpha,delmn,rud
       write(8,60000) beta,alpha,pauth,rauth
c
       goto 325
c
500     continue
c
c
c
515     continue
c       write(6,*)
'===============================================
======='
c       write(6,*) '    NO SOLUTION AT THIS POINT'
c       WRITE(6,*)
'===============================================
======='
       goto325
c
c
525     continue
c
c
c       write(6,*)
'===============================================
======='
c       write(6,*) '  SOLUTION WILL NOT CONVERGE AT THIS
POINT'
c       WRITE(6,*)
'===============================================
======='
c
c
c
c
       goto 325
600     continue
c       write(6,*) '*************************************'
c       write(6,*) ' Steady state lift condition violated'
c       write(6,*) ' Selecting next alpha value'
c       write(6,*) '*************************************'
325     alpha = alpha + inda
c
c       Reinitialize z
c

        z = 0
c
300     continue
```

```
c
      beta = beta + indb
c
200   continue
c
c
c
100 continue
c
      close(12)
      close(11)
      close(10)
      close(9)
      close(8)
c
c
      write(6,*) 'The data search is complete.'
      write(6,*) 'The total solution area is:',alnarea
c
c
5000 Format(a20)
10000 Format(f5.2)
20000 Format(4(2x,f10.7))
30000 Format(4(f4.2))
40000 Format(5(f4.2))
50000 Format(4(1x,f12.4))
60000 Format(f9.5,3(1x,f9.5))
      stop
      end
c
c     *********************************************
c              DYNPRSS
c     *********************************************
c
      subroutine dynprss(mach,q)
c
c     The pupose of this program is to determine dynamic pressure
c     based on the specified flight mach number and altitude.  It
c     is currently "wired" to request a value for q.
c
      real*8 as,vel,q
      parameter (gamma=1.4,rho=.0017564,t = 525)
      parameter (gc = 32.174,r=53.34)

c
c     as = sqrt(gamma*r*t*gc)
c     vel = mach * as
c     q = .5*rho*(vel**2)
      write(6,*) 'Please enter a value for q:'
      read(5,*) q
      return
      end



c
c     *********************************************
c              FIXZER
c     *********************************************
c
      subroutine fixzer(iaz,ibz,idz,coefz,nofncz)
c
      integer alp, bet, del, comb, nofn
      integer lift,drag,side,pitch,roll,yaw
      real*8 s(5), rsqr
      real*8 iaz(20,6),ibz(20,6),idz(20,6),coefz(20,6),nofncz(6)
      character * 10 force,control
      character * 11 name
      character * 4 old
      character * 4 ext
      character * 1 id(6)
      data id/'1','2','3','4','5','6'/
c
c
      lift = 1
      drag = 2
      side = 3
      pitch = 4
      roll = 5
      yaw = 6
c
c     column identifiers
c
      ifcns = 1
      ialpha = 2
      ibeta = 3
      idelta = 4
      izcf = 5
c
c
      old = 'fixz'
      ext = '.dat'
      do 100 i=1,6
        name = old//id(i)//ext
        open(14,file=name,status='old')
        read(14,10000) control
c       write(6,*) control
        read(14,10000) force
c       write(6,*) force
        read(14,*) rsqr
c       write(6,*) rsqr
        read(14,*) nofn
c
c
        do 15 j=1,60
          read(14,*,end=25,err=35) (s(k1),k1=1,5)
          iaz(j,i) = s(ialpha)
          ibz(j,i) = s(ibeta)
          idz(j,i) = s(idelta)
          coefz(j,i) = s(izcf)
          nofncz(i) = s(ifcns)
c       write(6,20000) iaz(j,i),ibz(j,i),idz(j,i),coefz(j,i)
15      continue
25      close(14)
35      continue
        close(14)
100     continue
c     write(6,*) 'Finished in Fixzer'
10000   Format(2x,a10)
20000   Format(4(2x,f10.7))
        return
        end
c
c     *********************************************
c              FIXLLB
c     *********************************************
c
      subroutine fixlle(ialle,iblle,idlle,coeflle,nflle)
c
      integer alp, bet, del, comb, nofn
      integer lift,drag,side,pitch,roll,yaw
      real*8 s(5), rsqr,nflle(6)
      real*8 ialle(20,6),iblle(20,6),idlle(20,6),coeflle(20,6)
      character * 10 force,control
      character * 11 name
      character * 6 old
      character * 4 ext
      character * 1 id(6)
      data id/'1','2','3','4','5','6'/
c
c
      lift = 1
      drag = 2
      side = 3
      pitch = 4
      roll = 5
      yaw = 6
c
c     column identifiers
c
      ifcns = 1
      ialpha = 2
      ibeta = 3
      idelta = 4
      illecf = 5
c
c
      old = 'fixlle'
      ext = '.dat'
      do 100 i=1,6
        name = old//id(i)//ext
        open(14,file=name,status='old')
        read(14,10000) control
c       write(6,*) control
        read(14,10000) force
c       write(6,*) force
        read(14,*) rsqr
c       write(6,*) rsqr
        read(14,*) nofn
c
c
        do 15 j=1,60
          read(14,*,end=25,err=35) (s(k1),k1=1,5)
```

```
      ialle(j,i) = s(ialpha)
      iblle(j,i) = s(ibeta)
      idlle(j,i) = s(idelta)
      coeflle(j,i) = s(illecf)
      nflle(i) = s(ifcns)
15    continue
25    close(14)
35    write(6,*)'Reading:',name
     close(14)
100   continue
c     write(6,*) 'Finished in Fixlle'
10000  Format(2x,a10)

      return
      end
c
c  **********************************************
c           FIXRLE
c  **********************************************
c
      subroutine fixrle(iarle,ibrle,idrle,coefrle,nfrle)
c
      integer alp, bet, del, comb, nofn
      integer lift,drag,side,pitch,roll,yaw
      real*8 s(5), rsqr,nfrle(6)
      real*8 iarle(20,6),ibrle(20,6),idrle(20,6),coefrle(20,6)
      character * 10 force,control
      character * 11 name
      character * 6 old
      character * 4 ext
      character * 1 id(6)
      data id/'1','2','3','4','5','6'/
c
c
      lift = 1
      drag = 2
      side = 3
      pitch = 4
      roll = 5
      yaw = 6
c
c     column identifiers
c
      ifcns = 1
      ialpha = 2
      ibeta = 3
      idelta = 4
      irlecf = 5
c
c
      old = 'fixrle'
      ext = '.dat'
      do 100 i=1,6
        name = old//id(i)//ext
        open(14,file=name,status='old')
        read(14,10000) control
c       write(6,*) control
        read(14,10000) force
c       write(6,*) force
        read(14,*) rsqr
c       write(6,*) rsqr
        read(14,*) nofn
c
c
        do 15 j=1,60
          read(14,*,end=25,err=35) (s(k1),k1=1,5)
          iarle(j,i) = s(ialpha)
          ibrle(j,i) = s(ibeta)
          idrle(j,i) = s(idelta)
          coefrle(j,i) = s(irlecf)
          nfrle(i) = s(ifcns)
15      continue
25      close(14)
35      write(6,*)'Reading:',name
        close(14)
100     continue
c     write(6,*) 'Finished in Fixrle'
10000  Format(2x,a10)

      return
      end
c
c
c  **********************************************
c           FIXRUD
c  **********************************************
c
      subroutine fixrud(iarud,ibrud,idrud,coefrud,nfrud)
c
      integer alp, bet, del, comb, nofn
      integer lift,drag,side,pitch,roll,yaw
      real*8 s(5), rsqr,nfrud(6)
      real*8 iarud(20,6),ibrud(20,6),idrud(20,6),coefrud(20,6)
      character * 10 force,control
      character * 11 name
      character * 6 old
      character * 4 ext
      character * 1 id(6)
      data id/'1','2','3','4','5','6'/
c
c
      lift = 1
      drag = 2
      side = 3
      pitch = 4
      roll = 5
      yaw = 6
c
c     column identifiers
c
      ifcns = 1
      ialpha = 2
      ibeta = 3
      idelta = 4
      irudcf = 5
c
c
      old = 'fixrud'
      ext = '.dat'
      do 100 i=1,6
        name = old//id(i)//ext
        open(14,file=name,status='old')
        read(14,10000) control
c       write(6,*) control
        read(14,10000) force
c       write(6,*) force
        read(14,*) rsqr
c       write(6,*) rsqr
        read(14,*) nofn
c
c
        do 15 j=1,60
          read(14,*,end=25,err=35) (s(k1),k1=1,5)
          iarud(j,i) = s(ialpha)
          ibrud(j,i) = s(ibeta)
          idrud(j,i) = s(idelta)
          coefrud(j,i) = s(irudcf)
          nfrud(i) = s(ifcns)
15      continue
25      close(14)
35      write(6,*)'Reading:',name
        close(14)
100     continue
c     write(6,*) 'Finished in Fixrud'
10000  Format(2x,a10)

      return
      end
c
c  **********************************************
c           FIXRFL
c  **********************************************
c
      subroutine fixrfl(iarfl,ibrfl,idrfl,coefrfl,nfrfl)
c
      integer alp, bet, del, comb, nofn
      integer lift,drag,side,pitch,roll,yaw
      real*8 s(5), rsqr,nfrfl(6)
      real*8 iarfl(20,6),ibrfl(20,6),idrfl(20,6),coefrfl(20,6)
      character * 10 force,control
      character * 11 name
      character * 6 old
      character * 4 ext
      character * 1 id(6)
      data id/'1','2','3','4','5','6'/
c
c
      lift = 1
      drag = 2
      side = 3
      pitch = 4
      roll = 5
      yaw = 6
c
```

```
c     column identifiers
c
      ifcns = 1
      ialpha = 2
      ibeta = 3
      idelta = 4
      irflcf = 5
c
c
      old = 'fixrfl'
      ext = '.dat'
      do 100 i=1,6
        name = old//id(i)//ext
        open(14,file=name,status='old')
        read(14,10000) control
c       write(6,*) control
        read(14,10000) force
c       write(6,*) force
        read(14,*) rsqr
c       write(6,*) rsqr
        read(14,*) nofn
c
c
        do 15 j=1,60
          read(14,*,end=25,err=35) (s(k1),k1=1,5)
          iarfl(j,i) = s(ialpha)
          ibrfl(j,i) = s(ibeta)
          idrfl(j,i) = s(idelta)
          coefrfl(j,i) = s(irflcf)
          nfrfl(i) = s(ifcns)
15      continue
25      close(14)
35      write(6,*)'Reading:',name
        close(14)
100   continue
c     write(6,*) 'Finished in Fixrfl'
10000 Format(2x,a10)

      return
      end
c
c     **********************************************
c               FIXLFL
c     **********************************************
c
      subroutine fixlfl(ialfl,iblfl,idlfl,coeflfl,nlfl)
c
      integer alp, bet, del, comb, nofn
      integer lift,drag,side,pitch,roll,yaw
      real*8 s(5), rsqr,nlfl(6)
      real*8 ialfl(20,6),iblfl(20,6),idlfl(20,6),coeflfl(20,6)
      character * 10 force,control
      character * 11 name
      character * 6 old
      character * 4 ext
      character * 1 id(6)
      data id/'1','2','3','4','5','6'/
c
c
      lift = 1
      drag = 2
      side = 3
      pitch = 4
      roll = 5
      yaw = 6
c
c     column identifiers
c
      ifcns = 1
      ialpha = 2
      ibeta = 3
      idelta = 4
      ilflcf = 5
c
c
      old = 'fixlfl'
      ext = '.dat'
      do 100 i=1,6
        name = old//id(i)//ext
        open(14,file=name,status='old')
        read(14,10000) control
c       write(6,*) control
        read(14,10000) force
c       write(6,*) force
        read(14,*) rsqr
c       write(6,*) rsqr
        read(14,*) nofn
c
c
        do 15 j=1,60
          read(14,*,end=25,err=35) (s(k1),k1=1,5)
          ialfl(j,i) = s(ialpha)
          iblfl(j,i) = s(ibeta)
          idlfl(j,i) = s(idelta)
          coeflfl(j,i) = s(ilflcf)
          nlfl(i) = s(ifcns)
15      continue
25      close(14)
35      write(6,*)'Reading:',name
        close(14)
100   continue
c     write(6,*) 'Finished in Fixlfl'
10000 Format(2x,a10)

      return
      end
c
c
c     **********************************************
c               FIXRHT
c     **********************************************
c
      subroutine fixrht(iarht,ibrht,idrht,coefrht,nfrht)
      integer alp, bet, del, comb, nofn
      integer lift,drag,side,pitch,roll,yaw
      real*8 s(5), rsqr,nfrht(6)
      real*8 iarht(20,6),ibrht(20,6),idrht(20,6),coefrht(20,6)
      character * 10 force,control
      character * 11 name
      character * 6 old
      character * 4 ext
      character * 1 id(6)
      data id/'1','2','3','4','5','6'/
c
c
      lift = 1
      drag = 2
      side = 3
      pitch = 4
      roll = 5
      yaw = 6
c
c     column identifiers
c
      ifcns = 1
      ialpha = 2
      ibeta = 3
      idelta = 4
      irhtcf = 5
c
c
      old = 'fixrht'
      ext = '.dat'
      do 100 i=1,6
        name = old//id(i)//ext
        open(14,file=name,status='old')
        read(14,10000) control
c       write(6,*) control
        read(14,10000) force
c       write(6,*) force
        read(14,*) rsqr
c       write(6,*) rsqr
        read(14,*) nofn
c
c
        do 15 j=1,60
          read(14,*,end=25,err=35) (s(k1),k1=1,5)
          iarht(j,i) = s(ialpha)
          ibrht(j,i) = s(ibeta)
          idrht(j,i) = s(idelta)
          coefrht(j,i) = s(irhtcf)
          nfrht(i) = s(ifcns)
15      continue
25      close(14)
35      write(6,*)'Reading:',name
        close(14)
100   continue
c     write(6,*) 'Finished in Fixrht'
10000 Format(2x,a10)

      return
      end
c
c
```

```
c     ************************************************
c             FIXLHT
c     ************************************************
c
      subroutine fixlht(ialbt,iblbt,idlbt,coeflbt,nflbt)
      integer alp, bet, del, comb, nofn
      integer lift,drag,side,pitch,roll,yaw
      real*8 s(5), rsqr,nflbt(6)
      real*8 ialbt(20,6),iblbt(20,6),idlbt(20,6),coeflbt(20,6)
      character * 10 force,control
      character * 11 name
      character * 6 old
      character * 4 ext
      character * 1 id(6)
      data id/'1','2','3','4','5','6'/
c
c
      lift = 1
      drag = 2
      side = 3
      pitch = 4
      roll = 5
      yaw = 6
c
c     column identifiers
c
      ifcns = 1
      ialpha = 2
      ibeta = 3
      idelta = 4
      ilbtcf = 5
c
c
      old = 'fixlbt'
      ext = '.dat'
      do 100 i=1,6
        name = old//id(i)//ext
        open(14,file=name,status='old')
        read(14,10000) control
c       write(6,*) control
        read(14,10000) force
c       write(6,*) force
        read(14,*) rsqr
c       write(6,*) rsqr
        read(14,*) nofn
c
c
        do 15 j=1,60
          read(14,*,end=25,err=35) (s(k1),k1=1,5)
          ialbt(j,i) = s(ialpha)
          iblbt(j,i) = s(ibeta)
          idlbt(j,i) = s(idelta)
          coeflbt(j,i) = s(ilbtcf)
          nflbt(i) = s(ifcns)
15      continue
25      close(14)
35      write(6,*)'Reading:',name
        close(14)
100   continue
c     write(6,*) 'Finished in Fixlbt'
10000 Format(2x,a10)

      return
      end
c
c
c
c     ************************************************
c             LEF
c     ************************************************
c
      subroutine lef(qbar,alpha,lle,rle,cflle,cfrle,ialle,iblle,idlle,
     x   beta,coeflle,nflle,iarle,ibrle,idrle,coefrle,nfrle,clle,crle)
c
c     The purpose of this program is to determine the Leading Edge
c     Flap setting based LEF scheduling.  From these settings the respective
c     force and moment data are calculated for use in the left hand side of
c     the linear equation.
c
      implicit real*8 (a-h,o-z)
      real * 8 mach,alpha,lle,rle,clle(6),crle(6),beta,x(3)
      real*8 ialle(20,6),iblle(20,6),idlle(20,6),coeflle(20,6)
      real*8 iarle(20,6),ibrle(20,6),idrle(20,6),coefrle(20,6)
      real*8 nflle(6),nfrle(6),func,cflle(6),cfrle(6)
      parameter ( gw = 19000, cg =27.208, btail=63.7)
      parameter (span =29.0,chord=10.937,wing=300,vtail=54.75)
      external f3
c
c     compute leading edge flap deflection in degrees
c
      if (alpha.le.-2) then
        lle = -2
        rle = -2
      elseif (alpha.ge.25) then
        lle = 25
        rle = 25
      else
        lle = 1.3164*alpha + 1.7
        rle = lle
      endif
c
c     initialize x vector for evaluating polynomial
c
      x(1) = alpha
      x(2) = beta
      x(3) = lle

c
c     evaluate predictor equations to obtain coeficients
c
      do 100 i=1,6
        clle(i) = 0.0
        do 200 j = 1,nflle(i)
          func = f3(j,x,ialle,iblle,idlle,i)
          clle(i) = clle(i) + coeflle(j,i)*func
200     continue
100   continue
c
c
c
      x(3) = rle
      do 300 i=1,6
        crle(i) = 0.0
        do 400 j = 1,nfrle(i)
          func = f3(j,x,iarle,ibrle,idrle,i)
          crle(i) = crle(i) + coefrle(j,i)*func
400     continue
300   continue
c
c     Calculate forces and moments and return.
c
      cflle(1) = clle(1)*qbar*wing
      cflle(2) = clle(2)*qbar*wing
      cflle(3) = clle(3)*qbar*wing
      cflle(4) = clle(4)*qbar*wing*chord
      cflle(5) = clle(5)*qbar*wing*span
      cflle(6) = clle(6)*qbar*wing*span
c
c
      cfrle(1) = crle(1)*qbar*wing
      cfrle(2) = crle(2)*qbar*wing
      cfrle(3) = crle(3)*qbar*wing
      cfrle(4) = crle(4)*qbar*wing*chord
      cfrle(5) = crle(5)*qbar*wing*span
      cfrle(6) = crle(6)*qbar*wing*span
c
c
      return
      end
c
c
c     ************************************************
c             ZERO
c     ************************************************
c
      subroutine zero(qbar,alpha,beta,iaz,ibz,idz,coefz,nofncz,cfz,cz)
c
c     The purpose of this subprogram is to calculate the values of
c     the force and moment coeficients at the delta equals zero condition.
c     these values are placed in the left hand b matrix
c
      implicit real*8 (a-h,o-z)
      real*8 iaz(20,6),ibz(20,6),idz(20,6),coefz(20,6),nofncz(6)
      real*8 alpha,beta,x(3),cz(6),cfz(6)
      parameter ( gw = 19000, cg =27.208, btail=63.7)
      parameter (span =29.0,chord=10.937,wing=300,vtail=54.75)
      external f3
c
c     initialize x vector for evaluating polynomial
c
      x(1) = alpha
      x(2) = beta
      x(3) = 0.0
c
```

```
c   evaluate predictor equations to obtain coeficients
c
    do 100 i=1,6
      cz(i) = 0.0
      do 200 j = 1,nofncz(i)
        func = f3(j,x,iaz,ibz,idz,i)
        cz(i) = cz(i) + coefz(j,i)*func
200     continue
100   continue
c
c   Calculate forces and moments and return.
c
c   write(6,*) 'the value of the drag coef is', cz(2)
    cfz(1) = cz(1)*qbar*wing
    cfz(2) = cz(2)*qbar*wing
    cfz(3) = cz(3)*qbar*wing
    cfz(4) = cz(4)*qbar*wing*chord
    cfz(5) = cz(5)*qbar*wing*span
    cfz(6) = cz(6)*qbar*wing*span
20000 Format(4(2x,f10.7))
    return
    end
c
c
c   **********************************************
c           FAILED
c   **********************************************
c
    subroutine failed(qbar,alpha,beta,rud,iarud,ibrud,idrud,coefrud,
   x              nfrud,cfrud,crud)
c
c   The purpose of this program is to determine the force and moment
c   values for the failed control surface for use in the left hand
c   "b" matrix. The rudder is currently programmed
c
    implicit real*8 (a-h,o-z)
    parameter ( gw = 19000, cg =27.208, btail=63.7)
    parameter (span =29.0,chord=10.937,wing=300,vtail=54.75)
    real*8 iarud(20,6),ibrud(20,6),idrud(20,6),coefrud(20,6)
    real*8 alpha,beta,x(3),crud(6),cfrud(6),nfrud(6),rud
    external f3
c
c   initialize x vector for evaluating polynomial
c
    x(1) = alpha
    x(2) = beta
    x(3) = rud

c
c   evaluate predictor equations to obtain coeficients
c
    do 100 i=1,6
      crud(i) = 0.0
      do 200 j = 1,nfrud(i)
        func = f3(j,x,iarud,ibrud,idrud,i)
        crud(i) = crud(i) + coefrud(j,i)*func
200     continue
100   continue
c
c   Calculate forces and moments and return.
c
    cfrud(1) = crud(1)*qbar*wing
    cfrud(2) = crud(2)*qbar*wing
    cfrud(3) = crud(3)*qbar*wing
    cfrud(4) = crud(4)*qbar*wing*chord
    cfrud(5) = crud(5)*qbar*wing*span
    cfrud(6) = crud(6)*qbar*wing*span
    return
    end
c
c
c
c   **********************************************
c           FLAPER
c   **********************************************
c
    subroutine flaper(qbar,alpha,beta,iarfl,ibrfl,idrfl,coefrfl,
   x         nrfl,cfrfl,crfl)
c
c   The purpose of this program is to calculate the control
c   derivatives for the right and left flaperon given a values
c   for q, alpha, beta.
c
c
c
c
    implicit real*8 (a-h,o-z)
    parameter ( gw = 19000, cg =27.208, btail=63.7)
    parameter (span =29.0,chord=10.937,wing=300,vtail=54.75)
    real*8 iarfl(20,6),ibrfl(20,6),idrfl(20,6),coefrfl(20,6)
    real*8 alpha,beta,x(3),crfl(6),cfrfl(6),nfrfl(6),cflfl(6)
    real*8 clfl(6)
c
c
    do 100 i=1,6
      crfl(i) = coefrfl(2,i)*alpha + coefrfl(3,i)*beta
   x          + coefrfl(1,i)
100 continue
c
c
c
    cfrfl(1) = crfl(1)*qbar*wing
    cfrfl(2) = crfl(2)*qbar*wing
    cfrfl(3) = crfl(3)*qbar*wing
    cfrfl(4) = crfl(4)*qbar*wing*chord
    cfrfl(5) = crfl(5)*qbar*wing*span
    cfrfl(6) = crfl(6)*qbar*wing*span
c
c
    return
    end
c
c
c
c   **********************************************
c           HRZTAIL
c   **********************************************
c
    subroutine hrztail(qbar,alpha,beta,iarbt,ibrbt,idrbt,coefrbt,
   x         nrbt,cfrbt,crbt)
c
c
    implicit real*8 (a-h,o-z)
    parameter ( gw = 19000, cg =27.208, btail=63.7)
    parameter (span =29.0,chord=10.937,wing=300,vtail=54.75)
    real*8 iarbt(20,6),ibrbt(20,6),idrbt(20,6),coefrbt(20,6)
    real*8 alpha,beta,crbt(6),cfrbt(6),nfrbt(6),cflbt(6)
    real*8 clbt(6)
c
c
c
    do 100 i=1,6
      crbt(i) = coefrbt(2,i)*alpha + coefrbt(3,i)*beta
   x          + coefrbt(1,i)
100 continue
c
c
c
    cfrbt(1) = crbt(1)*qbar*wing
    cfrbt(2) = crbt(2)*qbar*wing
    cfrbt(3) = crbt(3)*qbar*wing
    cfrbt(4) = crbt(4)*qbar*wing*chord
    cfrbt(5) = crbt(5)*qbar*wing*span
    cfrbt(6) = crbt(6)*qbar*wing*span
c
c
    return
    end
c
c
c
c   **********************************************
c           F3
c   **********************************************
c
    real*8 function f3(j,x,ia,ib,id,i_fcn)
    implicit real*8 (a-h,o-z)
    real*8 x(3)
    real*8 ia(20,6),ib(20,6),id(20,6),no_fcn(6)
c
    if (j.gt.100) write(6,*) '*** ERR - undeclared function for j=',j
    alpha = x(1)
    beta = x(2)
    delta = x(3)
    f3=poly(ia(j,i_fcn),alpha)
   x        *poly(ib(j,i_fcn),beta)
   x             *poly(id(j,i_fcn),delta)
    return
    end
c
c
c   ****************************************************
c           POLY
c   ****************************************************
```

```
c
c
      real*8 function poly(nfnc,x)
      implicit real*8 (a-h,o-z)
      real*8 nfnc
c
c This function returns values of the family of polynomials.
c
c  nfnc gives the power to raise x to.
c
      if(nfnc.eq.0) then
        poly=1.0
      else
        if (x.eq.0.0)  then
          poly=0.0
        else
          poly=x**nfnc
        end if
      end if
      return
      end
c
c   ****************************************************
c             AUTHOR
c   ****************************************************
c
      subroutine author(cflfl,cfrfl,cflht,cfrht,delta,
     x              pauth,rauth)
c
c
      implicit real*8 (a-h,o-z)
c
c   The purpose of this program is to calculate as a percentage
c   the pitch and roll controll authority remaining after the A/C
c   has been trimmed to achieve equilibrium.
c
c
      parameter ( gw = 19000, cg =27.208, htail=63.7)
      parameter (span =29.0,chord=10.937,wing=300,vtail=54.75)
      parameter (flmax=20,htmax=25)
      real*8 dmax1(4),dmax2(4),d1(4),d2(4),delta(4)
      real*8 cflfl(6),cfrfl(6),cflht(6),cfrht(6)
c
c
      rmag = 0
      pmag = 0
      rmxmag = 0
      pmxmag = 0
c
c
      d1(1) = cflfl(4)*delta(1)
      d1(2) = cfrfl(4)*delta(2)
      d1(3) = cflht(4)*delta(3)
      d1(4) = cfrht(4)*delta(4)
c
c
      d2(1) = cflfl(5)*delta(1)
      d2(2) = cfrfl(5)*delta(2)
      d2(3) = cflht(5)*delta(3)
      d2(4) = cfrht(5)*delta(4)
c
c
      dmax1(1) = cflfl(4)*flmax
      dmax1(2) = cfrfl(4)*flmax
      dmax1(3) = cflht(4)*htmax
      dmax1(4) = cfrht(4)*htmax
c
c
      dmax2(1) = cflfl(5)*flmax
      dmax2(2) = cfrfl(5)*flmax
      dmax2(3) = cflht(5)*htmax
      dmax2(4) = cfrht(5)*htmax
c
c
      do 100 i=1,4
        pmag = pmag + (d1(i))**2
        rmag = rmag + (d2(i))**2
        pmxmag = pmxmag + (dmax1(i))**2
        rmxmag = rmxmag + (dmax2(i))**2
100   continue
c
c
      pmag = sqrt(pmag)
      rmag = sqrt(rmag)
      pmxmag = sqrt(pmxmag)
      rmxmag = sqrt(rmxmag)
c
```

```
c
      pauth = (pmxmag - pmag)/pmxmag
      rauth = (rmxmag - rmag)/rmxmag


      return
      end
c
c
c   ****************************************
c          SVD_SOLVE
c   ****************************************
c
      Include svd_solve.for
Autrima.for
c
c   ****************************************************
c             AUTRIMA.FOR
c   ****************************************************
c   27 Sep 89 SMZ
c
      implicit real*8 (a-h,o-z)
c
c
      parameter ( gw = 19000, cg =27.208, htail=63.7)
      parameter (span =29.0,chord=10.937,wing=300,vtail=54.75)
      parameter (llemin=-2,llemax=25,lflmin=-20,lflmax=20)
      parameter (rflmin=-20,rflmax=20,lhtmin=-25,lhtmax=25)
      parameter (rhtmin=-25,rhtmax=25,rudmin=-30,rudmax=30)
      parameter (rlemin=-2.5,rlemax=25,pi=3.1415927)
      parameter (nf = 4,msize = 4)
c
c
      real*8  a(4,4),delta(4),b(4),zer(6)
      real*8  ranmin,ranmax,rud,lle,rle,rfl,lfl,lht,rht
      real*8 alpha,beta,ra,rb,rd,min(3),max(3)
      real*8 iax(20,6),ibx(20,6),idx(20,6),coefx(20,6),nofncx(6)
      real*8 ialle(20,6),iblle(20,6),idlle(20,6),coeflle(20,6)
      real*8 iarle(20,6),ibrle(20,6),idrle(20,6),coefrle(20,6)
      real*8 iarud(20,6),ibrud(20,6),idrud(20,6),coefrud(20,6)
      real*8 iarfl(20,6),ibrfl(20,6),idrfl(20,6),coefrfl(20,6)
      real*8 iarht(20,6),ibrht(20,6),idrht(20,6),coefrht(20,6)
      real*8 ialfl(20,6),iblfl(20,6),idlfl(20,6),coeflfl(20,6)
      real*8 ialht(20,6),iblht(20,6),idlht(20,6),coeflht(20,6)
      real*8 nflle(6),nfrle(6),nfrud(6),nfrfl(6),nflfl(6),cflle(6)
      real*8 nfrht(6),nflht(6),cfrle(6),cfx(6),cfrud(6),cfrfl(6)
      real*8 cflfl(6),cfrht(6),cflht(6),cft(6),ralp,phi1,phi1r
      real*8 crle(6),cx(6),crud(6),crfl(6),clle(6)
      real*8 clfl(6),crht(6),clht(6),ct(6),delmn,cdt
      real*8 inda,indb,indd
      real*8 fay,phi2,phi2r,err
c
c
      character*20  trim,phicon,drag,detail,auth
c
c
      external f3
      external poly



c   The purpose of this program is to search for trim solutions for
c   the F-16 given a rudder failure and the angle of deflection at which
c   this surface is locked into a "hardover" failure. Coeficients for the
c   computation of aerodynamic forces must be supplied as data files
c   which are called into subroutines in this program. This program assumes
c   a steady state condition of straight flight and that linear superposition
c   holds. The flight path angle is specified at zero.  Wings level flight
c   is not enforced and so in general the roll angle will have a non zero
c   value.
.
c   The trim routine uses the following control scheme to search for trim.
c   The leading edge flaps can be controlled but are limited to symetric
c   deployment, the flaperons are limited to utilization as ailerons
c   and the horizontal tail is differential so that it acts both as an
c   elevator and as an aileron.
c
c
c   The control surfaces in the delta vector are numbered as
c   follows:
c          1. Leading Edge Flaps
c          2. Aileron
c          3. Horizontal Tail Aileron
c          4. Horizontal Tail Elevator
c
c   This version of the program will write the specified information to data
c   files which can be evaluted in either SURFER or GRAPHER.
```

```
c
c
      write(6,*) '************************************'
      write(6,*) '         AUTRIMA.FOR'
      write(6,*) '************************************'
      write(6,*) ''
      write(6,*) ''
      write(6,*) 'Please enter the specified rudder defl in degs:'
      read(5,*) rud
      write(6,*) ''
      write(6,*) 'Please enter the min alpha in degs: '
      read(5,*) alpmin
      write(6,*) ''
      write(6,*) 'Please enter the max alpha in degs: '
      read(5,*) alpmax
      write(6,*) ''
      write(6,*) 'Please enter the index for alpha: '
      read(5,*) inda
      write(6,*) ''
      write(6,*) 'Please enter the min beta in degs: '
      read(5,*) betmin
      write(6,*) ''
      write(6,*) 'Please enter the max beta in degs: '
      read(5,*) betmax
      write(6,*) ''
      write(6,*) 'Please enter the index for beta: '
      read(5,*) indb
      write(6,*) ''
      write(6,*) ''
      write(6,*) ' The currently selected ranges for trim investigation
     x are as follows:'
      write(6,*) ''
      write(6,*) 'Failed surface:  Rudder'
      write(6,*) ''
      write(6,*) '   Min alpha:', alpmin
      write(6,*) '   Max alpha:', alpmax
      write(6,*) ''
      write(6,*) '    Min beta:', betmin
      write(6,*) '    Max beta:', betmax
c
c
c
      write(6,*) 'Enter a filename for trim solutions:'
      read(5,5000) trim
      open(12,file=trim,status='new')
c
c
      write(6,*) 'Enter a filename for Phi contours:'
      read(5,5000) phicon
      open(11,file=phicon,status='new')
c
c
      write(6,*) 'Enter a filename for Drag coef contours:'
      read(5,5000) drag
      open(10,file=drag,status='new')
c
c
c
      write(6,*) 'Enter a filename for Mean aileron contours:'
      read(5,5000) detail
      open(9,file=detail,status='new')
c
c
      write(6,*) 'Enter a filename control authority contours:'
      read(5,5000) auth
      open(8,file=auth,status='new')
c
      write(6,*) ''
      write(6,*) 'Opening file',trim
c
c
c  Initialize the min and max comparison vectors
c
      min(1) = llemin
      min(2) = rflmin
      min(3) = lbtmin
c
      max(1) = llemax
      max(2) = rflmax
      max(3) = lbtmax
c
c  initialize the ranges
c
      ralp =((alpmax-alpmin)/inda) + 1
      rb =((betmax-betmin)/indb) + 1
      z = 0
      t = 0
c
c
      call dynpres(mach,qbar)
      write(6,*) 'The value of the dynamic pressure is:', qbar
c
c  Call in the polynomial predictor equations for the forces
c  and moments.
c
      call fitzer(iaz,ibz,idz,coefz,nofncz)
      call fitlle(ialle,iblle,idlle,coeflle,nflle)
      call fitrle(iarle,ibrle,idrle,coefrle,nfrle)
      call fitrud(iarud,ibrud,idrud,coefrud,nfrud)
      call fitrfl(iarfl,ibrfl,idrfl,coefrfl,nfrfl)
      call fitrbt(iarbt,ibrbt,idrbt,coefrbt,nfrbt)
      call fitlfl(ialfl,iblfl,idlfl,coeflfl,nflfl)
      call fitlbt(ialbt,iblbt,idlbt,coeflbt,nflbt)
      write(6,*) 'Finished reading files'
c
c
       beta = betmin
       do 200 j = 1,rb
         alpha = alpmin
         do 300 k = 1,ralp
c
          call zero(qbar,alpha,beta,iaz,ibz,idz,coefz,nofncz,cfz,
     x          cz)
          call failed(qbar,alpha,beta,rud,iarud,ibrud,idrud,coefrud,
     x          nfrud,cfrud,crud)
c          call throttle(thtx,thtz,ctx,ctz,alpha)
c
c
      alpr = alpha*(pi/180)
      betr = beta*(pi/180)
c
c     Initial estimate for theta is alpha
c
      thtr1 = alpr
      salpha = sin(alpr)
      calpha = cos(alpr)
      sbeta = sin(betr);
      cbeta = cos(betr)
      ctht = cos(thtr1)
      talp = tan(alpr)
      tbet = tan(betr)
c
c     Calculate the aero forces in the body z and x axis respectively
c
      faz = salpha*(-1*cfz(2)) + calpha*(-1*cfz(1))
      fax = calpha*(-1*cfz(2)) - salpha*(-1*cfz(1))
      fay = (cfz(3)+cfrud(3))
c
c     Estimate Phi from side force equation
c
      fgw = fay/(-1*gw*ctht)
      if ((fgw.gt.1.0) then
        goto 600
      else if ((fgw.lt.-1.0) then
        goto 600
      else
       phi1r = asin(fgw)
      endif
c
c
50     cphi = cos(phi1r)
      ctht = cos(thtr1)
      stht = sin(thtr1)
      ftx = gw*sthet - fax
      phi1 = phi1r*(180/pi)
c
c     Construct the lefthand side of the linear problem with known
c     force and moment data.
c
c     The b vector contains the following force and moments by row
c
c     1. Normal
c     2. Pitch
c     3. Roll
c     4. Yaw
c
      b(1) = -1*gw*cphi*ctht - faz
      do 700 l = 1,3
        m = l + 3
        n = l+1
        b(n) = -1 * (cfz(m) + cfrud(m))
700   continue
c
c     Assemble the A matrix to be used in the linear problem.
```

```
c     This matrix is composed of the control derivatives of the
c     controls that will be used to effect a trim solution.
c
      call flaper(qbar,alpha,beta,iarfl,ibrfl,idrfl,coefrfl,nrfl,
     x      cfrfl,crfl)
      call hrztail(qbar,alpha,beta,iarbt,ibrbt,idrbt,coefrbt,nrbt,
     x      cfrbt,crbt)
      call flaper(qbar,alpha,beta,ialfl,ibifl,idlfl,coeflfl,nlfl,
     x      cflfl,clfl)
      call hrztail(qbar,alpha,beta,ialbt,iblbt,idlbt,coeflbt,nlbt,
     x      cflbt,clbt)
      call rlef(qbar,alpha,beta,iarle,ibrle,idrle,coefrle,
     x      nrle,cfrle,crle)
      call llef(qbar,alpha,beta,ialle,iblle,idlle,coeflle,
     x      nlle,cflle,clle)
c
c     Control derivatives are put in the body z axis
c
      a(1,1) = -1*(cflle(2)+cfrle(2))*salpha
     x      -1*(cflle(1)+cfrle(1))*calpha
      a(1,2) = -1*(cfrfl(2)-cflfl(2))*salpha
     x      -1*(cfrfl(1)-cflfl(1))*calpha
      a(1,3) = -1*(cfrbt(2)-cflbt(2))*salpha
     x      -1*(cfrbt(1)-cflbt(1))*calpha
      a(1,4) = -1*(cfrbt(2)+cflbt(2))*salpha
     x      -1*(cfrbt(1)+cflbt(1))*calpha
c
c
      do 800 l = 1,3
        m = l + 3
        n = l + 1
        a(n,1) = cflle(m)+cfrle(m)
        a(n,2) = cfrfl(m)-cflfl(m)
        a(n,3) = cfrbt(m)-cflbt(m)
        a(n,4) = cfrbt(m)+cflbt(m)
800   continue
c
c
c     Solve the linear problem which has been set up. Note
c     that the subroutine returns a different value of the a matrix.
c
         call svd_solve(a,b,delta,nf,nf,msize,msize)

c
c     Sum up side forces due to control deflections
c
      fay = (cfz(3) + cfrud(3))
      fay = fay + (delta(1) * (cflle(3)+cfrle(3)))
      fay = fay + (delta(2) * (cfrfl(3)-cflfl(3)))
      fay = fay + (delta(3) * (cfrbt(3)-cflbt(3)))
      fay = fay + (delta(4) * (cfrbt(3)+cflbt(3)))
c
c     Sum up Normal forces due to control deflections
c
      faz = salpha*(-1*cfz(2)) + calpha*(-1*cfz(1))
      a(1,1) = -1*(cflle(2)+cfrle(2))*salpha
     x      -1*(cflle(1)+cfrle(1))*calpha
      a(1,2) = -1*(cfrfl(2)-cflfl(2))*salpha
     x      -1*(cfrfl(1)-cflfl(1))*calpha
      a(1,3) = -1*(cfrbt(2)-cflbt(2))*salpha
     x      -1*(cfrbt(1)-cflbt(1))*calpha
      a(1,4) = -1*(cfrbt(2)+cflbt(2))*salpha
     x      -1*(cfrbt(1)+cflbt(1))*calpha
c
      fazt = fazt + (delta(1) * a(1,1))
      fazt = fazt + (delta(2) * a(1,2))
      fazt = fazt + (delta(3) * a(1,3))
      fazt = fazt + (delta(4) * a(1,4))
c
c     Adjust Pitch angle for the new roll angle
c
      thtadj =talp*(-1*fazt/gw)+(thet/calpha)*(-1*fay/gw)
      if (thtadj.gt.1.0) then
        goto 600
      else if (thtadj.lt.-1.0) then
        goto 600
      else
      thtr2=asin(thtadj)
      endif
      cthi = cos(thtr2)

c
c     Adjust Roll angle for new theta and control deflections
c
      fgw = fay/(-1*gw*cthi)
      if (fgw.gt.1.0) then
        goto 600
      else if (fgw.lt.-1.0) then
        goto 600
      else
      phi2r = asin(fgw)
      endif
c
c
      err1 = sqrt((phi1r - phi2r)**2)
      err2 = sqrt((thtr1 - thtr2)**2)
c     write(6,*) 'The error is:',err
c
c     Determine if new phi angle is within
c     .0001 radians of first approximation
c
      if (z.gt.21) then
        goto 525
      else if(err1.gt..0001) then
        phi1r = phi2r
        thtr1 = thtr2
        z = z + 1
        goto 50
      else if(err2.gt..0001) then
        phi1r = phi2r
        thtr1 = thtr2
        z = z + 1
        goto 50
      endif
c
c
c     Determine if the computed solution violates constraints on
c     control surface deflection limits and write the data to the
c     appropriate file.
c
         do 400 l=1,2
          if((delta(l).lt.min(l) .or. delta(l).gt.max(l)) then
            goto 500
          else
          endif
400       continue
c
c       Calculate equivalent Hrztail deflections
c       and check against constraints.
c
        rbt = delta(3) + delta(4)
        lbt = -1*(delta(3) - delta(4))
c
        if(rbt.lt.min(3) .or. rbt.gt.max(3)) then
          goto 500
        elseif(lbt.lt.min(3) .or. lbt.gt.max(3)) then
          goto 500
        else
        endif
c
450   continue
      phi2 = phi2r*(180/pi)
      theta = thtr2*(180/pi)
c
c
c

        delmn = (delta(1) + delta(2))/2
c
c
        cdt = crle(2)+cz(2)+crud(2)+crfl(2)+clle(2)+
     x      clfl(2)+crbt(2)+clbt(2)
c
c       Compute the solution area as of this pass.
c
          t = t+1
        alnarea = t*(inda*indb)
c
c     Calculate remaining pitch and roll authority
c
      call author(clfl,crfl,clbt,crbt,delta,clle,crle,
     x      pauth,rauth)
c
c
c       Write output to file for plotting in Grapher or Surfer
c
        write(12,60000) beta,alpha,rud,alnarea
        write(11,60000) beta,alpha,phi2,rud
        write(10,60000) beta,alpha,cdt,rud
        write(9,60000) beta,alpha,delmn,rud
        write(8,60000) beta,alpha,pauth,rauth
c
c

        goto 325
c
```

```
500     continue
c
c
515     continue
c       write(6,*)
'========================================
======='
c       write(6,*) '    NO SOLUTION AT THIS POINT'
c       WRITE(6,*)
'========================================
======='
c
c
c     write(6,*) ''
c     write(6,*) 'The value of alpha is:   ',alpha
c     write(6,*) 'The value of beta is:    ',beta
c     write(6,*) 'The rudder deflection is: ',rud
c     write(6,*) ''
c     WRITE (6,*) 'THE VALUE OF THE ROLL ANGLE IS:',phi1
c     write (6,*) ''
c       write(6,*)
'========================================
======='
c       write(6,*) '    DEFLECTION LIMITS EXCEEDED'
c       WRITE(6,*)
'========================================
======='
c       write(6,*) '$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$'
c       write(6,*) ''
c       write(6,*) 'LEF',delta(1)
c       write(6,*) ''
c       write(6,*) 'FA',delta(2)
c       write(6,*) ''
c       write(6,*) 'HA',delta(3)
c       write(6,*) ''
c       write(6,*) 'HE',delta(4)
c       write(6,*) ''
c       write(6,*) '$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$'
        goto 325
c
525     continue
c
c
c       write(6,*)
'========================================
======='
c       write(6,*) '   SOLUTION WILL NOT CONVERGE AT THIS
POINT'
c       WRITE(6,*)
'========================================
======='
c
        goto 325
c
600     continue
c       write(6,*) '*******************************  *******'
c       write(6,*) ' Steady state lift condition violated'
c       write(6,*) ' Selecting next alpha value'
c       write(6,*) '*************************************'
c
c
325     alpha = alpha + inda
        z = 0
c
300     continue
c
      beta = beta + indb
c
200   continue
c
c
c
100  continue
c
   write(12,*) 'The data search is complete.'
   close(12)
c
   close(11)
c
   close(10)
c
   close(9)
c
   close(8)
c
   write(6,*) 'The data search is complete.'
c
c
5000 Format(a20)
10000 Format(f5.2)
20000 Format(4(2x,f10.7))
30000 Format(4(f4.2))
40000 Format(5(f4.2))
50000 Format(4(1x,f12.4))
60000 Format(f9.5,3(1x,f9.5))
   stop
   end
c
c  *********************************************
c          LLEF
c  *********************************************
c
   subroutine llef(qbar,alpha,beta,ialle,iblle,idlle,coeflle,
   x       nlle,cflle,clle)
c
c  The purpose of this program is to calculate the control
c  derivatives for the left leading edge flaps given values
c  for q, alpha, beta.
c
c
c
c
   implicit real*8 (a-h,o-z)
   parameter ( gw = 19000, cg =27.208, htail=63.7)
   parameter (span =29.0,cbord=10.937,wing=300,vtail=54.75)
   real*8 ialle(20,6),iblle(20,6),idlle(20,6),coeflle(20,6)
   real*8 alpha,beta,x(3),clle(6),cflle(6),nlle(6)
c
c
   do 100 i=1,6
     clle(i) = coeflle(2,i)*alpha + coeflle(3,i)*beta
   x         + coeflle(1,i)
100  continue
c
c
   cflle(1) = clle(1)*qbar*wing
   cflle(2) = clle(2)*qbar*wing
   cflle(3) = clle(3)*qbar*wing
   cflle(4) = clle(4)*qbar*wing*cbord
   cflle(5) = clle(5)*qbar*wing*span
   cflle(6) = clle(6)*qbar*wing*span
c
   return
   end
c
c  *********************************************
c          RLEF
c  *********************************************
c
   subroutine rlef(qbar,alpha,beta,iarle,ibrle,idrle,coefrle,
   x       nrle,cfrle,crle)
c
c  The purpose of this program is to calculate the control
c  derivatives for the right leading edge flaps given values
c  for q, alpha, beta.
c
c
c
c
   implicit real*8 (a-h,o-z)
   parameter ( gw = 19000, cg =27.208, htail=63.7)
   parameter (span =29.0,cbord=10.937,wing=300,vtail=54.75)
   real*8 iarle(20,6),ibrle(20,6),idrle(20,6),coefrle(20,6)
   real*8 alpha,beta,x(3),crle(6),cfrle(6),nfrle(6)
c
c
   do 100 i=1,6
     crle(i) = coefrle(2,i)*alpha + coefrle(3,i)*beta
   x         + coefrle(1,i)
100  continue
c
c
   cfrle(1) = crle(1)*qbar*wing
   cfrle(2) = crle(2)*qbar*wing
   cfrle(3) = crle(3)*qbar*wing
   cfrle(4) = crle(4)*qbar*wing*cbord
   cfrle(5) = crle(5)*qbar*wing*span
   cfrle(6) = crle(6)*qbar*wing*span
c
c
   return
   end
c
c
```

```
c   ***************************************************
c          AUTHOR
c   ***************************************************
c
     subroutine author(cflfl,cfrfl,cflbt,cfrbt,delta,cflle,cfrle,
    x              pauth,rauth)
c
c
     implicit real*8 (a-h,o-z)
c
c   The purpose of this program is to calculate as a percentage
c   the pitch and roll controll authority remaining after the A/C
c   has been trimmed to achieve equilibrium.
c
c
     parameter ( gw = 19000, cg = 27.208, htail = 63.7)
     parameter (span = 29.0,chord = 10.937,wing = 300,vtail = 54.75)
     parameter (flmax = 20,htmax = 25,lefmax = 25)
     real*8 dmax1(4),dmax2(4),d1(4),d2(4),delta(4),cflle(6),cfrle(6)
     real*8 cflfl(6),cfrfl(6),cflbt(6),cfrbt(6)
c
c
     rmag = 0
     pmag = 0
     rmaxmag = 0
     pmaxmag = 0
c
c
     d1(1) = (cflle(4) + cfrle(4))*delta(1)
     d1(2) = (cfrfl(4)-cflfl(4))*delta(2)
     d1(3) = (cfrbt(4)-cflbt(4))*delta(3)
     d1(4) = (cfrbt(4) + cflbt(4))*delta(4)
c
c
     d2(1) = (cflle(5) + cfrle(5)) *delta(1)
     d2(2) = (cfrfl(5)-cflfl(5))*delta(2)
     d2(3) = (cfrbt(5)-cflbt(5))*delta(3)
     d2(4) = (cfrbt(5) + cflbt(5))*delta(4)
c
c
     dmax1(1) = (cflle(4) + cfrle(4))* lefmax
     dmax1(2) = (cfrfl(4)-cflfl(4))*flmax
     dmax1(3) = (cfrbt(4)-cflbt(4))*htmax
     dmax1(4) = (cfrbt(4) + cflbt(4))*htmax
c
c
     dmax2(1) = (cflle(5) + cfrle(5))* lefmax
     dmax2(2) = (cfrfl(5)-cflfl(5))*flmax
     dmax2(3) = (cfrbt(5)-cflbt(5))*htmax
     dmax2(4) = (cfrbt(5) + cflbt(5))*htmax
c
c
     do 100 i=1,4
        pmag = pmag + (d1(i))**2
        rmag = rmag + (d2(i))**2
        pmaxmag = pmaxmag + (dmax1(i))**2
        rmaxmag = rmaxmag + (dmax2(i))**2
100  continue
c
c
     pmag = sqrt(pmag)
     rmag = sqrt(rmag)
     pmaxmag = sqrt(pmaxmag)
     rmaxmag = sqrt(rmaxmag)
c
c
     pauth = (pmaxmag - pmag)/pmaxmag
     rauth = (rmaxmag - rmag)/rmaxmag


     return
     end
c
c
c
c   ***************************************************
c          DYNPRSS
c   ***************************************************
c
     include dynprss.for
c
c   ***************************************************
c          FIXZER
c   ***************************************************
     include fixzer.for
c
c   ***************************************************
c          FIXLLE
c   ***************************************************
     include fixlle.for
c
c   ***************************************************
c          FIXRLE
c   ***************************************************
c
     include fixrle.for
c
c   ***************************************************
c          FIXRUD
c   ***************************************************
c
     include fixrud.for
c   ***************************************************
c          FIXRFL
c   ***************************************************
c
     include fixrfl.for
c   ***************************************************
c          FIXRHT
c   ***************************************************
     include fixrht.for
c
c   ***************************************************
c          LEF
c   ***************************************************
c
     include lef.for
c
c   ***************************************************
c          ZERO
c   ***************************************************
c
     include zero.for
c
c   ***************************************************
c          FAILED
c   ***************************************************
c
     include failed.for
c
c
c   ***************************************************
c          FLAPER
c   ***************************************************
c
     include flaper.for
c
c
c   ***************************************************
c          HRZTAIL
c   ***************************************************
c
     include hrztail.for
c
c
c   ***************************************************
c          F3
c   ***************************************************
c
     include f3.for
c
c   ***************************************************
c          POLY
c   ***************************************************
c
     include poly.for
c
c
c
c   ***************************************************
c          SVD_SOLVE
c   ***************************************************
c
     include svdsolve.for

Autrimc.for
c
c   ***************************************************
c          AUTRIMC.FOR
c   ***************************************************
c   17 Oct 89 SMZ
c
     implicit real*8 (a-h,p-z)
c
```

```
c
      parameter ( gw = 19000, cg =27.208, htail=63.7)
      parameter (span =29.0,chord=10.937,wing=300,vtail=54.75)
      parameter (alt = 10000)
      parameter (llemin =-2,llemax=25,lflmin =-20,lflmax=20)
      parameter (rflmin =-20,rflmax=20,lhtmin =-25,lhtmax=25)
      parameter (rhtmin =-25,rhtmax=25,indll = 1.0,indrl = 1.0)
      parameter (rlemin =-2,rlemax=25,pi=3.1415927)
      parameter (nf = 4,msize = 4)
c
c
      real*8  a(4,4),delta(4),b(4),zer(6)
      real*8  ranmin,ranmax,rud,lle,rle,rfl,lfl,lht,rht
      real*8 alpha,beta,ra,rb,rd,min(4),max(4),delmn
      real*8 iaz(20,6),ibz(20,6),idz(20,6),coefz(20,6),nofncz(6)
      real*8 ialle(20,6),iblle(20,6),idlle(20,6),coeflle(20,6)
      real*8 iarle(20,6),ibrle(20,6),idrle(20,6),coefrle(20,6)
      real*8 iarud(20,6),ibrud(20,6),idrud(20,6),coefrud(20,6)
      real*8 iarfl(20,6),ibrfl(20,6),idrfl(20,6),coefrfl(20,6)
      real*8 iarbt(20,6),ibrbt(20,6),idrbt(20,6),coefrbt(20,6)
      real*8 ialfl(20,6),iblfl(20,6),idlfl(20,6),coeflfl(20,6)
      real*8 ialbt(20,6),iblbt(20,6),idlbt(20,6),coeflbt(20,6)
      real*8 nflle(6),nfrle(6),nfrud(6),nfrfl(6),nflfl(6),cflle(6)
      real*8 nfrbt(6),nflbt(6),cfrle(6),cfz(6),cfrud(6),cfrfl(6)
      real*8 cflfl(6),cfrbt(6),cflbt(6),cft(6),ralp,phi1,phi1r
      real*8 crle(6),cz(6),crud(6),crfl(6),clle(6),sbet,tbet,cbet
      real*8 clfl(6),crbt(6),clbt(6),ct(6),cthc,sthc,thtadj
      real*8 fay,alpmin,alpmax,betmin,betmax,alnarea,pauth,rauth
      real*8 inda,indb,indd,phi2r,err,dlef(2),albet(400,2)
c
c
      character*20  trim,notrim,phicon,drag,delail,auth
c
c
      external f3
      external poly


c  The purpose of this program is to search for trim solutions for
c  the F-16 given a failed control surface and the range over which
c  this surface is locked into a "hardover" failure.  The flight condition
c  and aircraft configuration are specified in parameter statements.
c  Coeficients for the computation of aerodynamic forces must be supplied
c  as data files which are called into subroutines in this program.
c  This program assumes a steady state condition of straight
c  flight and that linear superposition holds.  All six control surfaces
c  operate independently in this program with the lefs varying through their
c  range in one degree increments.
c  Wings level flight is not enforced in
c  this version of TRIM and so the roll angle will have a non zero value
c  The flight path angle is specified at zero.  This version of the program
c  will write the output alpha beta space where trim solutions exist to a
c  data file for use in either SURFER or GRAPHER.
c
      write(6,*) '****************************************'
      write(6,*) '        AUTRIMC.FOR'
      write(6,*) '****************************************'
      write(6,*) ''
      write(6,*) 'Please enter the specified rudder defl in degs:'
      read(5,*)  rud
      write(6,*) ''
      write(6,*) 'Please enter the min alpha in degs: '
      read(5,*) alpmin
      write(6,*) ''
      write(6,*) 'Please enter the max alpha in degs: '
      read(5,*) alpmax
      write(6,*) ''
      write(6,*) 'Please enter the index for alpha: '
      read(5,*) inda
      write(6,*) ''
      write(6,*) 'Please enter the min beta in degs: '
      read(5,*) betmin
      write(6,*) ''
      write(6,*) 'Please enter the max beta in degs: '
      read(5,*) betmax
      write(6,*) ''
      write(6,*) 'Please enter the index for beta: '
      read(5,*) indb
      write(6,*) ''
      write(6,*) ''
      write(6,*) ' The currently selected ranges for trim investigation
     x are as follows:'
      write(6,*) ''
      write(6,*) 'Failed surface:  Rudder'
      write(6,*) ''
      write(6,*) '    Min alpha:', alpmin
      write(6,*) '    Max alpha:', alpmax
      write(6,*) ''
      write(6,*) '     Min beta:', betmin
      write(6,*) '     Max beta:', betmax
c
c
c  The control surfaces in the delta vector are numbered as
c  follows:
c        1. Port Flaperon
c        2. Starboard Flaperon
c        3. Port Horizontal Tail
c        4. Starboard Horizontal Tail
c
c
      write(6,*) 'Enter a filename for trim solutions:'
      read(5,5000) trim
      open(12,file=trim,status='new')
c
c
c     write(6,*) 'Enter a filename for Phi contours:'
c     read(5,5000) phicon
c     open(11,file=phicon,status='new')
c
c
c     write(6,*) 'Enter a filename for Drag coef contours:'
c     read(5,5000) drag
c     open(10,file=drag,status='new')
c
c
c
c     write(6,*) 'Enter a filename for Mean aileron contours:'
c     read(5,5000) delail
c     open(9,file=delail,status='new')
c
c
c     write(6,*) 'Enter a filename for control authority contours:'
c     read(5,5000) auth
c     open(8,file=auth,status='new')
c
c     write(6,*) ''
c     write(6,*) 'Opening file',trim
c
c  Initialize the min and max comparison vectors
c
      min(1) = lflmin
      min(2) = rflmin
      min(3) = lhtmin
      min(4) = rhtmin
c
      max(1) = lflmax
      max(2) = rflmax
      max(3) = lhtmax
      max(4) = rhtmax
c
c  initialize the ranges
c
c
      ralp =((alpmax-alpmin)/inda) + 1
      rb =((betmax-betmin)/indb) + 1
      rrle = ((rlemax - rlemin)/indrl) + 1
      rlle = ((llemax - llemin)/indll) + 1
      z = 0
      t = 0
      cv = 0
c
c
      call dynprss(mach,qbar)
      write(6,*) 'The value of the dynamic pressure is:', qbar
c
c
c  Call in the polynomial predictor equations for the forces
c  and moments.
c

      call fzmer(iaz,ibz,idz,coefz,nofncz)
      call fzlle(ialle,iblle,idlle,coeflle,nflle)
      call fzrle(iarle,ibrle,idrle,coefrle,nfrle)
      call fzrud(iarud,ibrud,idrud,coefrud,nfrud)
      call fzrfl(iarfl,ibrfl,idrfl,coefrfl,nfrfl)
      call fzrbt(iarbt,ibrbt,idrbt,coefrbt,nfrbt)
      call fzlfl(ialfl,iblfl,idlfl,coeflfl,nflfl)
      call fzlbt(ialbt,iblbt,idlbt,coeflbt,nflbt)
      write(6,*) 'finished reading files'
c
c
```

```
      llef = llemin
      do 20 o = 1,rlle
      rlef = rlemin
      do 100 i = 1,rrle
      beta = betmin
      do 200 j=1,rb
        alpha = alpmin
        do 300 k=1,ralp
c
c         Assign Leading Edge Flap deflections
c
         dlef(1) = llef
         dlef(2) = rlef
c
         call lef(qbar,alpha,lle,rle,cflle,cfrle,ialle,iblle,idlle,
     x   beta,coeflle,nflle,iarle,ibrle,idrle,coefrle,nfrle,clle,
     x   crle,dlef)
c
         call zero(qbar,alpha,beta,iaz,ibz,idz,coefz,nofncz,cfz,cz)
c
         call failed(qbar,alpha,beta,rud,iarud,ibrud,idrud,coefrud,
     x           nfrud,cfrud,crud)
c

c     write(6,*) 'The value of alpha is:    ',alpha
c     write(6,*) 'The value of beta is:     ',beta
c     write(6,*) 'The rudder deflection is: ',rud
c
      alpr = alpha*(pi/180)
      betr = beta*(pi/180)
c
c     Specify the Flight Path angle equal to zero
c     which implies first estimate of theta is alpha
c
      thtr = alpr
      salpha = sin(alpr)
      calpha = cos(alpr)
      sbeta = sin(betr)
      cbeta = cos(betr)
      ctht = cos(thtr)
      talp = tan(alpr)
      tbet = tan(betr)
c
c     Calculate the aero forces in the body z and x axis respectively
c
      faz = salpha*(-1*(cfz(2)+cflle(2)+cfrle(2)))
     x     + calpha*(-1*(cfz(1)+cflle(1)+cfrle(1)))
      fax = calpha*(-1*(cfz(2)+cflle(2)+cfrle(2)))
     x     - salpha*(-1*(cfz(1)+cflle(1)+cfrle(1)))
      fay = (cfz(3) + cflle(3) + cfrle(3) + cfrud(3))
c
c
      fgw = fay/(-1*gw*ctht)
      if (fgw.gt.1.0) then
        goto 600
      else if (fgw.lt.-1.0) then
        goto 600
      else
       phi1r = asin(fgw)
      endif
c
c
50     cphi = cos(phi1r)
      ctht = cos(thtr1)
      stht = sin(thtr1)
      ftx = gw*stht - fax
      phi1 = phi1r*(180/pi)
c
c     Construct the lefthand side of the linear problem with known
c     force and moment data.
c
c     The b vector contains the following force and moments by row
c
c     1. Normal
c     2. Pitch
c     3. Roll
c     4. Yaw
c
      b(1) = -1*gw*cphi*ctht - faz
c     write(6,*) b(1)
      do 700 l = 1,3
        m = l + 3
        n = l+1
        b(n) = -1 * (cfz(m) + cflle(m) + cfrle(m) + cfrud(m))
700    continue
c
c     Assemble the A matrix to be used in the linear problem.
c     This matrix is composed of the control derivatives of the
c     controls that will be used to effect a trim solution.
c
       call flaper(qbar,alpha,beta,iarfl,ibrfl,idrfl,coefrfl,nrfl,
     x         cfrfl,crfl)
       call brztail(qbar,alpha,beta,iarbt,ibrbt,idrbt,coefrbt,nrbt,
     x         cfrbt,crbt)
       call flaper(qbar,alpha,beta,ialfl,iblfl,idlfl,coeflfl,nlfl,
     x         cflfl,clfl)
       call brztail(qbar,alpha,beta,ialbt,iblbt,idlbt,coeflbt,nlbt,
     x         cflbt,clbt)
c
c
      a(1,1) = -1*cflfl(2)*salpha -1*cflfl(1)*calpha
      a(1,2) = -1*cfrfl(2)*salpha -1*cfrfl(1)*calpha
      a(1,3) = -1*cflbt(2)*salpha -1*cflbt(1)*calpha
      a(1,4) = -1*cfrbt(2)*salpha -1*cfrbt(1)*calpha
c
      do 800 l = 1,3
        m = l + 3
        n = l + 1
        a(n,1) = cflfl(m)
        a(n,2) = cfrfl(m)
        a(n,3) = cflbt(m)
        a(n,4) = cfrbt(m)
800   continue
c
c
c     Solve the linear problem which has been set up.
c
         call svd_solve(a,b,delta,nf,nf,msize,msize)

c
c      Sum up side forces due to control deflections
c
      fay = (cfz(3) + cflle(3) + cfrle(3) + cfrud(3))
      fay = fay + (delta(1) * cflfl(3))
      fay = fay + (delta(2) * cfrfl(3))
      fay = fay + (delta(3) * cflbt(3))
      fay = fay + (delta(4) * cfrbt(3))
c
c      Sum up Normal forces due to control deflections
c
      fazt = salpha*(-1*(cfz(2)+cflle(2)+cfrle(2)))
     x     + calpha*(-1*(cfz(1)+cflle(1)+cfrle(1))),
c
      a(1,1) = -1*cflfl(2)*salpha -1*cflfl(1)*calpha
      a(1,2) = -1*cfrfl(2)*salpha -1*cfrfl(1)*calpha
      a(1,3) = -1*cflbt(2)*salpha -1*cflbt(1)*calpha
      a(1,4) = -1*cfrbt(2)*salpha -1*cfrbt(1)*calpha
c
      fazt = fazt + (delta(1) * a(1,1))
      fazt = fazt + (delta(2) * a(1,2))
      fazt = fazt + (delta(3) * a(1,3))
      fazt = fazt + (delta(4) * a(1,4))
c
c      Adjust Pitch angle for the new roll angle
c
      thtadj =talp*(-1*fazt/gw)+(tbet/calpha)*(-1*fay/gw)
      if (thtadj.gt.1.0) then
        goto 600
      else if (thtadj.lt.-1.0) then
        goto 600
      else
      thtr2=asin(thtadj)
      endif
      ctht = cos(thtr2)

c
c     Adjust Roll angle for new theta and control deflections
c
      fgw = fay/(-1*gw*ctht)
      if (fgw.gt.1.0) then
        goto 600
      else if (fgw.lt.-1.0) then
        goto 600
      else
       phi2r = asin(fgw)
      endif
c
c
      err1 = sqrt((phi1r - phi2r)**2)
      err2 = sqrt((thtr1 - thtr2)**2)
c     write(6,*) 'The error is:',err
c
c     Determine if new phi angle is within
c     .0001 radians of first approximation
```

```
c
     if (z.gt.21) then
       goto 525
     else if(err1.gt..0001) then
       phi1r = phi2r
       thtr1 = thtr2
       z = z + 1
       goto 50
     else if(err2.gt..0001) then
       phi1r = phi2r
       thtr1 = thtr2
       z = z + 1
       goto 50
     endif
c
c    Determine if the computed solution violates constraints on
c    control surface deflection limits and write the data to the
c    apropriate file.
c
        do 400 l=1,4
         if(delta(l).lt.min(l) .or. delta(l).gt.max(l)) then
          goto 500
         else
         endif
400      continue
c
c
450      continue
        phi2 = phi2r*(180/pi)
        theta = thtr2*(180/pi)
c
c
c
c    Check to see if this is the first time this point has been
c    found if so go to next alpha
c
     if (cv .eq. 0.0) go to 485

     do 475 ii = 1, cv
       if (albet(ii,1).eq.alpha .and. albet(ii,2).eq.beta)
    x  goto 325
475   continue
c
c
485   cv = cv + 1
     albet(cv,1) = alpha
     albet(cv,2) = beta

c
c    Compute the solution area as of this pass.
c

        slnarea = cv*(inda*indb)

c       write(6,*) ''
c       write(6,*) 'The value of alpha is:   ',alpha
c       write(6,*) 'The value of beta is:    ',beta
c       write(6,*) 'The rudder deflection is: ',rud
c       write(6,*) ''
c       write(6,*) 'The rlef deflection is: ',rlef
c       write(6,*) ''
c       write(6,*) 'The llef deflection is: ',llef
c       write(6,*) ''
c       WRITE (6,*) 'The value of the Roll angle is:',phi2
c       write(6,*) ''
c       WRITE (6,*) 'The value of the Pitch Angle is:',theta
c       write (6,*) ''
c       write(6,*) ' A solution exists at this point:'
c       write(6,*) '$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$'
c       write(6,*) ''
c       write(6,*) 'LFL',delta(1)
c       write(6,*) ''
c       write(6,*) 'RFL',delta(2)
c       write(6,*) ''
c       write(6,*) 'LHT',delta(3)
c       write(6,*) ''
c       write(6,*) 'RHT',delta(4)
c       write(6,*) ''
c       write(6,*) '$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$'
c
c       Write output to file for plotting in Grapher or Surfer
c
        write(12,60000) beta,alpha,rud,slnarea
c       write(11,60000) beta,alpha,phi2,rud
c       write(10,60000) beta,alpha,cdt,rud
c       write(9,60000) beta,alpha,delmn,rud
c       write(8,60000) beta,alpha,pauth,rauth
c
c
c
         goto 325
c
500      continue
c
c
c
515      continue
c        write(6,*)
'==========================================
=======' 
c        write(6,*) '    NO SOLUTION AT THIS POINT'
c        WRITE(6,*)
'==========================================
=======' 
         goto325
c
c
525      continue
c
c
c        write(6,*)
'==========================================
=======' 
c        write(6,*) '  SOLUTION WILL NOT CONVERGE AT THIS
POINT'
c        WRITE(6,*)
'==========================================
=======' 
c
         goto 325
600       continue
c        write(6,*) '************************************'
c        write(6,*) ' Steady state lift condition violated'
c        write(6,*) ' Selecting next alpha value'
c        write(6,*) '************************************'
c
325       alpha = alpha + inda
c
c         Reinitialize z
c

          z = 0
c
300       continue
c
       beta = beta + indb
c
200     continue
c
       rlef = rlef + indrl
c
100  continue
c
      llef = llef + indll
c
20   continue
c
      close(12)
c     close(11)
c     close(10)
c     close(9)
c     close(8)
c
c
      write(6,*) 'The data search is complete.'
      write(6,*) 'The total solution area is:',slnarea
c
c
5000  Format(a20)
10000 Format(f5.2)
20000 Format(4(2x,f10.7))
30000 Format(4(f4.2))
40000 Format(5(f4.2))
50000 Format(4(1x,f12.4))
60000 Format(f9.5,3(1x,f9.5))
       stop
       end
c
c     ***********************************************
c              DYNPRSS
c     ***********************************************
c
      subroutine dynprss(mach,q)
c
```

```
c ************************************************
c        FIXZER
c ************************************************
c
      subroutine fixzer(iaz,ibz,idz,coefz,nofncz)
c
c
c ************************************************
c        FIXLLE
c ************************************************
c
      subroutine fixlle(ialle,iblle,idlle,coeflle,nflle)
c
c
c ************************************************
c        FIXRLE
c ************************************************
c
      subroutine fixrle(iarle,ibrle,idrle,coefrle,nfrle)

c
c ************************************************
c        FIXRUD
c ************************************************
c
      subroutine fixrud(iarud,ibrud,idrud,coefrud,nfrud)
c
c ************************************************
c        FIXRFL
c ************************************************
c
      subroutine fixrfl(iarfl,ibrfl,idrfl,coefrfl,nfrfl)
c
c
c ************************************************
c        FIXLFL
c ************************************************
c
      subroutine fixlfl(ialfl,iblfl,idlfl,coeflfl,nflfl)
c
c ************************************************
c        FIXRHT
c ************************************************
c
      subroutine fixrht(iarht,ibrht,idrht,coefrht,nfrht)
c
c
c ************************************************
c        FIXLHT
c ************************************************
c
      subroutine fixlht(ialht,iblht,idlht,coeflht,nflht)
c
c
c ************************************************
c         LEF
c ************************************************
c
      subroutine lef(qbar,alpha,lle,rle,cflle,cfrle,ialle,iblle,idlle,
     x beta,coeflle,nflle,iarle,ibrle,idrle,coefrle,nfrle,clle,
     x crle,dlef)
c
c
c ************************************************
c        ZERO
c ************************************************
c
      subroutine zero(qbar,alpha,beta,iaz,ibz,idz,coefz,nofncz,cfz,cz)
c
c
c ************************************************
c        FAILED
c ************************************************
c ************************************************
c        FLAPER
c ************************************************
c
      subroutine flaper(qbar,alpha,beta,iarfl,ibrfl,idrfl,coefrfl,
     x        nrfl,cfrfl,crfl)
c
c
c ************************************************
c        HRZTAIL
c ************************************************
c
      subroutine hrztail(qbar,alpha,beta,iarht,ibrht,idrht,coefrht,
     x        nrht,cfrht,crht)
c
c
c ************************************************
c        F3
c ************************************************
c
      real*8 function f3(j,x,ia,ib,id,i_fcn)
c
c ******************************************************
c        POLY
c ******************************************************
c
c
      real*8 function poly(nfnc,x)
c
c ********************************************
c        SVD_SOLVE
c ********************************************
c
      subroutine svd_solve(a,b,x,n,m,np,mp)
```

Bibliography

1. Eslinger, Robert A. Multivariable Control Law Design for the AFTI/F-16 With a Failed Control Surface. MS Thesis, AFIT/GE/ENG/84D-28. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, December 1984

2. Etkin, Bernard. Dynamics of Atmospheric Flight. New York: John Wiley and Sons Inc., 1972.

3. Hawkins, Max L. An Investigation of the Departure Modes of a F-4D Aircraft From a Steady Sideslip Flight Condition. MS Thesis, AFIT/GAE/MC/74-6. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, December 1974.

4. Jane's All The World's Aircraft. 1980-1981, N. Y., Macmillian Co

5. Lipschutz, Seymour. Theory and Problems of Linear Algebra. McGraw-Hill Book Company, New York, 1968.

6. McRuer, Duane, et. al. Aircraft Dyanamics and Automatic Controls. Princeton, N.J.: Princeton University Press, 1973.

7. Press, William H. et. al. Numerical Recipes. Cambridge University Press, New York, 1986.

8. Raza, Syed J. Use of the Pseudo-Inverse For Design of a Reconfigurable Flight Control System. MS Thesis, AFIT/GAE/AA/81D-23. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, December 1986.

9. Reid, J. Gary. Linear System Fundamentals. New York: McGraw-Hill Book Company, 1983.

10. Roskam, Jan. Airplane Flight Dynamics and Automatic Flight Controls. Part I. Kansas: Roskam Aviation and Engineering Corporation, 1979.

11. Rubertus, Duane P. "Self-Repairing Flight Control Systems Overview," IEEE NAECON, 1280-1286 (1983)

12. Turhal, First Lieutenant Y. Ertugrul. Turkish Air Force. Investigation of F-16 Control Failures and Optimal Setting of Functional Controls. MS Thesis, AFIT/GAE/ENG/86D-19. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, December 1986.

13. Weiss J., et al. "Investigation of an Automatic Trim Algorithm for Restructurable Aircraft Control," IEEE NAECON, 400-406 (1986)

VITA

Captain Stephen Mark Zaiser [redacted] In 1980 he matriculated into Grove City College where he majored in mechanical engineering. On 19 May 1984 he graduated Cum Laude and received a Bachelors of Science degree and also recieving his commission as a Second Lieutenant through the Reserve Officer Training Corps. His first assignment was to the Air Force Armament Laboratory, Armament Division at Eglin AFB. In the Laboratory he worked first as a Project Engineer and later a Project Manager on the Advanced Gun Technology program. While at Eglin, he also spent a year assigned to the Munitions Division Plans and Programs Office and performed a three month assignment with the 3201st ABG as the ROTC Assistant Field Training Officer responsible for planning and coordinating the ROTC Field Encampments at Eglin. In May of 1988 he reported to Wright-Patterson AFB where he was assigned to the Air Force Institute of Technology.

[redacted]

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

1a. REPORT SECURITY CLASSIFICATION: Unclassified

1b. RESTRICTIVE MARKINGS:

2a. SECURITY CLASSIFICATION AUTHORITY:

2b. DECLASSIFICATION / DOWNGRADING SCHEDULE:

3. DISTRIBUTION / AVAILABILITY OF REPORT: Approved for public release; distribution unlimited

4. PERFORMING ORGANIZATION REPORT NUMBER(S): AFIT/GAE/ENY/89D-42

5. MONITORING ORGANIZATION REPORT NUMBER(S):

6a. NAME OF PERFORMING ORGANIZATION: School of Engineering

6b. OFFICE SYMBOL (If applicable): AFIT/ENY

7a. NAME OF MONITORING ORGANIZATION:

6c. ADDRESS (City, State, and ZIP Code): Air Force Institute of Technology (AU), Wright-Patterson AFB, Ohio 45433-6583

7b. ADDRESS (City, State, and ZIP Code):

8a. NAME OF FUNDING / SPONSORING ORGANIZATION:

8b. OFFICE SYMBOL (If applicable):

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER:

8c. ADDRESS (City, State, and ZIP Code):

10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO	PROJECT NO.	TASK NO	WORK UNIT ACCESSION NO

11. TITLE (Include Security Classification): STABILITY CHARACTERISTICS OF A COMBAT AIRCRAFT WITH CONTROL SURVACE FAILURE

12. PERSONAL AUTHOR(S): Stephen M. Zaiser, Capt., USAF

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
MS Thesis	FROM ______ TO ______	November, 1989	145

16. SUPPLEMENTARY NOTATION:

17. COSATI CODES

FIELD	GROUP	SUB-GROUP
20	04	
01	04	

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
Aircraft stability
Reconfigurable Flight Control System
Flight Control Surface Failure

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

In this thesis an investigation of the stability characteristics of an aircraft which has sustained damage to a primary control surface is performed. The analysis is performed using wind tunnel data taken on an F-16 model in a test performed by Turhal. The coupled, non-linear, aircraft equilibrium equations for constant altitude, rectilinear flight are developed and analyzed to identify aerodynamic coupling with implications for an aircraft with failed control surface(s). Regions in alpha-beta space where equilibrium is obtainable are investigated to identify remaining control authority, drag characteristics, and aircraft orientation.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT: ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION: UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL: Curtis H. Spenny, PE, Assistant Professor

22b. TELEPHONE (Include Area Code): (513)255-3517

22c. OFFICE SYMBOL: AFIT/ENY

DD Form 1473, JUN 86 — Previous editions are obsolete — SECURITY CLASSIFICATION OF THIS PAGE