



Insert Running Workflow Large

GRAYSCALE NAME:



HARDCOPY NAME:

1988002266



NASA Contractor Report 3985

Improvements to the Adaptive Maneuvering Logic Program

George H. Burgin
TITAN Systems, Inc.
La Jolla, California

Prepared for
Ames Research Center
Dryden Flight Research Facility
under Contract NAS2-11421



National Aeronautics
and Space Administration

Scientific and Technical
Information Branch

1986

TABLE OF CONTENTS

	PAGE
1.0 THE OFF-LINE BASELINE AML PROGRAM	1
1.1 BACKGROUND	1
1.2 THE INITIAL VERSION OF THE AML BASELINE PROGRAM	3
1.3 REMAINING PROBLEMS WITH THE 1983 BASELINE PROGRAM	4
2.0 THE NEW FORMULATION OF THE EQUATIONS OF MOTION	6
2.1 BACKGROUND	6
2.2 PERFORMANCE MODEL VERSUS CONTROL MODEL	7
2.3 OVERVIEW OF THE NEW EQUATIONS OF MOTION	11
2.4 DEFINITIONS OF COORDINATE SYSTEMS USED IN THE NEW EQUATIONS OF MOTION	13
2.4.1 The Earth-fixed Reference System	13
2.4.2 The Wind Axes System	14
2.4.3 The No-angle-of-attack Body Axes System	14
2.4.4 The Body Axes System	15

TABLE OF CONTENTS (Cont'd.)

	PAGE
2.4.5 The Moving Maneuver Plane System	16
2.5 TRANSFORMATION MATRICES	16
2.5.1 Single Rotation Matrices	17
2.5.2 Transformation Matrix from Earth Reference to Wind Axes	19
2.5.3 Transformation from Earth Reference to No-angle-of-attack Body Reference	20
2.5.4 Transformation from Earth Reference to Body Axes System	20
2.5.5 Transformation from Earth Reference to Moving Maneuver Plane System	23
2.6 DETERMINATION OF THE DESIRED AIRCRAFT ATTITUDE	24
2.7 THE TRANSITION FROM CURRENT ATTITUDE TO THE DESIRED ATTITUDE	33
2.8 CALCULATION OF THE AERODYNAMIC FORCES	39
3.0 A GENERIC AIRCRAFT MODEL SUITABLE FOR COMBAT SIMULATION	41
3.1 THE BASIC GENERIC AIRCRAFT MODEL	42

TABLE OF CONTENTS (Cont'd.)

	PAGE
3.2 SECONDARY PARAMETERS OF THE GENERIC AIRCRAFT MODEL	44
4.0 DATA SETS FOR AIRCRAFT C AND AIRCRAFT A	45
4.1 DATA SET REPRESENTING THE STANDARD IACO ATMOSPHERE	46
4.2 DATA SETS REPRESENTING THE AIRCRAFT C	46
4.2.1 Tables Representing Maximum LOAD FACTOR and Sustained Load Factor (Aircraft C)	46
4.2.2 Tables Representing THRUST Data for one Engine (Aircraft C)	46
4.2.3 ANGLE OF ATTACK as Function of C_L and Mach (Aircraft C)	47
4.2.4 Coefficient of DRAG as Function of Coefficient of Lift (Aircraft C)	47
4.2.5 DIVE RECOVERY ANGLE as Function of Altitude and Mach Number (Aircraft C)	47
4.3 DATA SETS REPRESENTING THE AIRCRAFT A	47
4.3.1 Maximum Coefficient of Lift (C_{Lmax}) as Function of Mach Number (Aircraft A)	48

TABLE OF CONTENTS (Cont'd.)

	PAGE
4.3.2 Tables of THRUST (Aircraft A)	50
4.3.3 ANGLE OF ATTACK as Function of C_L and Mach Number (Aircraft A)	50
4.3.4 C_L as Function of Angle of Attack and Mach Number (Aircraft A)	51
4.3.5 Coefficient of Drag as Function of Coefficient of Lift and Mach Number (Aircraft A)	51
4.3.6 Coefficient of Lift as Function of Coefficient of Drag (Aircraft A)	51
4.3.7 Dive Recovery Angle as Function of Mach and Altitude (Aircraft A)	52
4.3.8 Basic Physical Parameters of the Aircraft A as Used in the AML Program	53
4.3.9 Basic Physical Parameters of the Aircraft C as used in the AML Program	53
5.0 SUMMARY, CONCLUSIONS AND RECOMMENDATIONS	54
6.0 REFERENCES	56
TABLES	57

TABLE OF CONTENTS (Cont'd.)

- APPENDIX A: LISTING OF THE 1983 BASELINE ADAPTIVE MANEUVERING
LOGIC PROGRAM
- APPENDIX B: LISTINGS OF THE FORTRAN PROGRAMS TO PREPARE
AERODYNAMIC TABLES FOR AIRCRAFT A
- APPENDIX C: AERODYNAMICS AND PERFORMANCE DATA FOR AIRCRAFT B
AS USED IN THE AML PROGRAM

LIST OF TABLES

		PAGE
4.1-1	SPEED OF SOUND AND AIR DENSITY FOR STANDARD ATMOSPHERE	57
4.2.1-1	MAXIMUM LOAD FACTOR FOR AIRCRAFT C	58
4.2.1-2	SUSTAINED LOAD FACTOR FOR AIRCRAFT C	58
4.2.2-1	IDLE THRUST FOR ONE AIRCRAFT C ENGINE	58
4.2.2-2	MILITARY THRUST FOR ONE AIRCRAFT C ENGINE	59
4.2.2-3	AFTERBURNER THRUST FOR ONE AIRCRAFT C ENGINE	59
4.2.3-1	ANGLE OF ATTACK AS FUNCTION OF C_L AND MACH NUMBER FOR AIRCRAFT C	60
4.2.4-1	COEFFICIENT OF DRAG AS FUNCTION OF COEFFICIENT OF LIFT AND MACH NUMBER FOR AIRCRAFT C	60
4.2.5-1	DIVE RECOVERY ANGLE AS FUNCTION OF MACH NUMBER AND ALTITUDE FOR AIRCRAFT C	61
4.2.6-1	DYNAMIC PRESSURE AS FUNCTION OF ALTITUDE AND MACH NUMBER	61
4.3.1-1	COEFFICIENT OF LIFT AS FUNCTION OF MACH NUMBERS AND ANGLE OF ATTACK FOR AIRCRAFT A	62

LIST OF TABLES (Cont'd.)

	PAGE
4.3.1-2 C_{Lmax} AS A FUNCTION OF MACH NUMBER FOR THE AIRCRAFT A	63
4.3.2-1 IDLE THRUST AS FUNCTION OF MACH NUMBER AND ALTITUDE FOR AIRCRAFT A	64
4.3.2-2 MILITARY THRUST AS FUNCTION OF MACH NUMBER AND ALTITUDE FOR AIRCRAFT A	64
4.3.2-3 AFTERBURNER THRUST AS FUNCTION OF MACH NUMBER AND ALTITUDE FOR AIRCRAFT A	65
4.3.3-1 ANGLE OF ATTACK AS FUNCTION OF C_L AND MACH NUMBER FOR AIRCRAFT A	65
4.3.4-1 C_L AS FUNCTION OF MACH NUMBER AND ANGLE OF ATTACK FOR AIRCRAFT A	66
4.3.5-1 COEFFICIENT OF DRAG AS FUNCTION OF COEFFICIENT OF LIFT AND MACH NUMBER FOR THE AIRCRAFT A	66
4.3.6-1 COEFFICIENT OF LIFT AS FUNCTION OF COEFFICIENT OF DRAG AND MACH NUMBER FOR AIRCRAFT A	67
4.3.6-2 DIVE RECOVERY ANGLE AS FUNCTION OF ALTITUDE AND MACH NUMBER FOR AIRCRAFT A	68

1.0 THE OFF-LINE BASELINE AML PROGRAM

1.1 BACKGROUND

The first item in the Statement of Work for this contract called for an off-line baseline Adaptive Maneuvering Logic (AML) program which exhibits the highest possible performance in one-on-one combat, provided that the AML algorithm has complete and accurate information about the opponent's present and past states.

At the beginning of the contract, two candidate programs existed from which this baseline AML program could be derived:

- (1) The TITAN Systems, Inc. (Decision Science Division) in-house model. This version of AML was an improved version of the program described in References 1 and 2. (DSI Model)
- (2) The AML program as submitted to COSMIC by the NASA Langley Research Center (LRC). (COSMIC Model)

The decision as to which one of the two candidate programs to choose was a difficult one and was made after considerable effort to evaluate both programs.

TITAN's program had two significant advantages over the COSMIC version. First, TITAN personnel were thoroughly familiar with the Fortran code, to the point where the meaning of each variable was clear. Second, the documentation as provided in References 1 and 2 was relatively up to date and reflected the

code.

COSMIC's program, on the other hand, had the very significant advantage that it incorporated several years of changes made by LRC personnel based on experience in real-time engagements flown between the AML and experienced fighter pilots on the DMS. A definite disadvantage of the COSMIC version was its unsatisfactory documentation. Reference 3 simply described, in rather general terms, some of the improvements made to the DSI Model.

Both programs, from a computer programmer's point of view, were a minor disaster. They both evolved over many years, being modified by different programmers, often on a temporary, trial basis. If it turned out that the attempted changes were successful, the code was left in its "temporary" state. For example, large portions of the subroutine TRYNEXT in the COSMIC version will never be executed.

The conclusions of the evaluation of the two models were:

- (1) It is easier to incorporate some of the LRC-provided improvements into the DSI version than vice versa.
- (2) The improved dynamics as provided by the equations of motion of the COSMIC version are still unsatisfactory. To have AML represent an aircraft realistically under all flight conditions, a major revision of the approach to the equations of motion is necessary.

As a consequence of the above conclusions, the major thrust under this contract was to find an approach to the equations of

motion for a performance AML model which will adequately and realistically determine the motion of the AML-driven aircraft.

1.2 THE INITIAL VERSION OF THE AML BASELINE PROGRAM

The original version of the AML was written in extended Fortran IV, suitable for operation on the CDC 6000 Series computer. Subsequent AML versions ran on various CDC Cyber computers. Since the intent of the study was to advance the state-of-the-art of the AML program to run eventually on an airborne minicomputer, the baseline program was cleaned up so that variable names do not exceed six characters in length. Most of the programs written under this contract were developed on Digital Equipment Corporation VAX computers, both on VAX 11/780 and 11/750.

Appendix A to this report contains a listing of the source program of the AML Baseline Program. The major difference to the 1975 version of the program (the end product delivered to NASA LRC under Contract NAS1-9115) lies in a correction of the integration of the quaternions. In the original version, only one subroutine, QUAT, was used by both the attacker and the target. QUAT contained some local variables which, when used by both opponents, were set to the wrong values when QUAT was called.

A second important change concerned the times when the attacker and the target perform their decision process. In the version delivered to LRC in 1975, this time was deliberately selected slightly different for the attacker and for the target. The disadvantage of such a difference in maneuver timing is that under symmetrical initial conditions and under identical tactical decision parameters, the engagement between target and attacker would not result in a perfectly symmetrical outcome.

As a matter of fact, once the symmetry between the two opponents is disturbed, the engagements will become totally asymmetrical. For program verification and validation, it is very useful if one can simulate engagements for which it is known that the outcomes are symmetrical. This can become a necessary (though not a sufficient) condition for program correctness. After the above two corrections were made to the old version of the AML program, symmetry in the geometry of the engagement was maintained. (As a side remark, the COSMIC version of the AML would never maintain symmetry longer than for a few seconds of simulated engagement time, a fact which clearly points to some deficiencies in the COSMIC version.)

1.3 REMAINING PROBLEMS WITH THE 1983 BASELINE PROGRAM

It was stated earlier in this report that the intent--- after the decision to use the DSI program as baseline was reached---was to incorporate the significant improvements made by LRC into the baseline version. The first question therefore to be answered is this: "What are the most significant improvements made by LRC?". The improvements can be classified into two categories:

- (1) Improvements to the tactical decision process.
- (2) Improvements to the realism of the dynamics of the AML-controlled aircraft.

After talking to various users of the AML program in a real-time environment (for example, as a training tool on a flight simulator), it became quite clear that improving the

realism of the dynamics of the AML-driven aircraft have higher priorities than improvements of the tactical decision process.

There are two problems concerning the realism of the dynamics:

- (a) Jerky or erratic motion of aircraft when executing maneuvers with large changes in angle of attack
- (b) The difficulty in properly executing a vertical loop (Hankins called this the "over-the-top" problem, Reference 3, page 28)

NASA LRC corrected the two problems more or less independently. The first problem was alleviated by introducing logic for filtering the lift coefficient; the second problem, however, was hardly "solved." In Hankins' words, "In an effort to reduce the problem, a 'fix' was devised for the most severe cases." Most of the effort spent on the equations of motion under this contract attempted to really solve the over-the-top problem, not just to "fix" it for the most severe cases. By dropping some of the assumptions made in the original analysis, and by referencing certain geometrical conditions to the momentary body axes rather than the inertial reference system, we feel that the newly developed solution, which is described in detail in the section "The New Equations of Motion," completely eliminates the previously encountered "over-the-top" problem. It should also be noted here that the over-the-top problem presented much more than just an unrealistic dynamic behavior of the AML-driven aircraft, it severely impaired its tactical behavior, too.

2.0 THE NEW FORMULATION OF THE EQUATIONS OF MOTION

2.1 BACKGROUND

To fully understand the problems encountered with the old AML program (jerkyness of motion and the over-the-top problem) requires a thorough familiarity with the original treatment of the equations of motion. The reader of this report is therefore urged to first study pages 19 through 26 of Reference 2.

To appreciate the need for a new formulation and to justify the expended effort in man-hours and in computer time, the reader is urged to carefully read the section "Over-the-Top-Problem" in Reference 3, pages 28 to 32. To quote a key sentence from Hankins' report:

Yet it is believed that further improvement can and should be made by those who use the program in the future.

The work performed under this contract, we believe, has made these significant further improvements.

2.2 PERFORMANCE MODEL VERSUS CONTROL MODEL

Before explaining the improvements made to the equations of motion, it seems appropriate to clearly explain the source of the problem. To begin with, it is important to fully understand the difference between the performance model and the control model.

In the beginning, there was only a performance model. This was still true at the time References 1 and 2 were published. The important characteristics of the performance model are low computational requirements in "solving" the equations of motion. There are two reasons for this requirement: (1) to be able to perform, concurrently with a number of other tasks, in real time, the solution of the equations of motion; and, (2) to limit the required aerodynamic data of the AML-driven aircraft to an absolute minimum consistent with the performance of a specific fighter aircraft. This is why a deviation from the "classical" approach from the equations of motion for fixed-wing aircraft had to be made.

The classical approach to model aircraft for use in flight simulators is as follows: Define the state of the aircraft. This includes the momentary deflection angle of all control surfaces as well as the momentary orientation and magnitude of the velocity vector with respect to the surrounding air, which then in turn determines the aircraft's angle of attack and sideslip angle. Knowing \vec{v} , α , β , and the dynamic pressure, and being given the individual control surface deflection, and being provided with a set of stability and control derivatives consistent with the given flight conditions permits the calculation of the aerodynamic moment and aerodynamic force, expressed, for example, about the center of mass of the aircraft. Resolving the moments and forces into components in a

suitable Cartesian coordinate system then allows the formulation of the well known Euler's equation of motion of a rigid body with six degrees of freedom. On a flight simulator, these differential equations of motion are then integrated by some numerical method.

Given the control surface deflections, the above method is certainly quite straightforward. Admittedly, it requires a great deal of aerodynamic data (stability and control derivatives) and a substantial amount of computation time. The real reason why this approach is not the preferred way of determining the motion of the aircraft in the AML program, however, is not the requirement of data and computer time, but rather the requirement to know the control surface deflections.

The way the AML program operates is to essentially command bank angles and angles of attack to the aircraft. The basic idea of the performance model is to "move"---hopefully realistically---the aircraft so that it exhibits the desired bank angle (classical Euler angle ϕ measured in an inertial reference frame), and the desired angle of attack (with respect to the momentary velocity vector), and has a vanishing sideslip angle. Note that in this process, no moments are calculated and, therefore, the rotational differential equations of motion are not used. The number of degrees of freedom for the aircraft is five. The reason for the number being five, and not six, is the fact that the aircraft is always assumed to be in an attitude without sideslip; in other words, one degree of freedom is removed from the aircraft.

In the following discussion, the approach which uses control surface deflections and calculates moments will be called the six-degree-of-freedom method; the AML method will be called the five-degree-of-freedom method. At first glance, an

approach where no moments have to be calculated and where no differential equations for the rotational motion have to be solved might appear simpler than a true six-degree-of-freedom formulation including moments. Unfortunately, the contrary is true. The determination of the attitude of the AML-driven aircraft, given only a desired bank angle and angle of attack, is more difficult than in the full six-degree-of-freedom case. The difficulty is caused by the requirement of realistic motion. The original AML program made a number of simplifying assumptions and operations rules when moving the aircraft from some given attitude into some desired attitude. If, at a decision point, the rotation angle of the maneuver plane changed, the aircraft would enter a "transition mode." This transition mode lasted a predetermined number of integration steps. This number was a function of the required total angular change $\Delta\rho$ of the maneuver plane angle. Special rules for the motion during the transition mode applied. Once this transition mode was completed, the aircraft would move according to the rules of the "maneuver mode." Some of the rules for the transition mode were somewhat unrealistic (for example, the assumption that the magnitude of the lift remains unchanged during the transition mode) and were responsible for some of the jerkyness of the motion of the AML-driven aircraft. One of the design goals of the new equations of motion was to eliminate the distinction between transition mode and maneuver mode, and to have the magnitude of the lift always correspond to the momentary angle of attack, and to have the angle of attack change consistent with realistic body rotational rates.

The previous paragraphs explained some of the key features of the very first version of AML, the version where the aircraft was modeled as a five-degree-of-freedom rigid body. A later version of the AML program actually translated the AML-issued commands of bank angle and angle of attack into control surface

deflections such that the desired angles would be achieved in almost minimum time. The problem with such an AML version lies in the fact that for each particular aircraft type, a flight control system similar to an autopilot has to be designed which, from commanded angles, determines appropriate control surface deflections. This is by no means a trivial task, and Reference 4 describes such a control system for an F-4. An AML program using such a control system is called AML Control Model. Reference 3 compares the performance of these two different AML versions in simulated engagements between experienced fighter pilots.

Summarized, the advantages and disadvantages of the AML performance model and the AML control model are:

Performance Model

Advantages: No aerodynamic moments must be calculated and, therefore, control and stability derivative requirements are a minimum. Computation time to solve equations of motion is very small.

Disadvantages: Realistic movement of aircraft not easily accomplished. Pilot acceptance is not as good as of control model.

Control Model

Advantages: Aircraft moves with exactly the same realism as human-piloted aircraft on the flight simulator; therefore, pilot acceptance very good.

Disadvantages: Requires extensive aerodynamic data base for each specific aircraft. Requires a control system which will determine control surface deflections to achieve desired bank angle and angle of attack for flight in a specified maneuver plane.

To conclude this comparison between the AML performance model and the AML control model, it can be stated that:

- For real-time implementation on minicomputer-based flight simulators it is almost mandatory to use an AML performance model.
- To implement a large variety of aircraft types, where it may be difficult to get detailed control and stability derivative information, the AML performance model is preferred.
- If an AML performance model is selected, every effort must be made to make the aircraft motion look realistic.
- An AML performance model has a much higher chance to accommodate a "generic" aircraft model than an AML control model.

2.3 OVERVIEW OF THE NEW EQUATIONS OF MOTION

Before presenting the details about the new equations of motion, it might be helpful to clearly state their purpose and the interaction with AML's tactical decision logic. Under

normal conditions, the subroutine REACTT determines every second that a decision about the maneuver for the next time interval has to be made. The outcome of such a decision is described in terms of a maneuver plane (the desired flight path plane), the desired load factor for flight in this plane, and a desired throttle setting.

Given this command of the new maneuver and given the present status of the aircraft, the responsibility of the equations of motion is twofold:

- (1) The first task consists in determining what the aircraft's attitude should be to fly in the commanded maneuver plane with the commanded load factor.

- (2) Once it is known what the desired attitude should be, the second task is to determine a suitable set of body rotations which will move the aircraft from its present attitude into the desired attitude. The crucial point in this second step is not to exceed any of the aircraft's performance capabilities during this transition, most importantly, any of the aircraft's body rotational rates, p , q , and r . The most recent displays on a real-time flight simulator at Northrop Corporation, Hawthorne, seem to indicate that in order to achieve an acceptable realism, not only p , q , and r have to be limited to physically realizable values, but also their derivatives, \dot{p} , \dot{q} , and \dot{r} .

2.4 DEFINITIONS OF COORDINATE SYSTEMS USED IN THE NEW EQUATIONS OF MOTION

In deriving the desired attitude of the aircraft for flight in a commanded maneuver plane, five reference systems (all right-handed Cartesian coordinate systems) are used:

The Earth-fixed Reference System (Subscript e)

The Wind Axes System (Subscript w)

The No-angle-of-attack Body Axes System (Subscript bo)

The Body Axes System (Subscript b)

The Moving Maneuver Plane System (Subscript mmp)

2.4.1 The Earth-fixed Reference System (Subscript e)

The earth is assumed flat and for the purpose of air-to-air combat simulation can serve as an inertial reference system.

x_e pointing North

y_e pointing East

z_e pointing towards center of earth ("down")

2.4.2 The Wind Axes System (Subscript w)

The wind axes are defined as follows: The wind x-axis is aligned with the aircraft's velocity vector. The wind y-axis is assumed to be parallel to the x_e - y_e plane ("horizontal") and the wind-z axis completes a right-handed coordinate system.

To obtain the direction of the wind y-axis, intersect a plane normal to the velocity vector with the x_e - y_e plane, now rotate the x-wind axis clockwise in the plane defined by the velocity vector and the line of intersection. This will yield the positive y_w axis. In the special case where the velocity vector is perpendicular to the x_e - y_e plane (a vertical velocity vector), the wind y-axis is undefined.

Note that some authors use the name "wind axes" for an axes system where there are three rotations involved between inertial and wind axes. For the purpose of this report, the wind axes system has no "roll angle" with respect to the earth-fixed axes system.

2.4.3 The No-angle-of-attack Body Axes System (Subscript bo)

The bo system represents the aircraft in an attitude where the aircraft is pitched and yawed such that its

longitudinal axis aligns with the velocity vector and the airplane is then rolled about the x_{b_0} -axis. That means, the orientation of the bo axes system is defined by three rotations from the earth-fixed axis system; the three angles being $\bar{\psi}$, $\bar{\theta}$, and ϕ .

It is very important to realize that the angle ϕ is not equal to the conventional aircraft Euler angle Φ . The two angles ϕ and Φ are only equal for a no-angle-of-attack, no-sideslip flight condition.

The Euler angle Φ is the third rotation angle which brings the aircraft into its final attitude (including an angle α and possibly an angle β).

The angle ϕ is the third rotation angle if the aircraft undergoes four rotations to get into its final attitude with a finite, not necessarily small, angle of attack, but with no sideslip angle.

2.4.4 The Body Axes System (Subscript b)

This is the conventional body axes system, with the x_b -axis aligned with the aircraft longitudinal axis, the y_b -axis in the direction of the right wing and the z_b -axis completing a right-handed coordinate system. The orientation of the body axis system is obtained by the three conventional rotations Ψ , Θ , and Φ .

2.4.5 The Moving Maneuver Plane System (Subscript mmp)

The moving maneuver plane axes system is obtained by one rotation about the wind x_w -axis by the rotation angle ρ . The $y_{mmp}-(-z_{mmp})$ -plane defines the desired flight path plane.

A characteristic property of the mmp-axes system is that its x-axis is at all times aligned with the velocity vector, and its $y-z_{mmp}$ plane always contains some reference point on the desired flight path. This reference point, between maneuver decisions, is a fixed point in the inertial reference frame. One of the important features of the new equations of motion is the fact that at each integration step, the rotation angle ρ of the moving maneuver plane is recalculated. This is done as follows: Take a point p_{fix} (fixed in inertial space), located on the desired flight path at some time in the future. (Fortran name for this reference point: XFIXMMP, etc., for p_{fix} expressed in maneuver plane coordinates; XFIXE, etc., for p_{fix} expressed in the inertial reference frame.) A typical time interval would be to select p_{fix} five seconds from the time the projected flight path is calculated at an AML decision point.

2.5 TRANSFORMATION MATRICES

For reference purposes, the transformation matrices from the coordinate system to the five reference systems defined in the previous section are presented below:

Single Rotation Matrices

Transformation Matrix from Earth Reference to Wind Axis

Transformation from Earth Reference to No-angle-of-attack Body Reference

Transformation from Earth Reference to Body Axes System

Transformation from Earth Reference to Moving Maneuver Plane System

All these transformation matrices are successive applications of the following three basic rotations about the z-, y-, and x-axis in this order (yaw, pitch, roll). Note that, by definition, a positive rotation about an axis in a right-handed coordinate system is a clockwise rotation about that axis looking in the positive direction of the axis.

2.5.1 Single Rotation Matrices

Denoting the transformation matrices with $[R_z]$, $[R_y]$, and $[R_x]$, respectively, and the corresponding rotation angles with ψ , θ , and ϕ , the three matrices are:

$$[R_z] = \begin{bmatrix} \cos \psi & \sin \psi & 0 \\ -\sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$[R_y] = \begin{bmatrix} \cos \theta & 0 & -\sin \theta \\ 0 & 1 & 0 \\ \sin \theta & 0 & \cos \theta \end{bmatrix}$$

$$[R_x] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \phi & \sin \phi \\ 0 & -\sin \phi & \cos \phi \end{bmatrix}$$

Denoting the components of a vector in the original reference system with $(x_a, y_a, z_a)^T$, then that same vector expressed in the rotated coordinate system has components (x_b, y_b, z_b) , where:

$$\begin{bmatrix} x_b \\ y_b \\ z_b \end{bmatrix} = [R] \begin{bmatrix} x_a \\ y_a \\ z_a \end{bmatrix}$$

Also note that, due to the orthonormality of [R],

$$\begin{bmatrix} x_a \\ y_a \\ z_a \end{bmatrix} = [R]^T \begin{bmatrix} x_b \\ y_b \\ z_b \end{bmatrix}$$

And, finally, a unit vector along the original x_a axis will have components $(r_{11}, r_{21}, r_{31})^T$ in the rotated system.

2.5.2 Transformation Matrix from Earth Reference to Wind Axis

In the following, the notation for transformation matrices will be as follows: First subscript: original reference frame; second subscript: rotated reference frame.

$$[R_{e,w}] = \begin{bmatrix} \cos \bar{\theta} \cos \bar{\psi} & \cos \bar{\theta} \sin \bar{\psi} & -\sin \bar{\theta} \\ -\sin \bar{\psi} & \cos \bar{\psi} & 0 \\ \cos \bar{\psi} \sin \bar{\theta} & \sin \bar{\psi} \sin \bar{\theta} & \cos \bar{\theta} \end{bmatrix}$$

2.5.3 Transformation from Earth Reference to No-angle-of-attack Body Reference

$$[R_{e,b0}] = \begin{bmatrix} \cos \bar{\theta} \cos \bar{\psi} & \cos \bar{\theta} \sin \bar{\psi} & -\sin \bar{\theta} \\ \cos \bar{\psi} \sin \bar{\theta} \sin \phi & \cos \bar{\psi} \cos \phi & \cos \bar{\theta} \sin \phi \\ -\sin \bar{\psi} \cos \phi & +\sin \bar{\psi} \sin \bar{\theta} \sin \phi & \\ \cos \bar{\psi} \sin \bar{\theta} \cos \phi & \sin \bar{\psi} \sin \bar{\theta} \cos \phi & \cos \bar{\theta} \cos \phi \\ +\sin \bar{\psi} \sin \phi & -\cos \bar{\psi} \sin \phi & \end{bmatrix}$$

2.5.4 Transformation from Earth Reference to Body Axes System (e to b)

This transformation matrix may be expressed in two different ways. First, remember that the b-reference system is obtained

by a rotation about the angle α about the b_0 -y-axis, therefore,

$$[R_{e,b}]_1 = [R_y] [R_{e,b_0}] =$$

$$= \begin{bmatrix} \cos \bar{\theta} \cos \bar{\psi} \cos \alpha & \cos \bar{\theta} \sin \bar{\psi} \cos \alpha & -\sin \bar{\theta} \cos \alpha \\ -\cos \bar{\psi} \sin \bar{\theta} \cos \phi \sin \alpha & -\sin \bar{\psi} \sin \bar{\theta} \cos \phi \sin \alpha & -\cos \bar{\theta} \cos \phi \sin \alpha \\ \sin \bar{\psi} \sin \phi \sin \alpha & +\cos \bar{\psi} \sin \phi \sin \alpha & \\ \\ \cos \bar{\psi} \sin \bar{\theta} \sin \phi & \cos \bar{\psi} \cos \phi & \cos \bar{\theta} \sin \phi \\ -\sin \bar{\psi} \cos \phi & +\sin \bar{\psi} \sin \bar{\theta} \sin \phi & \\ \\ \cos \bar{\theta} \cos \bar{\psi} \sin \alpha & \cos \bar{\theta} \sin \bar{\psi} \sin \alpha & -\sin \bar{\theta} \sin \alpha \\ +\cos \bar{\psi} \sin \bar{\theta} \cos \phi \cos \alpha & +\sin \bar{\psi} \sin \bar{\theta} \cos \phi \cos \alpha & +\cos \bar{\theta} \cos \phi \cos \alpha \\ +\sin \bar{\psi} \sin \phi \cos \alpha & -\cos \bar{\psi} \sin \phi \cos \alpha & \end{bmatrix}$$

Or, it may be expressed in terms of conventional Euler angles:

$$[R_{e,b}]_2 = \begin{bmatrix} \cos \theta \cos \psi & \cos \theta \sin \psi & -\sin \theta \\ \cos \psi \sin \theta \sin \phi & \cos \psi \cos \phi & \cos \theta \sin \phi \\ -\sin \psi \cos \phi & +\sin \psi \sin \theta \sin \phi & \\ \cos \psi \sin \theta \cos \phi & \sin \psi \sin \theta \cos \phi & \cos \theta \cos \phi \\ +\sin \psi \sin \phi & -\cos \psi \sin \phi & \end{bmatrix}$$

Since the two transformation matrices $[R_{e,b}]_1$ and $[R_{e,b}]_2$ really are two different matrices yielding the same final orientation of the body axis system, equating terms in the $[R_{e,b}]_1$ matrix with corresponding terms in the $[R_{e,b}]_2$ matrix allows us to obtain an expression for the conventional Euler angle ϕ in terms of $\bar{\theta}$, ϕ , and α :

$$\phi = \tan^{-1} \frac{\cos \bar{\theta} \sin \phi}{-\sin \bar{\theta} \sin \alpha + \cos \bar{\theta} \cos \phi \cos \alpha}$$

Hankins, in Reference 3, has also shown the inverse relationship; that is, $\phi = f(\bar{\theta}, \alpha)$ and recommended to use this relationship in the transition mode. Since the new formulation of the equations of motion does no longer

differentiate between maneuver mode and transition mode, this inverse relationship is not used here.

2.5.5 Transformation from Earth Reference to Moving Maneuver

Plane System

Since the mmp-reference system is obtained by a "roll" rotation by the angle ρ about the wind x_w -axis, it follows:

$$[D_{e,mmp}] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \rho & \sin \rho \\ 0 & -\sin \rho & \cos \rho \end{bmatrix} [D_w]$$

$$= \begin{bmatrix} \cos \bar{\theta} \cos \bar{\psi} & \cos \bar{\theta} \sin \bar{\psi} & -\sin \bar{\theta} \\ \cos \bar{\psi} \sin \bar{\theta} \sin \rho & \cos \bar{\psi} \cos \rho & \cos \bar{\theta} \sin \rho \\ -\sin \bar{\psi} \cos \rho & +\sin \bar{\psi} \sin \bar{\theta} \sin \rho & \\ \cos \bar{\psi} \sin \bar{\theta} \cos \rho & \sin \bar{\psi} \sin \bar{\theta} \cos \rho & \cos \theta \cos \rho \\ +\sin \bar{\psi} \sin \rho & -\cos \bar{\psi} \sin \rho & \end{bmatrix}$$

2.6 DETERMINATION OF THE DESIRED AIRCRAFT ATTITUDE

The following problem is considered in this section:

Consider an instance of time where the AML routine REACTT has issued a new maneuver command. That command consists of a desired magnitude and sign of the load factor, a desired throttle setting, and a desired rotation angle of the maneuver plane. Remember that since the maneuver plane contains the velocity vector, the angle ρ uniquely defines the maneuver plane.

For the following discussion, it will be assumed that the commanded (desired) load factor is sufficiently large so that flight in the desired maneuver plane is physically possible. In order to achieve flight in the maneuver plane, the vector sum of all the forces acting on the aircraft (aerodynamic, propulsion, and gravity) must have no component along the maneuver plane y_{mmp} -axis. The mechanism by which the resultant force vector is forced to lie in the maneuver plane is the roll angle. To find the aircraft's Euler roll angle ϕ required to align the force vector into the maneuver plane, it is convenient to calculate first the airplane roll angle for the no-angle-of-attack condition. If then the aircraft is rotated by α about the y_{b_0} -axis, the lift vector (the only force affected by this α -rotation) will rotate about the y_{b_0} -axis, but it will remain in the y_{mmp} -z plane.

If the b_0 -reference system is defined, the following forces are defined both in magnitude and in direction:

Thrust (T) : Magnitude determined by throttle setting.

Direction along the x_b -axis which is rotated with respect to the x_{b0} -axis by α .

Drag (D) : Magnitude determined by known aerodynamic relationships.

Direction along negative x_{b0} -axis.

Gravity (mg) : Magnitude determined by momentary aircraft weight.

Direction along $+z_e$ -axis.

Lift (L) : Magnitude given by commanded load factors.

Direction to be determined such that net force lies in y_{mmp} - z_{mmp} -plane.

To achieve the desired goal of a vanishing force component along the y_{mmp} -axis, we have to express the component along that axis of each one of the above four forces.

Since, by definition, the drag vector is directed along the negative velocity vector, and since the x_{mmp} -axis is aligned with the velocity vector, \vec{D} never has a component along the y_{mmp} -axis. Of the three remaining forces, the component of gravity can be expressed readily:

$$\vec{F}_{g,mmp} = [R_{e,mmp}] \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix}$$

where $\vec{F}_{g,mmp}$ stands for the vector representing the force due to gravity expressed in the mmp-reference frame.

We are interested in the y_{mmp} component, which is:

$$\vec{F}_{g,y,mmp} = mg \cos \bar{\theta} \sin \rho$$

To express the remaining two forces in the y_{mmp} -reference system is slightly more involved. They can be expressed easily in the b_0 system, in which they have the following components:

$$\vec{L}_{b_0} = \begin{bmatrix} 0 \\ 0 \\ -L \end{bmatrix}$$

and,

$$\vec{T}_{bo} = \begin{bmatrix} T \cos \alpha \\ 0 \\ -T \sin \alpha \end{bmatrix}$$

In order to express the two vectors \vec{L} and \vec{T} in the mmp system, we need the transformation matrix from b_0 to the mmp reference system.

Since,

$$\begin{bmatrix} x_e \\ y_e \\ z_e \end{bmatrix} = [D_{bo}]^{-1} \cdot \begin{bmatrix} x_{bo} \\ y_{bo} \\ z_{bo} \end{bmatrix}$$

It follows that:

$$\begin{bmatrix} x_{mmp} \\ y_{mmp} \\ z_{mmp} \end{bmatrix} = [D_{mmp}] \cdot [D_{bo}]^{-1} \cdot \begin{bmatrix} x_{bo} \\ y_{bo} \\ z_{bo} \end{bmatrix}$$

The lift plus thrust forces can therefore be expressed in the mmp system as:

$$\vec{F}_{L+T, mmp} = [D_{mmp}] \cdot [D_{bo}]^{-1} \cdot \begin{bmatrix} T \cos \alpha \\ 0 \\ -L - T \sin \alpha \end{bmatrix}$$

The matrix $[D_{mmp}] [D_{bo}]^{-1}$ can be simplified (because $(AB)^{-1} = B^{-1} A^{-1}$ and for orthonormal matrices: $A^{-1} = A^T$) as follows:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \rho & \sin \rho \\ 0 & -\sin \rho & \cos \rho \end{bmatrix} \cdot [D_w] \cdot \left[\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \phi & \sin \phi \\ 0 & -\sin \phi & \cos \phi \end{bmatrix} \cdot [D_w] \right]^{-1} =$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \rho & \sin \rho \\ 0 & -\sin \rho & \cos \rho \end{bmatrix} \cdot [D_w] \cdot [D_w]^{-1} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \phi & \sin \phi \\ 0 & -\sin \phi & \cos \phi \end{bmatrix} =$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \rho \cos \phi & -\cos \rho \sin \phi \\ & +\sin \rho \sin \phi & +\sin \rho \cos \phi \\ 0 & -\sin \rho \cos \phi & \sin \rho \sin \phi \\ & +\cos \rho \sin \phi & +\cos \rho \cos \phi \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos (\rho-\phi) & \sin (\rho-\phi) \\ 0 & -\sin (\rho-\phi) & \cos (\rho-\phi) \end{bmatrix}$$

The relationship between the mmp and the bo reference systems is therefore:

$$\begin{bmatrix} x_{mmp} \\ y_{mmp} \\ z_{mmp} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos (\rho-\phi) & \sin (\rho-\phi) \\ 0 & -\sin (\rho-\phi) & \cos (\rho-\phi) \end{bmatrix} \cdot \begin{bmatrix} x_{bo} \\ y_{bo} \\ z_{bo} \end{bmatrix}$$

The y-component of \vec{F}_{L+T} in the mmp system is:

$$\vec{F}_{(L+T)y, \text{mmp}} = (-L-T \sin \alpha) \sin(\rho - \phi)$$

The sum of all forces along the y_{mmp} axis must vanish, therefore:

$$mg \cos \bar{\theta} \sin \rho + (-L-T \sin \alpha) \sin(\rho - \phi) = 0$$

The above equation must be solved for the unknown bank angle ϕ :

$$\sin(\rho - \phi) = \frac{mg \cos \bar{\theta} \sin \rho}{L + T \sin \alpha}$$

$$\rho - \phi = \arcsin \frac{mg \cos \bar{\theta} \sin \rho}{L + T \sin \alpha}$$

$$\phi = - \text{arc sin } \frac{mg \cos \bar{\theta} \sin \rho}{L + T \sin \alpha}$$

This important equation determines the angle ϕ (the third rotation angle in a sequence of four rotations, $\bar{\psi}, \bar{\theta}, \phi, \alpha$) such that, for a given magnitude of the lift force and a given angle of attack and a given maneuver plane rotation angle, the resulting force lies in the maneuver plane.

It is interesting to compare the expression for the bank angle ϕ obtained here with the bank angle calculated in the original version of the AML program.

The original version calculated the angle ϕ^* as the angle between the maneuver plane and the lift vector as:

$$\phi^* = -\text{arc sin } \frac{d_{23} mg}{L} \quad (\text{Ref. 2, page 20})$$

Where, $d_{23} = \cos \bar{\theta} \sin \bar{\phi}$ in other words,

$$\phi^* = -\text{arc sin } \frac{mg \cos \bar{\theta} \sin \bar{\phi}}{L}$$

where $\bar{\phi}$ was the "roll angle of the maneuver plane system." (Ref. 2, p. 21).

The first obvious discrepancy between the old and the new version is that in the old version the term " $T \sin \alpha$ " in the denominator of the argument for the arc sin function was missing. This inaccuracy is of minor importance since $T \sin \alpha$ is usually much smaller than L .

More critical is the argument of the sine function in the numerator. The angle $\bar{\phi}$ in the old version was obtained by aligning the maneuver plane reference system y -axis with the normal to some maneuver plane, which may have been defined several integration steps earlier.

In the new version, the angle ρ is calculated every integration step and thus, the mmp system always contains the velocity vector as well as the "desired" reference point on the flight path.

The most severe problem in the old version occurred during the "transition mode," where the angle $\bar{\phi}$ was really undefined.

To summarize this section: When AML subroutine REACTT has issued a new maneuver command, the first problem to be solved is

the following:

Given:	Velocity vector	\vec{V}
	Maneuver plane rotation angle	ρ
	Load factor which implies lift magnitude	L
	Angle of attack consistent with	\vec{V} and L

Find: Aircraft attitude consistent with the above four parameters and satisfying a no-sideslip condition.

This section has shown how to calculate the required aircraft bank angle ϕ satisfying the above conditions. Equivalently, it has also shown how, once ϕ is known, the conventional Euler roll angle Φ consistent with ϕ , can be calculated.

2.7 THE TRANSITION FROM CURRENT ATTITUDE TO THE DESIRED ATTITUDE

The crucial improvements between the old version of AML and the new equations of motion lie in the mechanism by which the aircraft flies when the maneuver command has changed; that is, when a new maneuver plane rotation angle and/or a new load factor have been commanded by REACTT.

Assume that at time t_n a new maneuver command has been issued by REACTT. The actual aircraft attitude at time t_n is known. It can be specified in terms of three conventional Euler angles, ψ , θ , and ϕ .

The desired attitude, compatible with flight in the newly commanded maneuver plane with the newly commanded load factor, is also known. (The previous section showed how to derive it.) The task therefore now is to "move" the aircraft from the actual attitude into the desired attitude. For a significant change in the maneuver plane rotation angle and/or the angle of attack, such a motion has to be performed over several integration steps in order not to violate any physical constraints of the aircraft's body rotational rates and rotational accelerations.

The most important difference between the old AML version and the new one is the following:

The old version, in calculating the motion from the actual attitude into the desired attitude, referenced the Euler angles of the new, desired attitude to the earth-fixed reference system and compared the new, desired Euler angles with the actual Euler angles. Based on the differences between these Euler angles, it calculated desired "Euler angle rates"; based on these desired Euler angle rates, it calculated desired body rotational rates.

The new version, in calculating the motion from

the actual attitude into the desired attitude, references the Euler angles of the new, desired attitude to the present body axes system and "compares" the new, desired Euler angles with the actual Euler angles. But now, expressed in the present body axis system, the actual Euler angles are of course zero! The values of the desired Euler angles now are directly proportional to the desired body rotational rates.

The most important advantage of this new approach is that an attitude with $\psi = +90$ no longer presents a singularity. Since we reference the new, desired attitude to the present body axes system, it is immaterial what the present attitude is, measured in the body axis system, it is always true that:

$$\psi_{act} = 0$$

$$\theta_{act} = 0$$

$$\phi_{act} = 0$$

In order to apply the above outlined method, it is necessary to know the transformation matrix from the actual body axes system to the desired body axes system.

Take an arbitrary vector with components (x_e', y_e', z_e') in the earth-fixed reference system. This vector, expressed in the actual body axes reference system, has components:

$$\begin{bmatrix} x_{act} \\ y_{act} \\ z_{act} \end{bmatrix} = [R_{e,bact}] \cdot \begin{bmatrix} x_{e'} \\ y_{e'} \\ z_{e'} \end{bmatrix}$$

On the other hand, expressed in the desired body axis system, it has components:

$$\begin{bmatrix} x_{des} \\ y_{des} \\ z_{des} \end{bmatrix} = [R_{e,bdes}] \cdot \begin{bmatrix} x_{e'} \\ y_{e'} \\ z_{e'} \end{bmatrix}$$

Solving the first of the above two equations for $(x_{e'}, y_{e'}, z_{e'})$ and substituting into the second, one obtains:

$$\begin{bmatrix} x_{des} \\ y_{des} \\ z_{des} \end{bmatrix} = [R_{e,bdes}] [R_{e,bact}]^{-1} \cdot \begin{bmatrix} x_{act'} \\ y_{act'} \\ z_{act'} \end{bmatrix}$$

Consequently, the transformation matrix from the actual body axes system to the desired body axes system is:

$$[R_{bdes,bact}] = [R_{e,bdes}] [R_{e,bact}]^{-1}$$

Since both matrices on the right-hand side are known, the elements of this transformation matrix can be calculated numerically.

We may now express the rotation from the actual body axes system into the desired body axes system by three Euler angle rotations. The first rotation, by the Euler angle $\hat{\psi}$, is about the actual, present aircraft body z-axes. The second rotation, about the new aircraft y-axis, is by an angle $\hat{\theta}$ and the final rotation, by $\hat{\phi}$, is about the new x-axis. Therefore a three-angle rotation by $\hat{\psi}$, $\hat{\theta}$, and $\hat{\phi}$ will bring the aircraft from its present attitude into an attitude consistent with flight in the desired maneuver plane and with the desired load factor.

One of the key points of the new formulation of the equations of motion is that the above three Euler angles present body rotations about the present body axes and are therefore directly related to the desired body rotational rates p, q, and r. Assume for a moment that the aircraft, in three time increments $3 \Delta t$, would rotate as follows:

in Δt_1 : with constant $r = \hat{\psi} / \Delta t$

in Δt_2 : with constant $q = \hat{\theta} / \Delta t$

in Δt_3 : with constant $p = \hat{\phi} / \Delta t$

then, at the end of the third time interval, the aircraft would be exactly in the desired attitude for flight in the maneuver plane. Of course, the aircraft does not transit in this manner from its present attitude into the desired attitude, for two reasons: (1) the above specified values for p, q, and r might exceed physically achievable rates; and (2) the aircraft does not sequentially yaw, pitch and roll, but in general performs a rotation in which p, q, and r all may have some finite value. Mathematically speaking, consider p, q, and r as vectors of rotational rates, say \vec{w}_x , \vec{w}_y , and \vec{w}_z . Then the vector sum of these three vectors indicates the momentary axis of the rotational motion. Nevertheless, the above determined values of p, q, and r are quite sensible values to chose to rotate the aircraft during the next interval of Δt . The rotational rates still have to be limited to their maximum values; for a given set of $\hat{\psi}$, $\hat{\theta}$ and $\hat{\phi}$, the appropriate values for p, q, and r therefore will be:

$$p = \min (p_{\max}, \text{abs}(\hat{\phi} / \Delta t) * \text{sign} (1, \hat{\phi}))$$

$$q = \min (q_{\max}, \text{abs}(\hat{\theta} / \Delta t) * \text{sign} (1, \hat{\theta}))$$

$$r = \min (r_{\max}, \text{abs}(\hat{\psi} / \Delta t) * \text{sign} (1, \hat{\psi}))$$

Consider the roll rate p. If p were unconstrained, the change in roll angle during the next integration or step would be chosen to be $\hat{\phi} / \Delta t$. But p is limited to p_{\max} , so if $|\hat{\phi} / \Delta t|$ exceeds p_{\max} , we limit roll rate to p_{\max} (with the appropriate sign). If, however $|\hat{\phi} / \Delta t|$ is less than p_{\max} , then to avoid overshooting, roll rate would be selected as $\hat{\phi} / \Delta t$.

The derivation given above of how to obtain p , q , and r to be applied during the coming time interval provides rotational rates for the general situation where the aircraft's present attitude and desired new attitude are given in terms of conventional Euler angles. It is important to realize that the new equations of motion apply the above procedure at every integration step; that is, not only when a new maneuver command has been issued, but also at each step between maneuver commands. Therefore, the somewhat arbitrary distinction between a transition mode and a maneuver mode, as it was made in the old AML program, no longer exists.

2.8 CALCULATION OF THE AERODYNAMIC FORCES

The desired attitude for flight in the commanded maneuver plane was derived for a no-sideslip condition. When, during the transition from the present attitude to the new desired attitude, the aircraft rotates with the above derived values of p , q , and r , there is of course no guarantee that after an integration step the aircraft's attitude will be such that there is no sideslip. In fact, applying the above determined values of p , q , and r will in general result in an attitude with some small sideslip angle. This is of no grave consequence because we calculate new values of $\hat{\psi}$, $\hat{\theta}$, and $\hat{\phi}$ such that the aircraft

will be rotated in the desired direction at the beginning of every integration step. Yet one problem remains to be solved: Given the actual aircraft's attitude at the beginning of an integration step, and given the velocity vector (magnitude and direction) at this time, what are the magnitude and direction of the aerodynamic forces. Since, by definition, drag acts in the opposite direction of the velocity vector, there is no problem in determining the direction of the drag vector.

The direction of the lift vector: To determine exactly the direction of the lift vector under an arbitrary flight condition with angle of attack and with a sideslip angle is not a trivial aerodynamic problem. The solution to this problem depends on the geometry of the aircraft; in particular, on the geometry of the wings. Fortunately, in the AML equations of motion, where at the beginning of each integration step values for p , q , and r which hopefully drive the aircraft into a non-sideslip attitude are calculated and then applied, it is well justified to assume that the sideslip angle remains always small. Then, a very reasonable assumption for the direction of the lift is:

- (a) perpendicular to the velocity vector
- (b) perpendicular to the aircraft body y -axis.

Forming the cross product between a unit vector along the y_b -axis and a unit vector along the velocity vector (in this order!) will produce a vector pointing in the direction of positive lift. This approach is equivalent to neglecting the spanwise component of the velocity vector over the wing in the calculation of the lift. Strictly speaking, therefore, in

calculating the magnitude of the lift, one also should only consider the projection of the velocity vector onto the x_b-z_b plane. However, the difference between the magnitude of the velocity vector and its projection is less than one percent up to a sideslip angle of approximately eight degrees.

3.0 A GENERIC AIRCRAFT MODEL SUITABLE FOR AIR COMBAT SIMULATION

This section discusses some minimum requirements for a generic aircraft model as well as some of the difficulties encountered with generic (aircraft) models.

To determine what might be an appropriate generic model for the AML program, one should begin by defining what, in this context, is meant by "generic aircraft model." A generic model can mean one of two things:

- A model whose simulated behavior closely resembles the dynamic behavior of the simuland. (In this point of view, the word "generic" is more closely associated with the simuland than with the model.)
- A model whose simulated dynamic behavior can be made to closely approximate the behavior of a specific simuland. (In this point of view, the word "generic" is more closely related to the word "model" than to the

word "simuland.")

To say it differently: The first approach reflects the concept of a model of a generic simuland; whereas in the second concept we deal with a generic model capable of representing a class of simulands.

For the purpose of this contract, it is quite clear which one of the two definitions applies, since the Statement of Work states: "Define a generic aircraft suitable to represent a variety of conventional modern fighter aircraft."

The next problem to be addressed is the purpose of the AML program wherein this generic model will be used. The highest degree of fidelity as far as representing the motion of an aircraft is concerned is required where the AML program drives the projected target in a manned flight simulator. Any motion of the aircraft perceived by the human opponent as unrealistic tends to destroy the pilot's acceptance of the AML as an interactive target. Such an acceptance, however, is a prerequisite for effective pilot training.

On the other end of the spectrum are non-real-time, off-line applications which merely require the generation of reasonable aircraft trajectories, without paying much attention to the aircraft's attitude while flying along such a trajectory. Requirements for the fidelity of the aircraft attitude dynamics in this case are low.

3.1 THE BASIC GENERIC AIRCRAFT MODEL

From the section on the equations of motion, some basic properties of the aircraft model can readily be derived.

The most important characteristics describing the aircraft are:

- Weight
- Wing reference area
- Maximum achievable lift, expressed either in form of g as a function of Mach and altitude or in form of C_L as a function of Mach
- Coefficient of drag as a function of coefficient of lift
- Idle, military and full afterburner thrust as function of Mach and altitude

These variables define basic performance limits of the aircraft. This set, however, is incomplete to provide enough realism for use in close-in air-to-air combat. What is needed in addition to the above data are characteristics of the aircraft defining its capability to change the attitude. This is a key factor influencing not only pilot acceptance of the model, but also the usefulness of the aircraft model for critical maneuvering, both in air-to-air combat, but even more so in missile evasion. Here lies the difficulty and the challenge of the AML performance model: Control p , q , and r such that on one hand, no jerky, unrealistic aircraft motion results; and on the other hand, that the full capability of the aircraft represented by this generic model is utilized.

At the present time, the problem of modeling p , q , and r in

drastic attitude changes is not yet solved completely satisfactorily. Instantaneous changes in p , q , and r are applied and the maximum values for \dot{p} , \dot{q} , and \dot{r} are three constants. Future work will refine this process as follows: From complete, nonlinear aircraft models, derive the control derivatives for aileron and elevator at various flight conditions (for example, at various values of dynamic pressure). Based on these values and the respective moments of inertia of the aircraft, determine maximum rates of p , q , and r and then apply, as a step function, these values of \dot{p} , \dot{q} , and \dot{r} . This is roughly equivalent to step changes in control surface deflections.

3.2 SECONDARY PARAMETERS OF THE GENERIC AIRCRAFT MODEL

An important tactical maneuver in air-to-air combat is to decelerate the aircraft. This is achieved by maintaining some near-military power throttle setting, and applying speed brakes. The aircraft model must therefore include a model of the speed brakes which, in general, consists of a maximum allowable (or achievable) speed-brake deflection angle and the incremental coefficient of drag associated with a given speed-brake deflection.

Some modern aircraft have high lift devices over which the pilot may have control. Employment of these high lift devices in general changes the basic aerodynamic properties of the vehicle. Here lies one of the problems of generic models of aircraft: How far should one incorporate such features as speed-brake deflection limits, slats/flaps-deflection limits, variable wing geometry, and maybe other functions of specific aircraft into a generic model?

In the following sections and throughout the remainder of this report, aircraft A refers to a modern high performance aircraft, aircraft B refers to an older high performance aircraft, and aircraft C refers to an older aircraft type.

4.0 DATA SETS FOR AIRCRAFT C AND AIRCRAFT A

The formats of the data sets for the Aircraft C and the Aircraft A are slightly different which is reflected by some minor changes in the equations of motion routines for the two aircraft. The first difference lies in the representation of the maximum load factor for a given flight condition; the second difference concerns the relationship between angle of attack and coefficient of lift.

Common to both aircraft models are new interpolation (table look-up) routines. The flexibility of the AML program can be enhanced by employing more general purpose interpolation routines than those used in the original AML version. The subroutine AERF4 (see listing in Appendix A) was the primary interpolation routine for the old AML version. Whenever there is a change in the values of the independent variable(s) at which function values (dependent variables) are given, major portions of subroutine AERF4 have to be recoded. The new interpolation routines TLU (for functions of one independent variable) and TLU2 (for functions of two independent variables) allow to specify, for each function, an array of independent variable values (or, for TLU2, two arrays for the independent

variable values), and a table of corresponding function values. "Remembering" the indices of the interval(s) in which the independent variable(s) were at the last table look-up, contributes significantly to the efficiency (which is so important in the real-time applications of AML) of the interpolation routines.

4.1 DATA SET REPRESENTING THE STANDARD IACO ATMOSPHERE

Table 4.1-1 represents speed of sound and air density as function of altitude for the standard IACO atmosphere.

4.2 DATA SETS REPRESENTING THE AIRCRAFT C

All tables for the Aircraft C were provided by NASA Langley Research Center.

4.2.1 Tables Representing Maximum LOAD FACTOR and Sustained Load Factor for Aircraft C

Tables 4.2.1-1 and 4.2.1-2 represent the maximum and the sustained load factor for an Aircraft C with a weight of 40,872 lbs. and a "standard" combat missile load as function of Mach and altitude.

4.2.2 Tables Representing THRUST Data for one Engine for Aircraft C

Tables 4.2.2-1 through 4.2.2-3 represent idle, military, and afterburner thrust values for one Aircraft C engine.

4.2.3 ANGLE OF ATTACK as Function of C_L and Mach for Aircraft C

Table 4.2.3-1 shows the relationship between the coefficient of lift and the angle of attack for various Mach numbers. Note that within a given Mach number, the relationship is exactly linear! (This table was obtained from LRC.)

4.2.4 Coefficient of DRAG as Function of Coefficient of Lift for Aircraft C

Table 4.2.4-1 presents the coefficient of drag as function of the coefficient of lift for various Mach numbers for the Aircraft C.

4.2.5 DIVE RECOVERY ANGLE as Function of Altitude and Mach Numbers for the Aircraft C

Table 4.2.5-1 presents the dive recovery angle in degrees as function of altitude for various Mach numbers. The "dive recovery" angle is defined as the steepest flight path angle from which, at a given altitude and airspeed, the aircraft can achieve level flight without hitting the ground. AML assumes flat earth with altitude of zero feet everywhere.

4.3 DATA SET REPRESENTING THE AIRCRAFT A

All tables representing Aircraft A data are either copies of entire tables, copies of parts of tables, or data derived from tables provided to TITAN Systems, Inc. by NASA Dryden Flight Facility in July 1984. These source tables provide the input data for the Dryden Flight Research Facility "Linearization Program" (Program LINEAR, written by Lee Duke, et

al, for the VAX Computer at Dryden).

4.3.1 Maximum Coefficient of Lift (C_{Lmax}) as Function of Mach Number

Table 4.3.1-1 presents the coefficient of lift as provided by NASA in Table F101, formatted in a readily readable form. Tabel F101 tabulated C_L as a function of Mach number, angle of attack, and horizontal tail deflection. In the original Table F1-1, C_L is listed as:

$$\begin{aligned} C_L(i,j,k) \quad i &= 1 \dots 10 \quad (\text{varied first}) \\ & \\ & j = 1 \dots 19 \\ & \\ & k = 1 \dots 5 \quad (\text{varied last}) \end{aligned}$$

with

$$i = \text{Mach numbers (0.2, 0.4, 0.6, 0.8, 0.9, 1.0, 1.1, 1.2, 1.4, 1.6)}$$

$$j = \text{Angle of Attack (-12 in steps of 4 to +60 degrees)}$$

$$k = \text{Horizontal Tail Deflection (-25 in steps of 10 to +15 degrees)}$$

Table 4.3.1-2 tabulates the maximum coefficient of lift taken from Table F101 for a tail deflection angle of +5 degrees. Note that the Aircraft A performance is no longer defined in terms of g_{max} as function of Mach and altitude, as was the case with the Aircraft C, but rather by the value of C_{Lmax} .

This has the advantage that maximum performance is valid for various weights of the aircraft. The g_{\max} table for the Aircraft C was strictly valid only for one specific value of the weight. The representation used for the Aircraft A also eliminates the ambiguity whether or not the component of thrust in the direction normal to the velocity vector ($\text{Thrust} * \sin\alpha$) is included in g_{\max} .

Given the value for $C_{L\max}$, the equations of motion routine proceeds then as follows to calculate the max. load factor:

$$\text{LODMXB} = \text{CLM} * \text{QBARS} / \text{WEITB}$$

IF (LODMXB. GT. GMAX) THEN

$$\text{LODMXB} = \text{GMAX}$$

$$\text{CLM} = \text{LODMXB} * \text{WEITB} / \text{QBARS}$$

ENDIF

Where,

LODMXB = Maximum allowable load factor for aircraft B
(corresponds to the Aircraft C version variable
FLOADMT)

CLM = Maximum coefficient of lift as found by
interpolation from Table 4.3.1-2.

QBARS = Dynamic pressure * reference area

WEITB = Weight of aircraft B

GMAX = Maximum allowable load factor due to structural limitations

4.3.2 Tables of Thrust for Aircraft A

Tables 4.3.2-1 through 4.3.2-3 tabulate the Aircraft A engine thrust for one single engine for idle, military, and afterburner throttle setting. These thrust data are essentially copies of the data provided in the file ECD.DAT by DFRF. The table for idle thrust was expanded by one Mach number (Mach = 1.6). The idle thrust at this Mach number was set to the corresponding military thrust at the same Mach number and altitude. Note that the original tables already had the idle thrust for Mach = 1.4 set to the military thrust for Mach 1.4. This is a feature of the Aircraft A. Above Mach 1.2, the thrust cannot be reduced to lower values than military thrust.

All three tables were restricted to altitudes up to 60,000 feet in order to maintain consistency with other tables in the AML program. A limitation to let AML not engage in dogfights above 60,000 feet seems no unreasonable restriction.

4.3.3 Angle of Attack as Function of C_L and Mach Number

Table 4.3.3-1 was constructed by inverting the function $C_L = f(\alpha, \text{Mach})$. A Fortran program, CLINVR.FOR (Appendix B), calculated the inverse function of the table presented in Table 4.3.3-1 for a tail deflection of -5 degrees.

4.3.4 C_L as Function of Angle of Attack and Mach Number

Table 4.3.4-1 tabulates C_L as a function of the angle of attack and Mach number. It is extracted from F101 for a tail deflection angle of -5 degrees, truncated for angle of attack values greater than 40 degrees. Limiting the AML-controlled aircraft to angles of attack less than 40 degrees seems reasonable.

4.3.5 Coefficient of Drag as Function of Coefficient of Lift and Mach Number

Table 4.3.5-1 tabulates C_D as a function of C_L and of Mach number. These values are extracted from the DFRF Table F201. The table presented here contains only C_D values for positive values of C_L . Inspection of Table F201 reveals that the C_D values for negative C_L 's are exactly the same as for the corresponding positive values. It is therefore sufficient to tabulate any values for positive C_L 's and in the interpolation for the C_D value to use the absolute value of C_L as argument.

4.3.6 Coefficient of Lift as Function of Coefficient of Drag

Table 4.3.6-1 tabulates C_L as function of C_D and of Mach number. It therefore represents the inverse function of the preceding Table 4.3.5-1.

$C_L(C_D)$ is used to calculate the sustained load factor (which, in the former version of the AML program was given for the Aircraft C in a Table F4SG, listing sustained g as function of Mach and altitude). The same advantages as for giving C_{Lmax} rather than g_{max} apply for the calculation of g sustained using $C_L(C_D)$. The Fortran program CDINVR.FOR included in Appendix B shows how $C_L = f(C_D)$ was obtained.

4.3.7 Dive Recovery Angle as Function of Mach and Altitude

Table 4.3.7-1 tabulates dive recovery angles for Aircraft A as a function of altitude and mach numbers. Dive recovery angle is defined as the steepest dive angle at which, at a given airspeed and altitude, the aircraft can fly and still be able to "pull out" without hitting the ground. Dive recovery angles are listed in degrees.

4.3.8 Basic Physical Parameters of the Aircraft A as Used in the AML Program

Weight:			40,700 lbs
Wing Reference Area:			608 feet ²
Wing Span b:			42.8 feet
Wing Mean Chord c:			15.95 feet
Moment of Inertia:	About roll axis	I_{xx}	28,700 slugs feet ²
Moment of Inertia:	About pitch axis	I_{yy}	165,100 slugs feet ²
Moment of Inertia:	About yaw axis	I_{zz}	187,900 slugs feet ²
Product of Inertia:		I_{xz}	= -520 slugs feet ²
Products of Inertia:		I_{xy}	= I_{yz} = 0

4.3.9 Basic Physical Parameters of Aircraft C as used in the AML Program

Weight (50% fuel + 6 Missiles):			40,827 lbs
Wing Reference Area:			530 feet ²
Wing Span b:			38.4 feet
Wing Mean Chord c:			16.0 feet
Moment of Inertia	About roll axis	I_{xx}	26,189 slugs feet ²
Moment of Inertia	About pitch axis	I_{yy}	126,657 slugs feet ²
Moment of Inertia	About yaw axis	I_{zz}	145,007 slugs feet ²
Product of Inertia		I_{xz}	3012 slugs feet ²
Products of Inertia		I_{xy}	= I_{yz} = 0

5.0 SUMMARY, CONCLUSIONS AND RECOMMENDATIONS

This report described three improvements and extensions to the Adaptive Maneuvering Logic program.

A "baseline" AML program, which permits off-line, non-real-time air-to-air combat simulations between two opposing aircraft, wherein both interact with the other's move, was generated. This program is a minor revision of an earlier program developed by Decision Science for the NASA Langley Research Center.

The equations determining the motion of an AML-driven aircraft have been completely revised. This new version of the equations of motion make the motion of the aircraft much more realistic than in the older version and resolves any difficulties when the aircraft flies a loop in a vertical plane.

A new data set representing an Aircraft A has been prepared based on aerodynamic and propulsion data of the Aircraft A delivered to TITAN Systems, Inc. by the NASA Dryden Flight Research Facility.

The new equations of motion using the Aircraft A data set were programmed for real-time execution on a flight simulator at Northrop in Hawthorne. Preliminary results seem to indicate that the Aircraft A model flies realistic trajectories. Attitude changes still appear to be somewhat unrealistically fast.

The next tasks, whose satisfactory completion is crucial for the future of the AML program "flying" an Aircraft A, are:

Refinement of motion during attitude changes (no step changes in p , q , and r).

Validation of the Aircraft A model performance by comparing simulation results, such as vertical loops and sustained horizontal loops, with actual Aircraft A flight test data.

Refinement of the AML tactics to take full advantage of the superior performance of the Aircraft A.

6.0 REFERENCES

1. Burgin, George H., Fogel, Lawrence J. and Phelps, J. Price. An Adaptive Maneuvering Logic Computer Program for the Simulation of One-on-One Air-to-Air Combat. Vol. I: General Description. NASA Contractor Report NASA CR-2582, Washington, D.C., 1975.
2. Burgin, George H., and Owens, A.J. An Adaptive Maneuvering Logic Computer Program for the Simulation of One-on-One Air-to-Air Combat. Vol. II: Program Description. NASA Contractor Report NASA CR-2583, Washington, D.C., 1975.
3. Hankins, Walter W., III. Computer-Automated Opponent for Manned Air-to-Air Combat Simulations. NASA Technical Paper 1518, September 1979.
4. Burgin, George H. and Eggleston, David M. Design of an All-Attitude Flight Control System to Execute Commanded Bank Angles and Angles of Attack. NASA CR-145004, Contract Number NAS1-13773, February 1976.

SPEED OF SOUND AND AIR DENSITY TABLES FOR STANDARD ATMOSPHERE

SPEED OF SOUND (FEET/SEC)

TABLE HAS 121 POINTS. ALTITUDE INCREMENT IS 500 FEET. ALTITUDE REGIME IS ZERO TO 60,000 FEET.

1116.45	1114.53	1112.61	1110.68	1108.75	1106.81	1104.88	1102.94	1100.99	1099.05	0 TO 4500	FEET
1097.10	1095.14	1093.19	1091.22	1089.26	1087.29	1085.32	1083.35	1081.37	1079.39	5000 TO 9500	FEET
1077.40	1075.42	1073.42	1071.43	1069.43	1067.43	1065.42	1063.41	1061.39	1059.38	10000 TO 14500	FEET
1057.36	1055.33	1053.30	1051.27	1049.23	1047.19	1045.15	1043.10	1041.05	1038.99	15000 TO 19500	FEET
1036.93	1034.86	1032.80	1030.72	1028.65	1026.57	1024.48	1022.39	1020.30	1018.20	20000 TO 24500	FEET
1016.10	1014.00	1011.89	1009.77	1007.65	1005.53	1003.40	1001.27	999.14	996.99	25000 TO 29500	FEET
994.85	992.70	990.55	988.39	986.22	984.05	981.88	979.70	977.52	975.34	30000 TO 34500	FEET
973.14	971.83	968.74	968.08	968.08	968.08	968.08	968.08	968.08	968.08	35000 TO 39500	FEET
968.08	968.08	968.08	968.08	968.08	968.08	968.08	968.08	968.08	968.08	40000 TO 44500	FEET
968.08	968.08	968.08	968.08	968.08	968.08	968.08	968.08	968.08	968.08	45000 TO 49500	FEET
968.08	968.08	968.08	968.08	968.08	968.08	968.08	968.08	968.08	968.08	50000 TO 54500	FEET
968.08	968.08	968.08	968.08	968.08	968.08	968.08	968.08	968.08	968.08	55000 TO 59500	FEET
968.08										60000	FEET

AIR DENSITY, $\rho_{000}=1000$, SLUGS /FT CUBED, AS A FUNCTION OF ALTITUDE

TABLE HAS 121 POINTS. ALTITUDE INCREMENT IS 500 FEET. ALTITUDE REGIME IS ZERO TO 60,000 FEET.

2.37680	2.34230	2.30810	2.27430	2.24080	2.20780	2.17510	2.14280	2.11090	2.07930	0 TO 4500	FEET
2.04810	2.01730	1.98680	1.95670	1.92690	1.89750	1.86840	1.83970	1.81130	1.78320	5000 TO 9500	FEET
1.75550	1.72810	1.70110	1.67430	1.64790	1.62180	1.59610	1.57060	1.54550	1.52060	10000 TO 14500	FEET
1.49610	1.47190	1.44800	1.42440	1.40100	1.37800	1.35530	1.33280	1.31070	1.28880	15000 TO 19500	FEET
1.26720	1.24590	1.22490	1.20410	1.18370	1.16340	1.14340	1.12370	1.10430	1.08510	20000 TO 24500	FEET
1.06620	1.04750	1.02910	1.01100	0.99310	0.97540	0.95800	0.94080	0.92380	0.90710	25000 TO 29500	FEET
0.89060	0.87440	0.85840	0.84260	0.82700	0.81170	0.79650	0.78160	0.76690	0.75240	30000 TO 34500	FEET
0.73820	0.72410	0.71020	0.69440	0.67800	0.66190	0.64630	0.63100	0.61600	0.60150	35000 TO 39500	FEET
0.58720	0.57330	0.55980	0.54650	0.53360	0.52100	0.50870	0.49660	0.48490	0.47340	40000 TO 44500	FEET
0.46220	0.45130	0.44060	0.43020	0.42000	0.41010	0.40040	0.39090	0.38170	0.37270	45000 TO 49500	FEET
0.36390	0.35530	0.34690	0.33870	0.33070	0.32290	0.31520	0.30780	0.30050	0.29340	50000 TO 54500	FEET
0.28650	0.27970	0.27310	0.26660	0.26030	0.25420	0.24820	0.24230	0.23660	0.23100	55000 TO 59500	FEET
0.22560										60000	FEET

TABLE 4.1-1 SPEED OF SOUND AND AIR DENSITY FOR STANDARD ATMOSPHERE

ALTITUDE \ MACH NO	MACH NO												
	0.20	0.50	0.80	0.90	1.00	1.10	1.20	1.50	1.80	2.00	2.20	2.40	
0.	1.0	4.2	7.0	7.0	6.6	6.2	6.0	6.0	1.0	0.0	0.0	0.0	
15000.	0.5	4.0	6.0	6.8	6.3	6.3	6.3	6.0	1.0	0.0	0.0	0.0	
30000.	0.4	1.5	3.3	4.3	5.4	6.3	6.4	6.4	6.4	0.0	0.0	0.0	
45000.	0.2	0.8	1.8	2.3	2.8	3.2	3.7	4.6	4.7	5.0	0.0	0.0	
55000.	0.1	0.5	0.9	1.3	1.5	1.8	2.5	3.8	3.8	4.0	0.0	0.0	

TABLE 4.2.1-1 MAXIMUM LOAD FACTOR FOR AIRCRAFT C

ALTITUDE \ MACH NO	MACH NO												
	0.20	0.50	0.80	0.90	1.00	1.10	1.20	1.50	1.80	2.00	2.20	2.40	
0.	0.6	3.6	6.4	6.9	6.2	2.4	0.0	0.0	0.0	0.0	0.0	0.0	
15000.	0.4	2.1	4.0	4.4	4.5	4.2	3.9	0.0	0.0	0.0	0.0	0.0	
30000.	0.2	1.2	2.2	2.5	2.6	2.7	2.8	2.6	1.1	0.0	0.0	0.0	
45000.	0.1	0.6	1.1	1.2	1.3	1.4	1.4	1.5	1.5	1.0	0.0	0.0	
55000.	0.1	0.3	0.6	0.7	0.8	0.8	0.9	0.9	0.9	0.6	0.0	0.0	

TABLE 4.2.1-2 SUSTAINED LOAD FACTOR FOR AIRCRAFT C

MACH NO \ ALTITUDE	ALTITUDE							
	0.	10000.	20000.	30000.	40000.	50000.	60000.	
0.2	30.	100.	140.	175.	260.	320.	340.	
0.4	-240.	-140.	-50.	90.	220.	320.	390.	
0.6	-570.	-440.	-270.	-70.	140.	270.	350.	
0.8	-1030.	-800.	-540.	-270.	-10.	170.	280.	
0.9	-1420.	-1010.	-670.	-370.	-90.	110.	240.	
1.0	-1820.	-1200.	-820.	-480.	-170.	60.	200.	
1.1	-950.	-1450.	-950.	-600.	-250.	0.	160.	
1.2	1500.	-1650.	-1120.	-710.	-360.	-60.	120.	
1.4	3550.	2300.	-1370.	-900.	-520.	-180.	30.	
1.6	1600.	3320.	2570.	-1000.	-660.	-330.	-50.	
1.8	-1950.	-1000.	2620.	2840.	250.	0.	0.	
2.0	-2940.	-2740.	-1900.	2700.	2550.	1600.	980.	
2.2	-3000.	-3000.	-2950.	-140.	1150.	780.	480.	
2.4	-3000.	-3000.	-3000.	-1700.	-500.	-270.	-150.	

TABLE 4.2.2-1 IDLE THRUST FOR ONE ENGINE (AIRCRAFT C)

MACH NO	ALTITUDE	0.	10000.	20000.	30000.	40000.	50000.	60000.
0.2		9550.	7070.	4720.	3080.	1800.	1010.	500.
0.4		9660.	7310.	5010.	3290.	1980.	1160.	720.
0.6		9720.	7690.	5480.	3590.	2230.	1340.	800.
0.8		10190.	8240.	6110.	4120.	2580.	1550.	900.
0.9		10560.	8600.	6470.	4450.	2790.	1690.	990.
1.0		10810.	8940.	6830.	4740.	3080.	1850.	1050.
1.1		10760.	9310.	7240.	5140.	3430.	2100.	1175.
1.2		9970.	9680.	7680.	5530.	3780.	2280.	1140.
1.4		8300.	9980.	8450.	6300.	4390.	2620.	1050.
1.6		6930.	9180.	8820.	7000.	4880.	2900.	940.
1.8		5780.	8330.	9095.	6890.	5030.	2990.	800.
2.0		4630.	6480.	7370.	6780.	5180.	3080.	570.
2.2		3350.	4300.	5200.	5400.	4030.	2450.	320.
2.4		2330.	2450.	3450.	4450.	3290.	2000.	0.

TABLE 4.2.2-2 MILITARY THRUST FOR ONE ENGINE (AIRCRAFT C)

MACH NO	ALTITUDE	0.	10000.	20000.	30000.	40000.	50000.	60000.
0.2		15600.	11380.	7700.	4880.	2550.	1170.	600.
0.4		16000.	11900.	8200.	5500.	3060.	1600.	700.
0.6		16700.	13000.	9300.	6200.	3750.	2200.	1100.
0.8		18700.	14800.	11070.	7600.	4800.	2900.	1400.
0.9		20000.	15960.	12150.	8350.	5400.	3200.	1700.
1.0		21350.	17200.	13400.	9300.	5900.	3600.	2000.
1.1		19880.	18750.	14500.	10500.	6600.	4000.	2300.
1.2		19620.	20300.	15700.	11600.	7300.	4500.	2700.
1.4		19150.	21200.	18300.	13600.	9000.	5600.	3300.
1.6		18600.	20920.	20300.	15600.	10800.	6600.	4000.
1.8		17750.	19900.	21100.	17300.	12400.	7700.	4500.
2.0		16900.	18850.	19450.	18200.	13400.	8200.	5100.
2.2		13500.	15100.	16350.	17500.	12150.	7300.	5700.
2.4		12100.	12500.	14600.	17400.	11900.	7000.	6300.

TABLE 4.2.2-3 AFTER BURNER THRUST FOR ONE ENGINE (AIRCRAFT C)

LIFT COEFF	MACH NO	0.2	0.5	0.8	0.9	1.0	1.1	1.2	1.5	1.8	2.0
	0.0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.1	0.0297	0.0302	0.0295	0.0283	0.0279	0.0279	0.0279	0.0300	0.0387	0.0562	0.0794
0.2	0.0593	0.0504	0.0590	0.0565	0.0559	0.0559	0.0559	0.0600	0.0775	0.1124	0.1588
0.3	0.0890	0.0906	0.0885	0.0848	0.0838	0.0838	0.0838	0.0901	0.1162	0.1686	0.2382
0.4	0.1187	0.1208	0.1180	0.1131	0.1117	0.1117	0.1117	0.1201	0.1550	0.2248	0.3176
0.5	0.1484	0.1510	0.1475	0.1414	0.1396	0.1396	0.1396	0.1501	0.1937	0.2810	0.3971
0.6	0.1780	0.1812	0.1770	0.1696	0.1676	0.1676	0.1676	0.1801	0.2325	0.3372	0.4765
0.7	0.2077	0.2114	0.2065	0.1979	0.1955	0.1955	0.1955	0.2101	0.2712	0.3934	0.5559
0.8	0.2374	0.2416	0.2360	0.2262	0.2234	0.2234	0.2234	0.2402	0.3100	0.4496	0.6353
0.9	0.2670	0.2717	0.2655	0.2545	0.2513	0.2513	0.2513	0.2702	0.3487	0.5058	0.0000
1.0	0.2967	0.3019	0.2950	0.2827	0.2793	0.2793	0.2793	0.3002	0.3875	0.5620	0.0000
1.1	0.3264	0.3321	0.3245	0.3110	0.3072	0.3072	0.3072	0.3302	0.4262	0.6182	0.0000
1.2	0.3560	0.3623	0.3540	0.3393	0.3351	0.3351	0.3351	0.3602	0.4650	0.6744	0.0000
1.3	0.3857	0.3925	0.3834	0.3676	0.3630	0.3630	0.3630	0.3903	0.5037	0.0000	0.0000
1.4	0.4154	0.4227	0.4129	0.3958	0.3910	0.3910	0.3910	0.4203	0.5424	0.0000	0.0000
1.5	0.4451	0.4529	0.4424	0.4241	0.4189	0.4189	0.4189	0.4503	0.5812	0.0000	0.0000
1.6	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
1.7	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
1.8	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
1.9	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
2.0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

TABLE 4.2.3-1 ANGLE OF ATTACK AS FUNCTION OF C_L AND MACH NUMBER FOR AIRCRAFT C

LIFT COEFF	MACH NO	0.2	0.5	0.8	0.9	1.0	1.1	1.2	1.5	1.8	2.0
	0.0	0.0200	0.0200	0.0199	0.0226	0.0300	0.0344	0.0367	0.0388	0.0388	0.0388
0.1	0.0212	0.0212	0.0211	0.0245	0.0320	0.0369	0.0389	0.0424	0.0424	0.0424	0.0424
0.2	0.0248	0.0248	0.0249	0.0295	0.0370	0.0424	0.0454	0.0504	0.0504	0.0525	0.0539
0.3	0.0314	0.0314	0.0329	0.0359	0.0454	0.0534	0.0569	0.0654	0.0654	0.0714	0.0747
0.4	0.0415	0.0417	0.0449	0.0495	0.0610	0.0689	0.0749	0.0887	0.1004	0.1004	0.1063
0.5	0.0579	0.0587	0.0649	0.0724	0.0829	0.0919	0.1009	0.1209	0.1209	0.1409	0.1679
0.6	0.0869	0.0879	0.0949	0.1044	0.1159	0.1264	0.1359	0.1692	0.1692	0.2149	0.2479
0.7	0.1299	0.1312	0.1409	0.1519	0.1679	0.1814	0.1979	0.2434	0.2434	0.2979	0.3279
0.8	0.1815	0.1825	0.1934	0.2050	0.2310	0.2469	0.2689	0.3302	0.3302	0.3849	0.4099
0.9	0.2149	0.2159	0.2464	0.2619	0.2929	0.3109	0.3389	0.4142	0.4142	0.4689	0.4909
1.0	0.2889	0.2902	0.3009	0.3199	0.3564	0.3734	0.4089	0.4999	0.4999	0.5539	0.5739
1.1	0.3275	0.3297	0.3400	0.3645	0.4025	0.4055	0.4280	0.6040	0.6040	0.5605	0.5905
1.2	0.3805	0.3805	0.3900	0.4185	0.4635	0.4645	0.4895	0.6915	0.6915	0.6275	0.6605
1.3	0.4335	0.4312	0.4400	0.4725	0.5245	0.5235	0.5510	0.7790	0.7790	0.6945	0.7305
1.4	0.4865	0.4820	0.4900	0.5265	0.5855	0.5825	0.6125	0.8665	0.8665	0.7615	0.8005
1.5	0.5395	0.5327	0.5400	0.5805	0.6465	0.6415	0.6740	0.9540	0.9540	0.8285	0.8705
1.6	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
1.7	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
1.8	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
1.9	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
2.0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

TABLE 4.2.4-1 COEFFICIENT OF DRAG AS FUNCTION OF C_L AND MACH NUMBER FOR AIRCRAFT C

ALTITUDE \ MACH NO	MACH NO											
	0.20	0.50	0.80	0.90	1.00	1.10	1.20	1.50	1.80	2.00	2.20	2.40
0.	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
5000.	90.0000	90.0000	45.0000	30.0000	27.0000	25.0000	24.0000	19.0000	17.0000	15.0000	15.0000	15.0000
10000.	90.0000	90.0000	90.0000	60.0000	53.0000	49.0000	47.0000	37.0000	33.0000	30.0000	30.0000	30.0000
15000.	90.0000	90.0000	90.0000	90.0000	80.0000	73.0000	70.0000	55.0000	50.0000	45.0000	45.0000	45.0000
20000.	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	77.0000	66.0000	57.0000	51.0000	51.0000
25000.	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	83.0000	74.0000	63.0000	57.0000	57.0000
30000.	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	82.0000	69.0000	63.0000	63.0000	63.0000
35000.	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	75.0000	68.0000	68.0000	68.0000
40000.	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	79.0000	79.0000	79.0000
45000.	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000

TABLE 4.2.5-1 DIVE RECOVERY ANGLE AS FUNCTION OF ALTITUDE AND MACH NUMBER FOR AIRCRAFT C

ALTITUDE \ MACH	MACH										
	0.2	0.4	0.5	0.6	0.8	0.9	1.0	1.1	1.2	1.4	1.6
0.	59.25	237.01	370.32	533.26	948.02	1199.84	1481.29	1792.36	2133.06	2903.33	3792.10
5000.	49.30	197.21	308.14	443.73	788.85	998.39	1232.58	1491.42	1774.91	2415.85	3155.39
10000.	40.76	163.02	254.72	366.80	652.09	825.30	1018.88	1232.85	1467.19	1997.01	2608.34
15000.	33.45	133.81	209.08	301.08	535.25	677.42	836.33	1011.96	1204.31	1639.20	2141.00
20000.	27.25	109.00	170.32	245.25	436.01	551.82	681.26	824.33	981.02	1335.27	1744.03
25000.	22.02	88.06	137.60	198.15	352.26	445.83	550.40	665.99	792.58	1078.79	1409.03
30000.	17.63	70.52	110.18	158.66	282.06	356.99	440.73	533.28	634.64	863.82	1128.26
35000.	13.98	55.93	87.38	125.83	223.70	283.13	349.54	422.94	503.34	685.09	894.82
40000.	11.01	44.02	68.79	99.06	176.10	222.88	275.16	332.94	396.22	539.31	704.40
45000.	8.66	34.65	54.15	77.97	138.61	175.43	216.58	262.06	311.88	424.50	554.45
50000.	6.82	27.28	42.63	61.39	109.13	138.12	170.52	206.33	245.55	334.22	436.53
55000.	5.37	21.48	33.56	48.33	85.92	108.74	134.25	162.44	193.32	263.13	343.68
60000.	4.23	16.92	26.43	38.06	67.66	85.63	105.72	127.92	152.24	207.21	270.64

TABLE 4.2.6-1 DYNAMIC PRESSURE AS FUNCTION OF ALTITUDE AND MACH NUMBER

Tail Deflection = -25.0 Degrees																			
α	-12	-8	-4	0	4	8	12	16	20	24	28	32	36	40	44	48	52	56	60
0.20	-1.00	-0.74	-0.48	-0.22	0.04	0.32	0.56	0.78	0.98	1.13	1.24	1.32	1.35	1.32	1.25	1.16	1.10	1.04	0.97
0.40	-1.01	-0.74	-0.48	-0.21	0.05	0.33	0.57	0.78	0.98	1.12	1.22	1.30	1.33	1.32	1.25	1.16	1.10	1.04	0.97
0.60	-1.03	-0.74	-0.47	-0.20	0.07	0.34	0.59	0.79	0.98	1.10	1.19	1.25	1.29	1.31	1.24	1.17	1.10	1.04	0.97
0.80	-1.10	-0.78	-0.47	-0.19	0.11	0.41	0.62	0.82	1.00	1.12	1.20	1.26	1.30	1.30	1.30	1.30	1.30	1.30	1.30
0.90	-1.13	-0.81	-0.48	-0.18	0.14	0.45	0.64	0.84	1.01	1.13	1.20	1.27	1.30	1.30	1.30	1.30	1.30	1.30	1.30
1.00	-1.25	-0.89	-0.52	-0.18	0.20	0.49	0.70	0.90	1.10	1.24	1.32	1.38	1.39	1.39	1.39	1.39	1.39	1.39	1.39
1.20	-1.20	-0.80	-0.46	-0.16	0.19	0.48	0.75	0.97	1.22	1.22	1.22	1.22	1.22	1.22	1.22	1.22	1.22	1.22	1.22
1.40	-1.10	-0.69	-0.39	-0.13	0.15	0.43	0.69	0.91	1.14	1.14	1.14	1.14	1.14	1.14	1.14	1.14	1.14	1.14	1.14
1.40	-0.90	-0.61	-0.33	-0.10	0.14	0.36	0.59	0.81	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01
1.60	-0.80	-0.55	-0.30	-0.09	0.12	0.33	0.52	0.72	0.89	0.89	0.89	0.89	0.89	0.89	0.89	0.89	0.89	0.89	0.89

Tail Deflection = -15.0 Degrees																			
0.20	-0.93	-0.66	-0.39	-0.12	0.13	0.42	0.65	0.86	1.07	1.24	1.34	1.41	1.42	1.38	1.30	1.20	1.12	1.04	0.97
0.40	-0.97	-0.69	-0.41	-0.13	0.14	0.42	0.66	0.87	1.07	1.23	1.32	1.38	1.40	1.38	1.30	1.20	1.12	1.04	0.97
0.60	-1.00	-0.75	-0.45	-0.15	0.15	0.43	0.68	0.88	1.08	1.28	1.34	1.34	1.38	1.38	1.38	1.38	1.38	1.38	1.38
0.80	-1.10	-0.78	-0.46	-0.15	0.16	0.46	0.70	0.90	1.11	1.21	1.28	1.34	1.38	1.38	1.38	1.38	1.38	1.38	1.38
0.90	-1.14	-0.81	-0.47	-0.14	0.18	0.51	0.73	0.93	1.15	1.23	1.29	1.35	1.38	1.38	1.38	1.38	1.38	1.38	1.38
1.00	-1.20	-0.87	-0.50	-0.15	0.22	0.55	0.79	1.00	1.19	1.32	1.40	1.45	1.46	1.46	1.46	1.46	1.46	1.46	1.46
1.10	-1.20	-0.89	-0.47	-0.14	0.19	0.51	0.81	1.05	1.30	1.30	1.30	1.30	1.30	1.30	1.30	1.30	1.30	1.30	1.30
1.20	-1.10	-0.75	-0.40	-0.12	0.17	0.47	0.75	1.00	1.25	1.25	1.25	1.25	1.25	1.25	1.25	1.25	1.25	1.25	1.25
1.40	-0.90	-0.62	-0.33	-0.09	0.17	0.42	0.65	0.88	1.09	1.09	1.09	1.09	1.09	1.09	1.09	1.09	1.09	1.09	1.09
1.60	-0.80	-0.53	-0.29	-0.07	0.16	0.37	0.56	0.77	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94

Tail Deflection = -5.0 Degrees																			
0.20	-0.79	-0.53	-0.27	-0.01	0.25	0.49	0.75	0.98	1.18	1.33	1.44	1.50	1.50	1.45	1.35	1.25	1.15	1.05	0.96
0.40	-0.84	-0.57	-0.30	-0.03	0.24	0.50	0.76	0.99	1.18	1.32	1.40	1.46	1.47	1.45	1.35	1.25	1.15	1.05	0.96
0.60	-0.89	-0.61	-0.33	-0.05	0.23	0.52	0.78	1.00	1.19	1.30	1.36	1.41	1.43	1.45	1.36	1.26	1.15	1.05	0.95
0.80	-0.98	-0.65	-0.36	-0.05	0.25	0.56	0.80	1.02	1.20	1.32	1.38	1.43	1.45	1.45	1.45	1.45	1.45	1.45	1.45
0.90	-1.07	-0.73	-0.38	-0.05	0.27	0.61	0.82	1.03	1.20	1.33	1.40	1.44	1.46	1.45	1.45	1.45	1.45	1.45	1.45
1.00	-1.10	-0.80	-0.43	-0.07	0.29	0.63	0.88	1.10	1.27	1.42	1.51	1.53	1.53	1.53	1.53	1.53	1.53	1.53	1.53
1.10	-1.00	-0.73	-0.39	-0.07	0.28	0.59	0.89	1.16	1.37	1.37	1.37	1.37	1.37	1.37	1.37	1.37	1.37	1.37	1.37
1.20	-0.90	-0.66	-0.34	-0.06	0.25	0.54	0.82	1.07	1.33	1.33	1.33	1.33	1.33	1.33	1.33	1.33	1.33	1.33	1.33
1.40	-0.80	-0.56	-0.28	-0.03	0.22	0.47	0.70	0.93	1.13	1.13	1.13	1.13	1.13	1.13	1.13	1.13	1.13	1.13	1.13
1.60	-0.70	-0.49	-0.24	-0.02	0.20	0.41	0.61	0.81	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99

TABLE 4.3.1-1 COEFFICIENT OF LIFT AS FUNCTION OF MACH NUMBER AND ANGLE OF ATTACK FOR AIRCRAFT A

Tail Deflection = 5.0 Degrees																			
α	-12	-8	-4	0	4	8	12	16	20	24	28	32	36	40	44	48	52	56	60
0.20	-0.75	-0.48	-0.21	0.06	0.33	0.58	0.84	1.07	1.29	1.44	1.54	1.59	1.58	1.53	1.42	1.29	1.17	1.05	0.93
0.40	-0.78	-0.50	-0.22	0.06	0.33	0.60	0.86	1.10	1.30	1.43	1.52	1.57	1.57	1.53	1.43	1.29	1.17	1.05	0.93
0.60	-0.82	-0.52	-0.24	0.05	0.34	0.62	0.89	1.13	1.31	1.41	1.49	1.54	1.55	1.52	1.44	1.30	1.17	1.06	0.94
0.80	-0.87	-0.56	-0.26	0.05	0.36	0.67	0.91	1.13	1.30	1.42	1.50	1.54	1.54	1.52	1.52	1.52	1.52	1.52	1.52
0.90	-0.92	-0.60	-0.28	0.05	0.37	0.71	0.92	1.13	1.30	1.42	1.50	1.54	1.54	1.52	1.52	1.52	1.52	1.52	1.52
1.00	-1.00	-0.69	-0.35	0.03	0.39	0.74	1.00	1.18	1.40	1.52	1.62	1.63	1.63	1.63	1.63	1.63	1.63	1.63	1.63
1.10	-0.90	-0.63	-0.31	0.02	0.36	0.67	0.96	1.21	1.47	1.47	1.47	1.47	1.47	1.47	1.47	1.47	1.47	1.47	1.47
1.20	-0.80	-0.57	-0.26	0.02	0.32	0.61	0.89	1.15	1.41	1.41	1.41	1.41	1.41	1.41	1.41	1.41	1.41	1.41	1.41
1.40	-0.70	-0.49	-0.22	0.02	0.27	0.52	0.76	0.98	1.17	1.17	1.17	1.17	1.17	1.17	1.17	1.17	1.17	1.17	1.17
1.60	-0.60	-0.43	-0.19	0.02	0.24	0.45	0.65	0.85	1.02	1.02	1.02	1.02	1.02	1.02	1.02	1.02	1.02	1.02	1.02

Tail Deflection = 15.0 Degrees																			
0.20	-0.62	-0.36	-0.10	0.16	0.42	0.66	0.92	1.17	1.40	1.55	1.65	1.68	1.66	1.60	1.50	1.36	1.22	1.08	0.93
0.40	-0.66	-0.36	-0.10	0.16	0.44	0.69	0.96	1.18	1.40	1.53	1.61	1.64	1.64	1.60	1.50	1.36	1.22	1.08	0.93
0.60	-0.70	-0.37	-0.10	0.17	0.46	0.73	1.00	1.20	1.39	1.51	1.57	1.60	1.61	1.60	1.50	1.36	1.21	1.07	0.93
0.80	-0.77	-0.44	-0.15	0.14	0.45	0.76	0.99	1.20	1.38	1.53	1.60	1.63	1.63	1.60	1.60	1.60	1.60	1.60	1.60
0.90	-0.85	-0.52	-0.19	0.12	0.45	0.78	0.98	1.20	1.38	1.55	1.62	1.65	1.65	1.60	1.60	1.60	1.60	1.60	1.60
1.00	-0.90	-0.58	-0.35	0.05	0.48	0.79	0.99	1.22	1.43	1.50	1.60	1.61	1.61	1.61	1.61	1.61	1.61	1.61	1.61
1.10	-0.80	-0.70	-0.37	0.05	0.43	0.72	1.02	1.28	1.51	1.51	1.51	1.51	1.51	1.51	1.51	1.51	1.51	1.51	1.51
1.20	-0.70	-0.57	-0.23	0.07	0.37	0.66	0.95	1.21	1.47	1.47	1.47	1.47	1.47	1.47	1.47	1.47	1.47	1.47	1.47
1.40	-0.60	-0.45	-0.16	0.08	0.32	0.57	0.81	1.03	1.23	1.23	1.23	1.23	1.23	1.23	1.23	1.23	1.23	1.23	1.23
1.60	-0.50	-0.38	-0.14	0.08	0.28	0.50	0.69	0.89	1.07	1.07	1.07	1.07	1.07	1.07	1.07	1.07	1.07	1.07	1.07

TABLE 4.3.1-1 COEFFICIENT OF LIFT AS FUNCTION OF MACH NUMBER AND ANGLE OF ATTACK
FOR AIRCRAFT A (CONCLUDED)

MACH	0.20	0.40	0.60	0.80	0.90	1.00	1.10	1.20	1.40	1.60
CLmax	1.68	1.64	1.61	1.63	1.65	1.61	1.51	1.47	1.23	1.07

TABLE 4.3.1-2 CLmax AS A FUNCTION OF MACH (FORTRAN NAME: CLMAXB(10))
(AIRCRAFT A)

MACH \ ALT	0.0	0.2	0.4	0.6	0.8	0.9	1.0	1.2	1.4	1.6
0.	1022.0	609.0	95.0	-781.0	-1937.0	-2847.0	-945.0	193.0	11556.0	11556.0
10000.	705.0	393.0	50.0	-541.0	-1384.0	-2097.0	219.0	1867.0	10103.0	10413.0
20000.	952.0	660.0	317.0	-210.0	-938.0	-1399.0	621.0	2691.0	8410.0	8744.0
30000.	1007.0	1007.0	743.0	412.0	-31.0	-421.0	441.0	2879.0	6659.0	7044.0
35000.	1185.0	1185.0	956.0	685.0	334.0	76.0	357.0	2855.0	5934.0	6262.0
40000.	1361.0	1361.0	1155.0	937.0	666.0	475.0	268.0	2295.0	4701.0	4991.0
45000.	1359.0	1359.0	1359.0	1149.0	934.0	800.0	628.0	1721.0	3545.0	3821.0
50000.	1311.0	1311.0	1311.0	1333.0	1152.0	1040.0	919.0	1280.0	2636.0	2854.0
55000.	1102.0	1102.0	1102.0	1102.0	1301.0	1243.0	1135.0	923.0	1962.0	2120.0
60000.	817.0	817.0	817.0	817.0	985.0	1102.0	1194.0	1130.0	1469.0	1587.0

TABLE 4.3.2-1 IDLE THRUST AS A FUNCTION OF MACH AND ALTITUDE
 FORTRAN NAME (THRIDB (10, 10)
 (AIRCRAFT A)

MACH \ ALT	0.0	0.2	0.4	0.6	0.8	0.9	1.0	1.2	1.4	1.6
0.	11547.0	11720.0	11655.0	11517.0	11165.0	10893.0	10741.0	11437.0	11556.0	11556.0
10000.	8068.0	8307.0	8634.0	9276.0	9446.0	9403.0	9335.0	9862.0	10103.0	10413.0
20000.	5479.0	5807.0	6066.0	6505.0	7449.0	7485.0	7647.0	8110.0	8410.0	8744.0
30000.	3621.0	3621.0	3802.0	4215.0	4907.0	5376.0	5773.0	6567.0	6659.0	7044.0
35000.	2836.0	2836.0	2977.0	3297.0	3830.0	4211.0	4623.0	5789.0	5881.0	6262.0
40000.	2178.0	2178.0	2282.0	2535.0	2945.0	3242.0	3591.0	4549.0	4701.0	4991.0
45000.	1733.0	1733.0	1773.0	1930.0	2253.0	2484.0	2769.0	3427.0	3545.0	3821.0
50000.	1311.0	1311.0	1311.0	1461.0	1714.0	1894.0	2154.0	2564.0	2636.0	2854.0
55000.	1102.0	1102.0	1102.0	1102.0	1301.0	1449.0	1639.0	1863.0	1962.0	2120.0
60000.	817.0	817.0	817.0	817.0	985.0	1102.0	1194.0	1324.0	1469.0	1587.0

TABLE 4.3.2-2 MILITARY THRUST AS A FUNCTION OF MACH AND ALTITUDE
 (FORTRAN NAME: (THRMLB (10, 10)
 (AIRCRAFT A)

MACH \ ALT	0.0	0.2	0.4	0.5	0.8	0.9	1.0	1.2	1.4	1.6
	0.	18587.0	20022.0	21288.0	22814.0	24270.0	25355.0	27001.0	30219.0	35414.0
10000.	13008.0	14190.0	15865.0	18258.0	20207.0	21283.0	22488.0	25103.0	29156.0	33724.0
20000.	9033.0	10145.0	11222.0	12804.0	15817.0	16531.0	17945.0	20083.0	23212.0	26734.0
30000.	6452.0	6452.0	7218.0	8532.0	10464.0	11863.0	13283.0	15657.0	17898.0	20569.0
35000.	4802.0	4802.0	5385.0	6401.0	7867.0	8921.0	10103.0	13055.0	15029.0	17277.0
40000.	3829.0	3829.0	4313.0	5144.0	6359.0	7211.0	8226.0	10606.0	12260.0	14179.0
45000.	3156.0	3156.0	3156.0	3835.0	4784.0	5456.0	6342.0	8045.0	9306.0	10944.0
50000.	2226.0	2226.0	2226.0	2770.0	3526.0	4062.0	4889.0	6039.0	6984.0	8319.0
55000.	1889.0	1889.0	1889.0	1889.0	2513.0	3032.0	3670.0	4386.0	5211.0	6261.0
60000.	1379.0	1379.0	1379.0	1379.0	1767.0	2167.0	2568.0	3068.0	3865.0	4672.0

TABLE 4.3.2-3 AFTERBURNER THRUST AS A FUNCTION OF MACH AND ALTITUDE (THRABB (10, 10))

MACH \ CL	0.2	0.4	0.5	0.8	0.9	1.0	1.1	1.2	1.4	1.6
	-1.20	-12.00	-12.00	-12.00	-12.00	-12.00	-12.00	-12.00	-12.00	-12.00
-1.00	-12.00	-12.00	-12.00	-12.00	-11.18	-10.67	-12.00	-12.00	-12.00	-12.00
-0.80	-12.00	-11.41	-10.71	-9.82	-8.82	-8.00	-9.04	-10.33	-12.00	-12.00
-0.60	-9.08	-8.44	-7.86	-7.31	-6.51	-5.84	-6.47	-7.25	-8.67	-10.10
-0.40	-6.00	-5.48	-5.00	-4.55	-4.23	-3.67	-4.12	-4.75	-5.71	-6.56
-0.20	-2.92	-2.52	-2.14	-1.94	-1.82	-1.44	-1.63	-2.00	-2.72	-3.27
0.00	0.15	0.44	0.71	0.67	0.63	0.78	0.80	0.77	0.48	0.36
0.20	3.23	3.41	3.57	3.33	3.13	3.00	3.09	3.35	3.68	4.00
0.40	6.50	6.46	6.34	5.94	5.53	5.29	5.55	6.07	6.88	7.81
0.60	9.69	9.54	9.23	8.67	7.88	7.65	8.13	8.86	10.26	11.80
0.80	12.87	12.70	12.36	12.00	11.62	10.72	10.80	11.71	13.74	15.80
1.00	16.40	16.21	16.00	15.64	15.43	14.18	13.63	14.88	17.40	40.00
1.20	20.53	20.57	20.36	20.00	20.00	18.35	16.76	18.00	40.00	40.00
1.40	26.55	28.00	31.20	29.60	28.00	23.47	40.00	40.00	40.00	40.00
1.60	40.00	40.00	40.00	40.00	40.00	40.00	40.00	40.00	40.00	40.00

TABLE 4.3.3-1 ANGLE OF ATTACK (in degrees) AS A FUNCTION OF Ca₀BL AND MACH (ALFCLB (15, 10))

ALPHA \ MACH	-12.	-8.	-4.	0.	4.	8.	12.	16.	20.	24.	28.	32.	36.	40.
0.20	-0.79	-0.53	-0.27	-0.01	0.25	0.49	0.75	0.98	1.18	1.33	1.44	1.50	1.50	1.45
0.40	-0.84	-0.57	-0.30	-0.03	0.24	0.50	0.76	0.99	1.18	1.32	1.40	1.46	1.47	1.45
0.60	-0.89	-0.61	-0.33	-0.05	0.23	0.52	0.78	1.00	1.19	1.30	1.36	1.41	1.43	1.45
0.80	-0.98	-0.65	-0.36	-0.05	0.25	0.56	0.80	1.02	1.20	1.32	1.38	1.43	1.45	1.45
0.90	-1.07	-0.73	-0.38	-0.05	0.27	0.61	0.82	1.03	1.20	1.33	1.40	1.44	1.46	1.45
1.00	-1.10	-0.80	-0.43	-0.07	0.29	0.63	0.88	1.10	1.27	1.42	1.51	1.53	1.53	1.53
1.10	-1.00	-0.73	-0.39	-0.07	0.28	0.59	0.89	1.16	1.37	1.37	1.37	1.37	1.37	1.37
1.20	-0.90	-0.66	-0.34	-0.06	0.25	0.54	0.82	1.07	1.33	1.33	1.33	1.33	1.33	1.33
1.40	-0.80	-0.56	-0.28	-0.03	0.22	0.47	0.70	0.93	1.13	1.13	1.13	1.13	1.13	1.13
1.60	-0.70	-0.49	-0.24	-0.02	0.20	0.41	0.61	0.81	0.99	0.99	0.99	0.99	0.99	0.99

TABLE 4.3.4-1 C_{eubL} AS A FUNCTION OF ANGLE OF ATTACK (in degrees) AND MACH
 FORTRAN NAME: CLFALB (10, 14)
 (AIRCRAFT A)

MACH \ CL	0.2	0.4	0.6	0.8	0.9	1.0	1.1	1.2	1.4	1.6
0.0	0.01960	0.01960	0.01960	0.02010	0.02040	0.02610	0.03960	0.04660	0.04490	0.04300
0.2	0.02180	0.02180	0.02180	0.02180	0.02270	0.03210	0.04620	0.05230	0.05250	0.05250
0.4	0.03760	0.03760	0.03760	0.03700	0.03800	0.05540	0.07720	0.08600	0.09120	0.09670
0.6	0.07780	0.07780	0.07680	0.07450	0.07530	0.10370	0.13400	0.14960	0.16980	0.19060
0.8	0.15170	0.15170	0.15450	0.15830	0.15820	0.18090	0.21360	0.24200	0.30000	0.36000
1.0	0.25820	0.25820	0.26910	0.28620	0.29930	0.31660	0.33790	0.37110	0.40000	0.50000
1.2	0.44380	0.44380	0.46630	0.50920	0.52750	0.53700	0.54000	0.55000	0.56000	0.58000
1.4	0.85590	0.85590	0.87740	0.90000	0.90500	0.87870	1.00000	1.10000	1.20000	1.40000
1.6	1.28730	1.28730	1.31180	1.32220	1.32300	1.50000	1.60000	1.70000	1.80000	2.00000

TABLE 4.3.5-1 COEFFICIENT OF DRAG AS A FUNCTION OF MACH AND COEFFICIENT OF LIFT
 FORTRAN NAME: CDFCLB (9, 10)
 (AIRCRAFT A)

MACH \ CD	0.2	0.4	0.6	0.8	0.9	1.0	1.1	1.2	1.4	1.6
0.02	0.0364	0.0364	0.0364	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.04	0.4119	0.4119	0.4122	0.4160	0.4107	0.2678	0.0121	0.0000	0.0000	0.0000
0.06	0.5114	0.5114	0.5143	0.5227	0.5180	0.4190	0.2890	0.2457	0.2388	0.2339
0.08	0.6059	0.6059	0.6082	0.6131	0.6113	0.5019	0.4099	0.3644	0.3421	0.3244
0.10	0.6601	0.6601	0.6597	0.6609	0.6596	0.5847	0.4803	0.4440	0.4224	0.4070
0.20	0.8907	0.8907	0.8794	0.8652	0.8592	0.8282	0.7658	0.7091	0.6464	0.6111
0.30	1.0450	1.0450	1.0313	1.0124	1.0006	0.9755	0.9390	0.8899	0.8000	0.7292
0.40	1.1528	1.1528	1.1328	1.1021	1.0883	1.0757	1.0614	1.0323	1.0000	0.8571
0.50	1.2273	1.2273	1.2164	1.1918	1.1759	1.1664	1.1604	1.1441	1.1250	1.0000
0.60	1.2758	1.2758	1.2650	1.2465	1.2384	1.2369	1.2261	1.2182	1.2125	1.2049
0.70	1.3243	1.3243	1.3137	1.2976	1.2914	1.2954	1.2696	1.2546	1.2437	1.2293
0.80	1.3729	1.3729	1.3623	1.3488	1.3444	1.3539	1.3130	1.2909	1.2750	1.2537
0.90	1.4204	1.4204	1.4104	1.4000	1.3974	1.4069	1.3565	1.3273	1.3062	1.2780
1.00	1.4668	1.4668	1.4565	1.4474	1.4455	1.4390	1.4000	1.3636	1.3375	1.3024
1.20	1.5595	1.5595	1.5485	1.5422	1.5411	1.5034	1.4667	1.4333	1.4000	1.3512
1.40	1.6000	1.6000	1.6000	1.6000	1.6000	1.5678	1.5333	1.5000	1.4667	1.4000
1.60	1.6000	1.6000	1.6000	1.6000	1.6000	1.6000	1.6000	1.5667	1.5333	1.4667
1.80	1.6000	1.6000	1.6000	1.6000	1.6000	1.6000	1.6000	1.6000	1.6000	1.5333
2.00	1.6000	1.6000	1.6000	1.6000	1.6000	1.6000	1.6000	1.6000	1.6000	1.6000

TABLE 4.3.6-1 COEFFICIENT OF LIFT AS A FUNCTION OF COEFFICIENT OF DRAG AND MACH

FORTRAN NAME: CLFCDB (19, 10)
(AIRCRAFT A)

MACH	0.4	0.5	0.6	0.7	0.8	0.9	1.0	1.2	1.5
ALT									
200.	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2500.	63.0	50.0	45.0	40.0	35.0	33.0	32.0	18.0	16.0
5000.	90.0	90.0	82.0	70.0	60.0	57.0	55.0	30.0	28.0
7500.	90.0	90.0	90.0	90.0	88.0	80.0	75.0	38.0	37.0
10000.	90.0	90.0	90.0	90.0	90.0	90.0	90.0	48.0	45.0
12500.	90.0	90.0	90.0	90.0	90.0	90.0	90.0	60.0	55.0
15000.	90.0	90.0	90.0	90.0	90.0	90.0	90.0	72.0	65.0
20000.	90.0	90.0	90.0	90.0	90.0	90.0	90.0	90.0	88.0

TABLE 4.3.6-2 DIVE RECOVERY ANGLE AS A FUNCTION OF MACH AND ALTITUDE

FORTRAN NAME: RECAGB (8, 9)

(AIRCRAFT A)

APPENDIX A

LISTING OF THE 1983 BASELINE
ADAPTIVE MANEUVERING LOGIC PROGRAM

Table of Contents for Appendix A

PROGRAM AML83V1	A1
SUBROUTINE AERF4	A10
SUBROUTINE CLOSS	A13
SUBROUTINE CMTRX	A13
SUBROUTINE CSRHO	A14
SUBROUTINE ERMSG	A14
SUBROUTINE FIRCON	A14
SUBROUTINE GETRXN	A16
SUBROUTINE HUBLO	A16
SUBROUTINE INRD	A17
SUBROUTINE MEINTL	A21
SUBROUTINE NORPLN	A23
SUBROUTINE OILER	A23
SUBROUTINE PAIRCR	A23
SUBROUTINE PCELL	A25
SUBROUTINE PCSRO	A27
SUBROUTINE PRESR	A28
SUBROUTINE PRTF4	A29
SUBROUTINE PRETNW	A32
SUBROUTINE QUATEX	A33
SUBROUTINE RELGN	A34
SUBROUTINE TRAFER	A35
SUBROUTINE TRAPL	A35
SUBROUTINE EQMOTT	A36
SUBROUTINE EXTRT	A43
SUBROUTINE FILTRT	A44
SUBROUTINE THRTL	A45
SUBROUTINE TRYNXT	A46
SUBROUTINE QUATT	A51
SUBROUTINE REACTT	A52
SUBROUTINE STATET	A57
SUBROUTINE EQMOTA	A62
SUBROUTINE EXTRA	A69
SUBROUTINE FILTRA	A70
SUBROUTINE QUATA	A71
SUBROUTINE REACTA	A72
SUBROUTINE STATEA	A77
SUBROUTINE THRTLA	A81
SUBROUTINE TRYNXA	A82
FUNCTION CLADES	A88
FUNCTION GG	A88
FUNCTION GGG	A88
FUNCTION GGGG	A88
FUNCTION RADIFY	A88
FUNCTION SBDEFA	A89
FUNCTION SLAPDF	A89

PROGRAM AML83V1

C **** 1983 BASELINE VERSION OF AML
C -----

C *** DEVELOPED FOR:
C
C NASA DRYDEN FLIGHT RESEARCH FACILITY

C *** DEVELOPED BY:
C
C GEORGE H. BURGIN
C TITAN SYSTEMS INC.
C 9191 TOWNE CENTRE DRIVE
C SAN DIEGO CA 92122
C
C 9619) 453-9500

C *** MAIN PROGRAM FOR THE AML 83 BASELINE VERSION
C (CORRECTED QUATERNION INTEGRATION)
C

COMMON/K411 / KTAPE, KPR, KAPR
COMMON/ TER / NPNT, NPNT, NLAB, NKTIC, KPRT, DHOR, DVER
COMMON/ K4/DT, TBEGN, TIME, PI, PIDV2, PIDV4, TWOPI, DEGRD, RADDG, G
1 ,VAR(20), IVAR(20), TEND

C *** IVAR(1) USE FOR SUPPRESSING TACTICS PRINTOUT
C *** IVAR(2) USED TO FLAG END OF JOB IN INRD
C

COMMON/AEROUT/TIDLEX ,TMILX ,TABX
1 ,FLODMX ,FLODSX
2 ,FMMINX ,FMMAXX ,RECANX
3 ,ALPHAX
4 ,CDX ,SBCDX
COMMON/ATTTAR/ANGETA, ANGETT, ANGXIA, ANGXIT, CRGR(3,3), DLOSSA,
1 DLOSST, DRGR(3,3), FAZMSA, FAZMST, FELVSA, FELVST, FLOSSA,
2 FLOSST, PSUBSA, PSUBST, RANGE, RRATE, XAINT, XTINA, YAINT,
3 YTINA, ZAINT, ZTINA
COMMON/CELSTA/AN1A, AN2A, AN3A, DTPRA, GLEVLA, ICMNWA, ICOMDA, ICRECA,
1 ISTRGA, JVALUA, LCELLA(40), LVALA(40), LWEGHA(40),
2 NCELLA, NCLSTA, NOWPRA, NTILTA, NTRYA, ROTSA, ROTA, TIMEPA
COMMON/CELSTT/AN1T, AN2T, AN3T, DTPRT, GLEVL, ICMNWT, ICOMDT, IRECT,
1 ISTRGT, JVALUT, LCELLT(40), LVALT(40), LWEGHT(40),
2 NCELLT, NCLSTT, NOWPRT, NTILT, NTRYT, ROTST, ROTT, TIMEPT
COMMON/EXTPOA/TIMEXA, DELTA, XZEROT, YZEROT, ZZEROT, XMIN1T, YMIN1T,
2 ZMIN1T, XMIN2T, YMIN2T, ZMIN2T, XEXPT, YEXPT, ZEXPT,
2 XDEXPT, YDEXPT, ZDEXPT, VELEXT, DRCEXT(3,3)
COMMON/EXTPOT/TIMEXT, DELTT, XZEROA, YZEROA, ZZEROA, XMIN1A, YMIN1A,
1 ZMIN1A, XMIN2A, YMIN2A, ZMIN2A, XEXPA, YEXPA, ZEXPA,
2 XDEXPA, YDEXPA, ZDEXPA, VELEXA, DRCEXA(3,3)

COMMON/HASSLE/IABLEA , IABLET

COMMON/PREDIC/CNEW(3,3), VXNEW, VYNEW, VZNEW, XXNEW, YYNEW, ZZNEW
COMMON/SCALES/KALEA (15), KALEB (15), KALEC (25), KALED (25)
1 , VAKALA (7), VAKALB (5), VAKALC (14), VAKALD (12)
2 , DELALT , DELACH , DELCEL
COMMON/TABLES/CS0 (121) , RHO0 (121)
1 , TIDLE (7, 14, 2) , TMIL (7, 14, 2) , TAB (7, 14, 2)
2 , SURF (2) , F4VG (5, 12, 2) , F4SG (5, 12, 2)
3 , F4MMI (13, 2) , F4MMA (13, 2) , RECANG (10, 12, 2)
4 , F4CLA (10, 2) , F4ALP (10, 21, 2) , CLMAX (14, 2)
5 , F5CLA (10, 2) , F5ALP (10, 21, 2)
6 , F4CDR (10, 21, 2) , SBCDR (14, 2)
7 , F5CDR (10, 21, 2)
COMMON/TRIALA/AN1TRA (10), AN2TRA (10), AN3TRA (10), CPRA (3, 3, 10),
1 DLOSRA (10), DRAGTA (10), DRGPRA (10), DXEPRA (10),
2 DYEPR (10), DZEPRA (10), FLDTRA (10), FLOSRA (10),
3 ICNPR (10), ICTRYA (10), ISTRPA (20, 10), ISTRYA (10),
4 IVALTA, IVPRA (10), KNEWA, KRO TSA, PSTRA (10), PXEPRA (10),
5 PYEPRA (10), PZEPRA (10), RNGPRA (10), ROTNCA, ROTN2A,
6 ROTRYA (10), THTRA (10), TPOTRA (10), XEXT, YEXT, ZEXT
COMMON/TRIALT/AN1TRT (10), AN2TRT (10), AN3TRT (10), CPRT (3, 3, 10),
1 DLOSRT (10), DRATT (10), DRGPRT (10), DXEPRT (10),
2 DYEPR (10), DZEPRT (10), FLDTRT (10), FLOSRT (10),
3 ICNPR (10), ICTRYT (10), ISTRPT (20, 10), ISTRYT (10),
4 IVALTT, IVPRT (10), KNEWT, KROTST, PSTRT (10), PXEPRT (10),
5 PYEPRT (10), PZEPRT (10), RNGPRT (10), ROTNCT, ROTN2T,
6 ROTRYT (10), THTRT (10), TPOTRT (10), XEXA, YEXA, ZEXA
COMMON/VARBLA/ACCXEA, ACCYEA, ACCZEA, ALFAA, CBARA (3, 3), CBA (3, 3), CDA,
1 CLALFA, CLA, COPHIA, COPSIA, COTHTA , CSA, C (3, 3), DRAGA,
2 FFLOSA, FLIFTA, FLODMA, FMACHA, FMMAXA, FMMINA, INIZA,
3 PHIBRA, PHIA, PMAXA, PPDTA, PA, PSIBRA, PSIA, QMAXA, QPDTA,
4 QA, RECANA, RHOA, RMAXA, RPDTA, RA, SIPHIA, SIPSIA, SITHTA ,
5 SPECEA, SA, TABA, THETAA, THETBA, TIDLEA, TMILA, TPOSA,
6 TRSTA, UA, VELA, VHORA, VA, WEITA, WA, XEDOTA, XEA, YEDOTA,
7 YEA, ZEDOTA, ZEA, FLODSA
COMMON/VARBLT/ACCXET, ACCYET, ACCZET, ALFAT, CBART (3, 3), CBT (3, 3), CDT,
1 CLALFT, CLT, COPHIT, COPSIT, COTHTT , CST, D (3, 3), DRAGT,
2 FFLOST, FLIFTT, FLODMT, FMACHT, FMMAXT, FMINT, INIZT,
3 PHIBRT, PHIT, PMAXT, PPDTT, PT, PSIBRT, PSIT, QMAXT, QPDTT,
4 QT, RECANT, RHOT, RMAXT, RPDTT, RT, SIPHIT, SIPSIT, SITHTT ,
5 SPECET, ST, TABT, THETAT, THETBT, TIDLET, TMILT, TPOST,
6 TRSTT, UT, VELT, VHORT, VT, WEITT, WT, XEDOTT, XET, YEDOTT,
7 YET, ZEDOTT, ZET, FLODST
COMMON/TALLY/ OPTIMA , OPTIMT , GUNTMA , GUNTMT , ANGOFA, ANGOFT,
1 DEVAA, DEVAT

C

COMMON/IDS/IDTAG (2)
COMMON/UNITS/LP, CR, TAPE1, TAPE2, TAPE3, TAPE4
INTEGER CR, TAPE1, TAPE2, TAPE3, TAPE4
DIMENSION IDENT (2)
DATA IDENT (1)/4HATKR/, IDENT (2)/4HTRGT/

```

C
PRINT *, ' PROGRAM STARTED EXECUTION '
LP=51
DO 4524 J=1,20
IVAR(J)=0
VAR(J)=0.
4524 CONTINUE

```

```

C
PI = 4.*(ATAN(1.))
PIDV4 = PI / 4.
PIDV2 = PIDV4 + PIDV4
TWOPI = PI + PI
DEGRD = 180. / PI
RADDG = 1. / DEGRD
G = 32.17
IFIRS = 1
THEAD=1.
ITAPP = 0
DHOR = 7.
DVER = 8.5
KPRT=0
NLAB=16

```

```

C
C
C
READ CYCLE

```

```

90 FORMAT(2A4)
91 FORMAT(15F5.2)
92 FORMAT(10F8.7)
93 FORMAT(7F10.0)

```

```

READ(35, 90)KODWOR
IF(KODWOR.NE.4HATMO) CALL EXIT
SOUND SPEED (ALT)

```

```

C
C
READ(35, 92)(CS0(J),J=1,121)
AIR DENSITY (ALT)
READ(35, 92)(RHO0(J),J=1,121)
KEE = 0
LRD = 2
LRDV = 1
READ(35, 90)LABEL, IDT
IF(LABEL.EQ.IDENT(1))KEE= 1
IF(LABEL.EQ.IDENT(2))KEE= 2
IF(KEE.NE.0)GO TO 270
LRD = 1
KEE = 1

```

```

270 CONTINUE
IDTAG(KEE)= IDT
C
REFERENCE AREA
READ(35, 92)SURF(KEE)
READ(35, 90)KODWOR
IF(KODWOR.NE.4HLDET) CALL EXIT
C
MAX LOADFACTOR (ALT,MACH)
DO 271 I=1,5

```

```

      READ(35, 91) (F4VG(I,J,KEE),J=1,12)
271 CONTINUE
C   SUSTAINED LOADFACTOR (ALT,MACH)
      DO 272 I=1,5
      READ(35, 91) (F4SG(I,J,KEE),J=1,12)
272 CONTINUE
C   ENGINE PERFORMANCE INPUT
      READ(35, 90)KODWOR
      IF(KODWOR.NE.4HENGN) CALL EXIT
C   IDLE THRUST (ALT,MACH) FOR 1/2 OF PLANES ENGINES
      DO 278 I=1,7
      READ(35, 93) (TIDLE(I,J,KEE),J=1,14)
278 CONTINUE
C   MILITARY THRUST (ALT,MACH) FOR 1/2 OF PLANES ENGINES
      DO 279 I=1,7
      READ(35, 93) (TMIL(I,J,KEE),J=1,14)
279 CONTINUE
C   AFTERBURN THRUST (ALT,MACH) FOR 1/2 OF PLANES ENGINES
      DO 280 I=1,7
      READ(35, 93) (TAB(I,J,KEE),J=1,14)
280 CONTINUE
      READ(35, 90)KODWOR
      IF(KODWOR.NE.4HARDT) CALL EXIT
C   ANGLE OF ATTACK (MACH,CL) SLATS/FLAPS IN
      DO 274 I=1,21
      READ(35, 92) (F4ALP(J,I,KEE),J=1,10)
274 CONTINUE
C   ANGLE OF ATTACK (MACH,CL) SLAPS OUT
      DO 275 I=1,21
      READ(35, 92) (F5ALP(J,I,KEE),J=1,10)

275 CONTINUE
C   DRAG COEFFICIENT (MACH,CL) SLAPS IN
      DO 276 I=1,21
      READ(35, 92) (F4CDR(J,I,KEE),J=1,10)

276 CONTINUE
C   DRAG COEFFICIENT (MACH,CL) SLAPS OUT
      DO 277 I=1,21
      READ(35, 92) (F5CDR(J,I,KEE),J=1,10)
277 CONTINUE
      READ(35, 90)KODWOR
      IF(KODWOR.NE.4HLMTS) CALL EXIT
C   MAX CL (MACH)
      READ(35, 91) (CLMAX(J,KEE),J=1,14)
C   MACH MIN (ALT)
      READ(35, 91) (F4MMI(J,KEE),J=1,13)
C   MACH MAX (ALT)
      READ(35, 91) (F4MMA(J,KEE),J=1,13)
C   RECOVERY ANGLE (ALT,MACH)
      DO 273 I=1,10
      READ(35, 91) (RECANG(I,J,KEE),J=1,12)
273 CONTINUE
      READ(35, 90)KODWOR

```

```

IF(KODWOR.NE.4HSPBR) CALL EXIT
C SPEEDBRAKE DRAG COEFFICIENT (MACH) FULL DEFLECTION
READ(35, 91)(SBCDR(J,KEE),J=1,14)
IF(LRDV.EQ.LRD)GO TO 300
LRDV = 2
KEE = 3-KEE
READ(35, 90)LABEL, IDT
IF(LABEL.EQ.IDENT(KEE))GO TO 270
CALL EXIT
300 CONTINUE
IF(LRD.EQ.2)GO TO 330
C TRANSFER DATA
CALL TRAFER(F4VG(1,1,1),F4VG(1,1,2),60)
CALL TRAFER(F4SG(1,1,1),F4SG(1,1,2),60)
CALL TRAFER(F4MMI(1,1),F4MMI(1,2),13)
CALL TRAFER(F4MMA(1,1),F4MMA(1,2),13)
CALL TRAFER(CLMAX(1,1),CLMAX(1,2),14)
CALL TRAFER(RECANG(1,1,1),RECANG(1,1,2),120)
CALL TRAFER(F4ALP(1,1,1),F4ALP(1,1,2),210)
CALL TRAFER(F5ALP(1,1,1),F5ALP(1,1,2),210)
CALL TRAFER(F5CDR(1,1,1),F5CDR(1,1,2),210)
CALL TRAFER(F4CDR(1,1,1),F4CDR(1,1,2),210)
CALL TRAFER(F4CLA(1,1),F4CLA(1,2),10)
CALL TRAFER(F5CLA(1,1),F5CLA(1,2),10)
CALL TRAFER(SBCDR(1,1),SBCDR(1,2),14)
CALL TRAFER(TIDLE(1,1,1),TIDLE(1,1,2),98)
CALL TRAFER(TMIL(1,1,1),TMIL(1,1,2),98)
CALL TRAFER(TAB(1,1,1),TAB(1,1,2),98)
IDTAG(2)= IDTAG(1)
SURF(2)= SURF(1)
330 CONTINUE

SA = SURF(1)
ST = SURF(2)
C DATA CONVERSION
DO 355 J=1,240
RECANG(J,1,1)= RECANG(J,1,1)*RADDG

355 CONTINUE
DO 360 J=1,420
F4ALP(J,1,1)= F4ALP(J,1,1)*RADDG
F5ALP(J,1,1)= F5ALP(J,1,1)*RADDG
360 CONTINUE
C DATA PREPARATION
DCL = DELCEL+DELCEL
DO 370 KEE=1,2
DO 365 J=1,10
F4CLA(J,KEE)= DCL/(F4ALP(J,4,KEE)-F4ALP(J,2,KEE))
F5CLA(J,KEE)= DCL/(F5ALP(J,4,KEE)-F5ALP(J,2,KEE))
365 CONTINUE
370 CONTINUE
READ(35, 90)KODWOR
IF(KODWOR.NE.4HPRNT) GO TO 567
C TABLE OUTPUT

```

```

CALL PCSRO
CALL PRTE4 (IDTAG (1) ,LRD)
CALL FIRCON (0 ,DUMMY ,DUMMY ,DUMMY ,DUMMY)
C
C *** REENTRY POINT FOR STARTING ADDITIONAL CASES
C
567 CONTINUE
C
CALL INRD
C
-----
IF (IVAR (2) .EQ. 1) RETURN
C
CALL HUBLO (-1 ,DUMMY ,DUMMY ,DUMMY)
IVAR (1) = 0
IF (KAPR .GE. 1) IVAR (1) = 1
WRITE (46 ,572)
572 FORMAT ( 1H1 )
TENDD = TEND + DT / 2.
ICMNA=0
ICMNAWT=0
KOUN=9999
THEADS = TBEGN + THEAD
ITAPE = 9999
ITAPC = 0
XYMAX = -10.E30
XYMIN = 10.E30
C
C *** INITIALIZE EQUATION OF MOTION AND REACTION ROUTINES
C
INIZA=1
INIZT=1
ICRECA=1
ICRECT=1
CALL EQMOTA
CALL EQMOTT
CALL REACTA (XET ,YET ,ZET ,D)
CALL REACTT (XEA ,YEA ,ZEA ,C)
CALL MEINTL
INIZA=0
INIZT=0
ICRECA=0
ICRECT=0
C
C GET APPROXIMATE LOS ANGLE AT TIME MINUS DT
C
XA = XEA - XEDOTA * DT
YA = YEA - YEDOTA * DT
ZA = ZEA - ZEDOTA * DT
XT = XET - XEDOTT * DT
YT = YET - YEDOTT * DT
ZT = ZET - ZEDOTT * DT
CALL CLOSS (XA ,YA ,ZA ,XT ,YT ,ZT ,XEDOTA ,YEDOTA ,ZEDOTA ,VELA ,FFLOSA)

```

```

      CALL CLOSS (XT, YT, ZT, XA, YA, ZA, XEDOTT, YEDOTT, ZEDOTT, VELT, FFLOST)
C
C *** OPERATE LOOP
C *****
C
500 CONTINUE
C
      CALL EQMOTA
      CALL EQMOTT
      -----
      CALL HUBLO (0, 1, DT, FLODMA*GLEVLA)
      CALL HUBLO (0, 2, DT, FLODMT*GLEVLTA)
C
      VELA=SQRT (XEDOTA**2+YEDOTA**2+ZEDOTA**2)
      VELT=SQRT (XEDOTT**2+YEDOTT**2+ZEDOTT**2)
      DO 575 I =1, 3
      DO 575 J = 1, 3
      DRGR (I, J) = D (I, J)
575 CRGR (I, J) = C (I, J)
      CALL RELGN (XEA, YEA, ZEA, XEDOTA, YEDOTA, ZEDOTA, XET, YET, ZET,
1          XEDOTT, YEDOTT, ZEDOTT, FFLOSA, FFLOST, DT, VELA, VELT)
      FFLOSA= FLOSSA
      FFLOST= FLOSST
      CALL MEINTL
C
C PRINT AND TAPE CYCLE
C
      IF (TIME.GT.TENDD) GO TO 1200
      IESTAT= 0
      KOUN = KOUN+1
      IF (KOUN.GE.KPR) IESTAT= 1
C
      IF (KTAPE.LE.0) GO TO 590
      ITAPE = ITAPE+1
      IF (ITAPE.LT.KTAPE) GO TO 590
      IESTAT= IESTAT+2
590 CONTINUE
      IF (IESTAT.EQ.0) GO TO 1200
      CALL STATEA
C
      CALL STATET
      IF (IESTAT.EQ.2) GO TO 600
      KOUN = 0
      CALL PAIRCR
      CALL PCELL
600 CONTINUE
      IF ( KTAPE .LE. 0 ) GO TO 1200
      IF ( XEA .GT. XYMAX ) XYMAX = XEA
      IF ( YEA .GT. XYMAX ) XYMAX = YEA
      IF ( XET .GT. XYMAX ) XYMAX = XET
      IF ( YET .GT. XYMAX ) XYMAX = YET
      IF ( XEA .LT. XYMIN ) XYMIN = XEA
      IF ( YEA .LT. XYMIN ) XYMIN = YEA
      IF ( XET .LT. XYMIN ) XYMIN = XET

```

```

IF( YET .LT. XYMIN ) XYMIN = YET
ITAPE = 0
ITAPC = ITAPC + 1
ITAPP = ITAPP + 1
VALA = JVALUA
VALT = JVALUT
WRITE(1),TIME,VALA,VALT,FLOSSA,FLOSST,RANGE,RRATE,FMACHA,FMACHT
1      ,XEA,YEA,ZEA,XET,YET,ZET
2      ,PSIA,THETAA,PHIA,PSIT,THETAT,PHIT
1200 CONTINUE
C
TPA    = TPOSA
TPT    = TPOST
IDLEA  = 0
IDLET  = 0
TPOSA  = 17.0
TPOST  = 17.0
C
CALL THRTLA
CALL THRTL T
C
IF(TPOSA.NE.17.0)IDLEA= 1
IF(TPOST.NE.17.0)IDLET= 1
TPOSA  = TPA
TPOST  = TPT
C
CALL REACTA(XET,YET,ZET,D)
CALL REACTT(XEA,YEA,ZEA,C)
C
IF(IDLEA.NE.0)TPOSA= 0.0
IF(IDLET.NE.0)TPOST= 0.0
C
C *** INTEGRATE USING EULERS METHOD
C
XEA=XEA+XEDOTA*DT
YEA=YEA+YEDOTA*DT
ZEA=ZEA+ZEDOTA*DT
XEDOTA=XEDOTA+ACCXEA*DT
YEDOTA=YEDOTA+ACCYEA*DT
ZEDOTA=ZEDOTA+ACCZEA*DT
XET=XET+XEDOTT*DT
YET=YET+YEDOTT*DT
ZET=ZET+ZEDOTT*DT
YEDOTT=YEDOTT+ACCYET*DT
XEDOTT=XEDOTT+ACCXET*DT
ZEDOTT=ZEDOTT+ACCZET*DT
TIME =TIME+DT
IF( TIME .LT. TENDD ) GO TO 500
C
C *** END OF CASE
C -----
C

```

```
IF( KTAPE .LE. 0 ) GO TO 567
NPNT = ITAPC
NPNTP = NPNT
IF(NPNTP.GT.139)NPNTP= MIN0((NPNTP+1)/2,100)
WRITE(46,1504) ITAPC, ITAPP
1504 FORMAT( '0' I5 ' POINTS WRITTEN ON TAPE THIS CASE
1,/,I6,' TOTAL NO. OF POINTS WRITTEN ON TAPE FOR ALL CASES')
NKTIC = ( XYMAX - XYMIN ) / 7000.
```

C

```
REWIND 1
CALL TRAPL
REWIND 1
GO TO 567
END
```


SUBROUTINE AERF4 (XX,YY,IFLG)

C
C

```

COMMON/AEROUT/TIDLEX          ,TMILX          ,TABX
1          ,FLODMX          ,FLODSX
2          ,FMMINX          ,FMMAXX          ,RECANX
3          ,ALPHAX
4          ,CDX          ,SBCDX
COMMON/HASSLE/IABLEA          ,IABLET
COMMON/      K4/DT,TBEGN,TIME,PI,PIDV2,PIDV4,TWOPI,DEGRD,RADDG,G
1          ,VAR(20),IVAR(20),TEND
COMMON/SCALES/KALEA(15),KALEB(15),KALEC(25),KALED(25)
1          ,VAKALA(7),VAKALB(5),VAKALC(14),VAKALD(12)
2          ,DELALT          ,DELACH          ,DELCEL
COMMON/TABLES/CS0(121)          ,RHO0(121)
1          ,TIDLE(7,14,2)          ,TMIL(7,14,2)          ,TAB(7,14,2)
2          ,SURF(2)          ,F4VG(5,12,2)          ,F4SG(5,12,2)
3          ,F4MMI(13,2)          ,F4MMA(13,2)          ,RECANG(10,12,2)
4          ,F4CLA(10,2)          ,F4ALP(10,21,2)          ,CLMAX(14,2)
5          ,F5CLA(10,2)          ,F5ALP(10,21,2)
6          ,F4CDR(10,21,2)          ,SBCDR(14,2)
7          ,F5CDR(10,21,2)
DIMENSION TITUDE(2)
DATA (KALEA(I),I=1,15) /1,1,2,2,3,3,4,4,5,5,6,6,6,6,6/
1, (VAKALA(I),I=1,7) /0.0,10000.0,20000.0,30000.0,40000.0,50000.0,
2          ,60000.0/
DATA (KALEB(I),I=1,15) /1,1,1,2,2,2,3,3,3,4,4,4,4,4,4/
1, (VAKALB(I),I=1,5) /0.0,15000.0,30000.0,45000.0,55000.0/
DATA (KALEC(I),I=1,25) /1,1,1,1,2,2,3,3,4,5,6,7,8,8,9,9,10,10,11
1          ,11,12,12,13,13,13/
2, (VAKALC(I),I=1,14) /0.2,0.4,0.6,0.8,0.9,1.0,1.1,1.2,1.4,1.6
3          ,1.8,2.0,2.2,2.4/
DATA (KALED(I),I=1,25) /1,1,1,1,1,2,2,2,3,4,5,6,7,7,7,8,8,8,9,9,10
2          ,10,11,11,11/
3, (VAKALD(I),I=1,12) /0.2,0.5,0.8,0.9,1.0,1.1,1.2,1.5,1.8,2.0
4          ,2.2,2.4/
DATA DELALT/5000.0/,DELACH/0.1/,DELCEL/0.1/
KRAFT= 1 -ATTACKER ,KRAFT=2 -TARGET
KRAFT = 1
IF(IFLG.LT.0)KRAFT= 2
IFLAG = IABS(IFLG)
GO TO (8000,8000,3000,4000),IFLAG
3000 CONTINUE
ALT = XX
ACH = YY
IF(ALT.LT.0.0)GO TO 8000
IF(ACH.LT.0.0)GO TO 8000
IALT = ALT/DELALT
IALT = IALT+1
IALT = MIN0(IALT,12)
MACH = ACH/DELACH
MACH = MACH+1
MACH = MIN0(MACH,25)

```

C

```

C   G - LIMIT
C
IA   = KALEB (IALT)
MA   = KALED (MACH)
TA   = GGGG (VAKALB (IA) ,ALT)
TM   = GGGG (VAKALD (MA) ,ACH)
G1   = GG (F4VG (IA,MA,KRAFT) ,TA)
FLODMX= GGG (F4VG (IA,MA+1,KRAFT) ,G1,TA, TM)
G1   = GG (F4SG (IA,MA,KRAFT) ,TA)
FLODSX= GGG (F4SG (IA,MA+1,KRAFT) ,G1,TA, TM)

C   RECOVERY ANGLE
C
TA   = AMOD (ALT,DELALT) /DELALT
RECANX= PIDV2
IF (ALT.GE.45000.0)GO TO 3250
IR   = MIN0 (IALT,9)
G1   = GG (RECANG (IR,MA,KRAFT) ,TA)
RECANX= GGG (RECANG (IR,MA+1,KRAFT) ,G1,TA, TM)
3250 CONTINUE

C   MACH MIN , MACH MAX
C
FMMINX= GG (F4MMI (IALT,KRAFT) ,TA)
FMMAXX= GG (F4MMA (IALT,KRAFT) ,TA)

C   THRUST EVALUATION
C
IA   = KALEA (IALT)
MA   = KALEC (MACH)
TA   = GGGG (VAKALA (IA) ,ALT)
TM   = GGGG (VAKALC (MA) ,ACH)

C   G1   = GG (TIDLE (IA,MA,KRAFT) ,TA)
TIDLEX= GGG (TIDLE (IA,MA+1,KRAFT) ,G1,TA, TM)

C   G1   = GG (TMIL (IA,MA,KRAFT) ,TA)
TMILX = GGG (TMIL (IA,MA+1,KRAFT) ,G1,TA, TM)

C   G1   = GG (TAB (IA,MA,KRAFT) ,TA)
TABX  = GGG (TAB (IA,MA+1,KRAFT) ,G1,TA, TM)

C   SAVE ALTITUDE
TITUDE (KRAFT) = ALT

C   RETURN

C   4000 CONTINUE
C
ACH   = XX
CEL   = YY
MACH  = ACH/DELACH
MACH  = MACH+1
MACH  = MIN0 (MACH,25)

```

```

MA      = KALEC (MACH)
TM      = GGGG (VAKALC (MA) ,ACH)
C
C      MAX LIFT TEST
C
      CELAX = GG (CLMAX (MA ,KRAFT) ,TM)
      IF (ABS (CEL) .LT. CELAX) GO TO 4050
      CEL   = SIGN (CELAX-1.0E-06 ,CEL)
      YY    = CEL
4050 CONTINUE
C
C      SPEEDBRAKE DRAG
C
      SBCDX = 0.0
      NDXSB = IABLEA
      IF (KRAFT.EQ.2) NDXSB= IABLET
      IF (NDXSB.NE.0) GO TO 4070
      FRADEL= SBDEFA (TITUDE (KRAFT) ,ACH ,KRAFT)
      SBCDX = GG (SBCDR (MA ,KRAFT) ,TM) *FRADEL
4070 CONTINUE
C
      MACH  = MIN0 (MACH ,20)
      MA    = KALEC (MACH)
      TM    = GGGG (VAKALD (MA) ,ACH)
      ICEL  = ABS (CEL) /DELCEL
      ICEL  = ICEL+1
      ICEL  = MIN0 (ICEL ,20)
      TC    = AMOD (ABS (CEL) ,DELCEL) /DELCEL
      IF (CEL.GE.0.0) GO TO 4150
C
C      ALPHA FOR NEGATIVE CEL
      ALSLAP= 0.0
      SLAPAN= 0.0
      ALPHAX= GG (F4ALP (MA ,1 ,KRAFT) ,TM) +CEL/GG (F4CLA (MA ,KRAFT) ,TM)
      GO TO 4200
4150 CONTINUE
C
C      ALPHA INTERPOLATION
C
      G1     = GG (F4ALP (MA ,ICEL ,KRAFT) ,TM)
      ALPHAX= GGG (F4ALP (MA ,ICEL+1 ,KRAFT) ,G1 ,TM ,TC)
      G1     = GG (F5ALP (MA ,ICEL ,KRAFT) ,TM)
      ALSLAP= GGG (F5ALP (MA ,ICEL+1 ,KRAFT) ,G1 ,TM ,TC)
      SLAPAN= SLAPDF (TITUDE (KRAFT) ,ACH ,KRAFT)
4200 CONTINUE
      ALPHAX= ALPHAX+SLAPAN* (ALSLAP-ALPHAX)
C
C      DRAG COEFFICIENT
C
      G1     = GG (F4CDR (MA ,ICEL ,KRAFT) ,TM)
      CDX    = GGG (F4CDR (MA ,ICEL+1 ,KRAFT) ,G1 ,TM ,TC)
      G1     = GG (F5CDR (MA ,ICEL ,KRAFT) ,TM)
      CDLAPS= GGG (F5CDR (MA ,ICEL+1 ,KRAFT) ,G1 ,TM ,TC)
C

```

```

C      CDX      = CDX+SLAPAN*(CDLAPS-CDX)
C
C      ADD SPEEDBRAKE DRAG INCREMENT
C
C      CDX      = CDX+SBCDX
C
C      RETURN
8000  CONTINUE
      CALL EXIT
      END

```

```

SUBROUTINE CLOSS ( XA, YA, ZA, XT, YT, ZT, XD, YD, ZD, VEL, XLOS )
-----

```

```

C
C
      TX = XT - XA
      TY = YT - YA
      TZ = ZT - ZA
      RAN = SQRT ( TX * TX + TY * TY + TZ * TZ )
      IF(RAN.LT.0.001)RAN= 0.001
      IF(VEL.LT.0.001)VEL= 0.001
      ANUM = XD * TX + YD * TY + ZD *TZ
      ARG=ANUM/(VEL*RAN)
      IF(ARG.GT.1.) ARG=1.
      IF(ARG.LT.-1.) ARG=-1.
      XLOS=ACOS(ARG)
      RETURN
      END

```

```

SUBROUTINE CMTRX(PHI, THETA, PSI, C)
-----

```

```

C
C
      DIMENSION C(3,3)
      CPSI=COS(PHI)
      SPSI=SIN(PHI)
      CTHET=COS(THETA)
      STHET=SIN(THETA)
      CPHI=COS(PHI)
      SPHI=SIN(PHI)
C
      C(1,1)=CTHET*CPSI
      C(1,2)=CTHET*SPSI
      C(1,3)=-STHET
C
      C(2,1)=CPSI*STHET*SPHI-SPSI*CPHI
      C(2,2)=CPSI*CPHI+SPSI*STHET*SPHI
      C(2,3)=CTHET*SPHI
C
      C(3,1)=CPSI*STHET*CPHI+SPSI*SPHI
      C(3,2)=SPSI*STHET*CPHI-CPSI*SPHI
      C(3,3)=CTHET*CPHI
C

```

RETURN
END

SUBROUTINE CSRHO (XX, CS, RHO)

C-----
C
COMMON/ K4/DT, TBEGN, TIME, PI, PIDV2, PIDV4, TWOPI, DEGRD, RADDG, G
1 ,VAR(20), IVAR(20), TEND
COMMON/TABLES/CS0(121) ,RHO0(121)
1 ,TIDLE(7,14,2) ,TMIL(7,14,2) ,TAB(7,14,2)
2 ,SURF(2) ,F4VG(5,12,2) ,F4SG(5,12,2)
3 ,F4MMI(13,2) ,F4MMA(13,2) ,RECANG(10,12,2)
4 ,F4CLA(10,2) ,F4ALP(10,21,2) ,CLMAX(14,2)
5 ,F5CLA(10,2) ,F5ALP(10,21,2)
X=XX

C
C LIMIT ALTITUDE INPUT FOR TABLE INTERPOLATION TO RANGE OF TABLE
C ZERO MINIMUM, 60000 FEET MAXIMUM
IF (X . LE . (0.)) X = 0.1
IF(X.GE.60000.) X=59999.
IS=X*0.002
SX=X*0.002-IS
CS=(1.-SX)*CS0(IS+1)+SX*CS0(IS+2)
RHO=(1.-SX)*RHO0(IS+1)+SX*RHO0(IS+2)
END

SUBROUTINE ERMSG(IFLAG, DUMMY)

C-----
C
COMMON/ K4/DT, TBEGN, TIME, PI, PIDV2, PIDV4, TWOPI, DEGRD, RADDG, G
1 ,VAR(20), IVAR(20), TEND
WRITE(46,900) IFLAG, DUMMY, TIME
900 FORMAT(40H SUBROUTINE ERMSG WAS CALLED WITH IFLAG=,I4,8H, DUMMY=,
1 E15.5,8H AT TIME,F10.3)
RETURN
END

SUBROUTINE FIRCON(KAORT, RANGE, DEVANG, ANGOFF, FIRYES)

C-----
C
INTEGER SECTOR
COMMON/ATTTAR/ANGETA, ANGETT, ANGXIA, ANGXIT, CRGR(3,3), DLOSSA,
1 DLOSST, DRGR(3,3), FAZMSA, FAZMST, FELVSA, FELVST, FLOSSA
2 FLOSST, PSUBSA, PSUBST, RANGX, RRATE, XAINT, XTINA, YAINT,
3 YTINA, ZAINT, ZTINA
COMMON/ K4/DT, TBEGN, TIME, PI, PIDV2, PIDV4, TWOPI, DEGRD, RADDG, G
1 ,VAR(20), IVAR(20), TEND
DIMENSION RANGL(3,2), RANGU(3,2), DEVAN(3,2), ANGOFF(3,2)
DIMENSION SECMM(4,2)
DIMENSION NAME(2)
DATA NAME(1)/4HATKR/, NAME(2)/4HTRGT/

```

C   ATTACKER OPTIONS
DATA (RANGL(I,1),I=1,3)/ 0.0,2000.,2000./
DATA (RANGU(I,1),I=1,3)/3000.,18000.,18000./
DATA (DEVAN(I,1),I=1,3)/0.175,0.7,0.35/
DATA (ANGOF(I,1),I=1,3)/1.,3.142,1.57/
C   TARGET OPTIONS
DATA (RANGL(I,2),I=1,3)/ 0.0,2000.,2000./
DATA (RANGU(I,2),I=1,3)/3000.,18000.,18000./
DATA (DEVAN(I,2),I=1,3)/0.175,0.7,0.35/
DATA (ANGOF(I,2),I=1,3)/1.,3.142,1.57/
DATA ((SECMM(I,J),J=1,2),I=1,4)/8000.,30000.,7000.,30000.,7000.,
127000.,3000.,12000./
C   KR      = 1 - ATTACKER
C   KR      = 2 - TARGET
C   KR      = KAORT
IF(KR.LE.0)GO TO 2000
FIRYES= 0.0
IF(KR.EQ.1) GO TO 3000
C   TEST 3 FIRE OPTIOS DEFINED IN TERMS OF
C   RANGE,DEVIATION AND OFF-ANGLE
DO 1000 LOPT=1,3
IF(RANGE.LT.RANGL(LOPT,KR))GO TO 1000
IF(RANGE.GT.RANGU(LOPT,KR))GO TO 1000
IF(ABS(DEVANG).GT.DEVAN(LOPT,KR))GO TO 1000
IF(ABS(ANGOFF).GT.ANGOF(LOPT,KR))GO TO 1000
FIRYES= 1.0
RETURN
1000 CONTINUE
RETURN
2000 CONTINUE
DO 2500 J=2,2
WRITE(46,900)NAME(J)
DO 2400 LOPT=1,3
WRITE(46,901)RANGL(LOPT,J),RANGU(LOPT,J),DEVAN(LOPT,J),
+           ANGOF(LOPT,J)
2400 CONTINUE
2500 CONTINUE
WRITE(46,4000) (SECMM(I,1),I=1,4),(SECMM(I,2),I=1,4)
4000 FORMAT(//56H                ATTACKERS MISSILE FIRING ENVELO
1PE//61H                SECTOR          1          2          3
2 4//28H                MINIMUM RANGE(FEET),F6.0,3F10.0/28H          MAXIMU
3M RANGE(FEET),F6.0,3F10.0)
RETURN
900 FORMAT(//,40H OPPORTUNITIES TO ACCUMULATE GUN TIME - ,A8,/)
901 FORMAT(F8.1,15H.LE. RANGE .LE.,F8.1,20H ABS(DEVIATION).LE.
1      ,F6.3,20H ABS(OFF-ANGLE).LE.,F6.3)
C IS DEVIATION ANGLE LE 30
3000 IF(FLOSST.LE.30.*RADDG) GO TO 3019
3010 RETURN
C CONVERT AZIMUTH ANGLE TO INTERNAL ANGLE BETA
3019 BETA=ANGXIT
C DETERMINE AZIMUTH SECTOR
3020 IF(BETA-180.*RADDG) 3030,3080,3150
3030 IF(BETA-45.*RADDG) 3040,3040,3050

```

```

3040 SECTOR=1
      GO TO 3100
3050 IF (BETA-90.*RADDG) 3060,3060,3070
3060 SECTOR=2
      GO TO 3100
3070 IF (BETA-135.*RADDG) 3090,3090,3080
3080 SECTOR=4
      GO TO 3100
3090 SECTOR=3
3100 IF (RANGE.LT.SECMM(SECTOR,1))GO TO 3010
      IF (RANGE.GT.SECMM(SECTOR,2))GO TO 3010
      FIRYES=1
      GO TO 3010
C CALCULATE SUPPLEMENTARY ANGLE
3150 BETA=360.*RADDG-ANGXIT
      GO TO 3030
      END

```

```

      SUBROUTINE GETRXN (ROTS,ROTN,ROTX,ICOMD)

```

```

C-----
C
C ROTX IS CALCULATED CLOSEST ANGLE TO ROTN IF DISCRETE INCREMENTS
C OF ROTN ARE ALLOWED
C
      ATES=ABS (ROTS)/ROTN
      ITES=ATES
      ROTX=ITES*ROTN
      IF ((ROTX+(ROTN/2.)).LE.(ABS (ROTS))) GO TO 50
      ICOMD=ITES
      GO TO 100
50 ROTX=ROTX+ROTN
      ICOMD=ITES+1
100 IF (ROTS.GE.0.) RETURN
      ROTX=-ROTX
      ICOMD=-ICOMD
      RETURN
      END

```

```

      SUBROUTINE HUBLO (NDX,LABLAT,TINC,GLEVEL)

```

```

C-----
C
C THIS SUBROUTINE CALCULATES THE BLACK-OUT INDEX
C MAN(1).NE.0 ATTACKER SUBJECT TO BLACK-OUT
C MAN(2).NE.0 TARGET SUBJECT TO BLACK-OUT
C BOND(1) BLACK-OUT INDEX OF ATTACKER
C BOND(2) BLACK-OUT INDEX OF TARGET
C
      COMMON/ K4/DT,TBEGN,TIME,PI,PIDV2,PIDV4,TWOPI,DEGRD,RADDG,G
1      ,VAR(20),IVAR(20),TEND
      DIMENSION BOND(2),KOMNT(2),MAN(2)
      DATA KOMNT(1)/4HATKR/,KOMNT(2)/4HTRGT/
      DATA GMOD/4.99/

```

```

DATA MAN(1)/1/,MAN(2)/1/
IF(NDX.GE.0)GO TO 500
C   INITIALIZATION
   BOND(1)= 0.0
   BOND(2)= 0.0
   IF(MAN(1).NE.0)WRITE(46,910)KOMNT(1)
   IF(MAN(2).NE.0)WRITE(46,910)KOMNT(2)
   RETURN
500 CONTINUE
   KAORT = LABLAT
   IF(MAN(KAORT).EQ.0)RETURN
   KOUNT = -1
510 CONTINUE
C   EVALUATION OF BLACK-OUT INDEX
   IF(KOUNT.GT.0)STOP 7
   KOUNT = KOUNT+1
   BOND(1)=-100.0
   IF(GLEVEL.GT.1.0)BOND(1)=-100.0/GLEVEL
   IF(GLEVEL.LT.5.0)GO TO 550
   BOND(1)=((GLEVEL*0.2)**5.7234)*0.3333333
550 CONTINUE
   BOND(1)= BOND(1)*TINC
   BOND = BOND(1)+BOND(KAORT)
   IF(NDX.NE.0)GO TO 2000
C   ACCUMULATE B-O INDEX
   BOND(KAORT)= AMAX1(0.0,BOND)
   IF(BOND.GT.100.0)WRITE(46,900)KOMNT(KAORT),BOND,TIME
   RETURN
2000 CONTINUE
C   TEST B-O INDEX,DO NOT ACCUMULATE
   IF(BOND.LT. 80.0)RETURN
   IF(KOUNT.NE.0)RETURN
   IF(BOND(1).LT.0.0)RETURN
   WRITE(46,920)KOMNT(KAORT),GLEVEL,GMOD
   GLEVEL= GMOD
   GO TO 510
900 FORMAT(1X,A8,20H BLACK-OUT INDEX = ,F5.1,10H AT TIME ,F10.4)
910 FORMAT(//,1X,A8,22H SUBJECT TO BLACK-OUT )
920 FORMAT(28H HUBLO MODIFIES G-LEVEL OF ,A8,20H TRIAL MANEUVER FROM
1,F6.2,4H TO,F6.2)
END

```

SUBROUTINE INRD

```

C-----
C
C   READS INPUT DATA FOR INDIVIDUAL CASE AND DOES SOME INITIALIZATION
C
COMMON/K411 / KTAPE, KPR, KAPR
COMMON/AEROUT/TIDLEX ,TMILX ,TABX
1 ,FLODMX ,FLODSX
2 ,FMMINX ,FMMAXX ,RECANX
3 ,ALPHAX
4 ,CDX ,SBCDX

```


COMMON/ATTTAR/ANGETA, ANGETT, ANGXIA, ANGXIT, CRGR (3, 3), DLOSSA,
1 DLOSST, DRGR (3, 3), FAZMSA, FAZMST, FELVSA, FELVST, FLOSSA,
2 FLOSST, PSUBSA, PSUBST, RANGE, RRATE, XAINT, XTINA, YAINT,
3 YTINA, ZAINT, ZTINA
COMMON/CELSTA/AN1A, AN2A, AN3A, DTPRA, GLEVLA, ICMNWA, ICOMDA, ICRECA,
1 ISTRGA, JVALUA, LCELLA (40), LVALA (40), LWEGHA (40),
2 NCELLA, NCLSTA, NOWPRA, NTILTA, NTRYA, ROTSA, ROTA, TIMEPA
COMMON/CELSTT/AN1T, AN2T, AN3T, DTPRT, GLEVLT, ICMNWT, ICOMDT, ICRECT,
1 ISTRGT, JVALUT, LCELLT (40), LVALT (40), LWEGHT (40),
2 NCELLT, NCLSTT, NOWPRT, NTILTT, NTRYT, ROTST, ROTT, TIMEPT
COMMON /ERNION/ AB (4, 2)
COMMON/EXTPOA/TIMEXA, DELTA, XZEROT, YZEROT, ZZEROT, XMIN1T, YMIN1T,
2 ZMIN1T, XMIN2T, YMIN2T, ZMIN2T, XEXPT, YEXPT, ZEXPT,
2 XDEXPT, YDEXPT, ZDEXPT, VELEXT, DRCEXT (3, 3)
COMMON/EXTPOT/TIMEXT, DELTT, XZEROA, YZEROA, ZZEROA, XMIN1A, YMIN1A,
1 ZMIN1A, XMIN2A, YMIN2A, ZMIN2A, XEXPA, YEXPA, ZEXPA,
2 XDEXPA, YDEXPA, ZDEXPA, VELEXA, DRCEXA (3, 3)
COMMON/HASSLE/IABLEA , IABLET
COMMON/ K4/DT, TBEGN, TIME, PI, PIDV2, PIDV4, TWOPI, DEGRD, RADDG, G
1 , VAR (20), IVAR (20), TEND
COMMON/PREDIC/CNEW (3, 3), VXNEW, VYNEW, VZNEW, XXNEW, YYNEW, ZZNEW
COMMON/TABLES/CS0 (121) , RHO0 (121)
1 , TIDLE (7, 14, 2) , TMIL (7, 14, 2) , TAB (7, 14, 2)
2 , SURF (2) , F4VG (5, 12, 2) , F4SG (5, 12, 2)
3 , F4MMI (13, 2) , F4MMA (13, 2) , RECANG (10, 12, 2)
4 , F4CLA (10, 2) , F4ALP (10, 21, 2) , CLMAX (14, 2)
5 , F5CLA (10, 2) , F5ALP (10, 21, 2)
6 , F4CDR (10, 21, 2) , SBCDR (14, 2)
7 , F5CDR (10, 21, 2)
COMMON/TRIALA/AN1TRA (10), AN2TRA (10), AN3TRA (10), CPRA (3, 3, 10),
1 DLOSRA (10), DRAGTA (10), DRGPRA (10), DXEPRA (10),
2 DYEPRA (10), DZEPRA (10), FLDTRA (10), FLOSRA (10),
3 ICNPRA (10), ICTRYA (10), ISTPRA (20, 10), ISTRYA (10),
4 IVALTA, IVPRA (10), KNEWA, KRO TSA, PSTRA (10), PXEPRA (10),
5 PYEPRA (10), PZEPRA (10), RNGPRA (10), ROTNCA, ROTN2A,
6 ROTRYA (10), THTRA (10), TPOTRA (10), XEXT, YEXT, ZEXT
COMMON/TRIALT/AN1TRT (10), AN2TRT (10), AN3TRT (10), CPRT (3, 3, 10),
1 DLOSRT (10), DRATT (10), DRGPRT (10), DXEPRT (10),
2 DYEPRT (10), DZEPRT (10), FLDTRT (10), FLOSRT (10),
3 ICNPRT (10), ICTRYT (10), ISTPRT (20, 10), ISTRYT (10),
4 IVALTT, IVPRT (10), KNEWT, KROTST, PSTRT (10), PXEPRT (10),
5 PYEPRT (10), PZEPRT (10), RNGPRT (10), ROTNCT, ROTN2T,
6 ROTRYT (10), THTRT (10), TPOTRT (10), XEXA, YEXA, ZEXA
COMMON/VARBLA/ACCXEA, ACCYEA, ACCZEA, ALFAA, CBARA (3, 3), CBA (3, 3), CDA,
1 CLALFA, CLA, COPHIA, COPSIA, COTHTA , CSA, C (3, 3), DRAGA,
2 FFLOSA, FLIFTA, FLODMA, FMACHA, FMMAXA, FMMINA, INIZA,
3 PHIBRA, PHIA, PMAXA, PPDTA, PA, PSIBRA, PSIA, QMAXA, QPDTA,
4 QA, RECANA, RHOA, RMAXA, RPDTA, RA, SIPHIA, SIPSIA, SITHTA ,
5 SPECEA, SA, TABA, THETA A, THETBA, TIDLEA, TMILA, TPOSA,
6 TRSTA, UA, VELA, VHOA, VA, WEITA, WA, XEDOTA, XEA, YEDOTA,
7 YEA, ZEDOTA, ZEA, FLODSA
COMMON/VARBLT/ACCXET, ACCYET, ACCZET, ALFAT, CBART (3, 3), CBT (3, 3), CDT,
1 CLALFT, CLT, CPHIT, COPSIT, COTHTT , CST, D (3, 3), DRAGT,
2 FFLOST, FLIFTT, FLODMT, FMACHT, FMMAXT, FMINT, INIZT,

```

3          PHIBRT, PHIT, PMAXT, PPDTT, PT, PSIBRT, PSIT, QMAXT, QPDTT,
4          QT, RECANT, RHOT, RMAXT, RPDTT, RT, SIPHIT, SIPSIT, SITHTT,
5          SPECET, ST, TABT, THETAT, THETBT, TIDLET, TMILT, TPOST,
6          TRSTT, UT, VELT, VHORT, VT, WEITT, WT, XEDOTT, XET, YEDOTT,
7          YET, ZEDOTT, ZET, FLODST
DIMENSION ITIT(20)
NCLSTA=15
NCLSTT=15
READ(45,105)(ITIT(J),J=1,20)
105 FORMAT ( 20A4 )
ITES = 4HEND
IF(ITIT(1).NE.ITES) GO TO 3456
WRITE(46,3459)
PRINT 3459
3459 FORMAT(///,22H CALCULATION CYCLE END )
IVAR(2)=1
RETURN
3456 CONTINUE
WRITE(46,106) (ITIT(J), J= 1,20 )
106  FORMAT ( '1' 19X, 20A4 )
C
C *** READ ATTACKER DATA AND DO SOME ATTACKER VARIABLE INITIALIZATION
C -----
C
WRITE(46,119)
119 FORMAT ( '0 ATTACKER DATA ' )
READ(45,*)WEITA
WRITE(46,112)WEITA
112 FORMAT(/,' WEIGHT ',F10.1,' LB',/)
READ(45,*) XEA,YEA, HA, VELA, PSINA,THEINA, PHINA
110 FORMAT ( 8F10.4 )
WRITE(46,120) XEA,YEA, HA, VELA, PSINA,THEINA, PHINA
120 FORMAT( / ' XEA YEA HA VELA PSIA'
1 ' THETA PHIA'
2 / 3F10.2, F10.3, 3F10.3 )
READ(45,*)FSTRGA,GLEVL, TPOSA,DTPRA,TIMEPA,FNTILA
WRITE(46,125)
125 FORMAT ( '0 FSTRGA = +1 STRAIGHT FLIGHT, BELLY DOWN ' /
1 ' FSTRGA = -1 STRAIGHT FLIGHT, BELLY UP ' /
2 ' FSTRGA = 0 TURN INTO A MANEUVER PLANE ' /
3 ' DTPRA = TIME BETWEEN ATTACKER TACTICAL DECISI
3ONS' /
4 ' TIMEPA = ATTACKER FORWARD PREDICTION TIME' )
WRITE(46,130)FSTRGA,GLEVL, TPOSA,DTPRA,TIMEPA,FNTILA
130 FORMAT( '0 FSTRGA GLEVL TPOSA DTPRA TIMEPA'
1 ' FNTILA',/,6F10.3)
ZEA = - HA
PSIA = PSINA * RADDG
THETAA = THEINA * RADDG
PHIA = PHINA * RADDG
ROTA = PHIA
CALL QUATIN(PSIA,THETAA,PHIA,AB(1,1))
VHORA = VELA * COS(THETAA)
XEDOTA = VHORA * COS ( PSIA )

```

```

YEDOTA = VHORA * SIN ( PSIA )
ZEDOTA = - VELA * SIN ( THETAA )
ISTRGA=FSTRGA
NTILTA= FNTILA + 0.00001
ROTNCA=PIDV2/(NTILTA+1)
ROTN2A=ROTNCA/2.
ICOMDA=0
IF(ISTRGA.EQ.0) ICOMDA=ROTA/(ROTNCA-0.001)
IF(ISTRGA.EQ.1) ROTA=0.
IF(ISTRGA.EQ.-1) ROTA=PI
READ(45,*) (LWEGHA(I), I=1, NCLSTA)
5843 FORMAT(16I5)
WRITE(46,5844)
5844 FORMAT(/20X, 'WEIGHT FACTORS FOR ATTACKER DECISION PARAMETERS')
WRITE(46,5845) (I, I=1, NCLSTA)
5845 FORMAT(/' QUESTION' 5I5, 3X, 5I5, 3X, 5I5)
WRITE(46,5846) (LWEGHA(I), I=1, NCLSTA)
5846 FORMAT(/' WEIGHT ' 5I5, 3X, 5I5, 3X, 5I5)
C
C *** READ TARGET DATA AND DO SOME INITIALIZATION FOR TARGET
C -----
C
WRITE(46,1119)
1119 FORMAT ( '0 TARGET DATA ' )
READ(45,*)WEITT
WRITE(46,112)WEITT
READ(45,*) XET, YET, HT, VELT, PSINT, THEINT, PHINT
WRITE(46,140) XET, YET, HT, VELT, PSINT, THEINT, PHINT
140 FORMAT ( / ' XET YET HT VELT PSIT'
1 ' THETT PHIT'
2 / 3F10.2, F10.3, 3F10.3 )
READ(45,*) FSTRGT, GLEVL, TPOST, DTPRT, TIMEPT, FNTILT
WRITE(46,1125)
1125 FORMAT ( '0 FSTRGT = +1 STRAIGHT FLIGHT, BELLY DOWN ' /
1 ' FSTRGT = -1 STRAIGHT FLIGHT, BELLY UP ' /
2 ' FSTRGT = 0 TURN INTO A MANEUVER PLANE ' /
3 ' DTPRT = TIME BETWEEN TARGET TACTICAL DECISION
3S' /
4 ' TIMEPT = TARGET FORWARD PREDICTION TIME' )
WRITE(46,150) FSTRGT, GLEVL, TPOST, DTPRT, TIMEPT, FNTILT
150 FORMAT ( '0 FSTRGT GLEVL TPOST DTPRT TIMEPT'
1 ' FNTILT', /, 6F10.3)
ZET = - HT
PSIT = PSINT * RADDG
THETAT = THEINT * RADDG
PHIT = PHINT * RADDG
ROTT = PHIT
CALL QUATIN(PSIT, THETAT, PHIT, AB(1,2))
VHORT = VELT * COS( THETAT )
XEDOTT = VHORT * COS( PSIT )
YEDOTT = VHORT * SIN ( PSIT )
ZEDOTT = - VELT * SIN ( THETAT )
ISTRGT=FSTRGT
NTILT= FNTILT + 0.00001

```

```

ISTRGT=FSTRGT
ROTNCT=PIDV2/(NTILTT+1)
ROTN2T=ROTNCT/2.
ICOMDT=0
IF(ISTRGT.EQ.1) ROTT=0.
IF(ISTRGT.EQ.-1) ROTT=PI
IF(ISTRGT.EQ.0) ICOMDT=ROTT/(ROTNCT-0.001)

```

C

```

READ(45,*) (LWEGHT(I), I=1, NCLSTT)
WRITE(46,5944)
5944 FORMAT(//20X, 'WEIGHT FACTORS FOR TARGET DECISION PARAMETERS')
WRITE(46,5845) (I, I=1, NCLSTT)
WRITE(46,5846) (LWEGHT(I), I=1, NCLSTT)

```

C

C

C

C

```

*** READ PROGRAM CONTROL CARD
-----

```

```

READ(45,*) TBEGN, TEND, DT, FKPR, FKAPR, FKTAP
WRITE(46,175) TBEGN, TEND, DT, FKPR, FKAPR, FKTAP
175 FORMAT ('0 PROGRAM CONTROL DATA ' / ' TBEGN TEND'
1 ' DT'
2 ' FKPR FKAPR FKTAP '/2F10.3,F10.5,3F10.2)
KPR = FKPR + 0.00001
KAPR = FKAPR + 0.00001
TIME = TBEGN
KTAPE = FKTAP + 0.00001
350 CONTINUE

```

C

```

RETURN
END

```

SUBROUTINE MEINTL

C

C

C

```

TO CALCULATE OFFENSIVE AND GUN TIME
FOR ATTACKER AND TARGET
COMMON/ATTTAR/ANGETA, ANGETT, ANGXIA, ANGXIT, CRGR(3,3), DLOSSA,
1 DLOSST, DRGR(3,3), FAZMSA, FAZMST, FELVSA, FELVST, FLOSSA,
2 FLOSST, PSUBSA, PSUBST, RANGE, RRATE, XAINT, XTINA, YAINT,
3 YTINA, ZAINT, ZTINA
COMMON/ K4/DT, TBEGN, TIME, PI, PIDV2, PIDV4, TWOPI, DEGRD, RADDG, G
1 , VAR(20), IVAR(20), TEND
COMMON/VARBLA/ACCXEA, ACCYEA, ACCZEA, ALFAA, CBARA(3,3), CBA(3,3), CDA,
1 CLALFA, CLA, COPHIA, COPSIA, COTHTA, CSA, C(3,3), DRAGA,
2 FFLOSA, FLIFTA, FLODMA, FMACHA, FMAXA, FMMINA, INIZA,
3 PHIBRA, PHIA, PMAXA, PPDTA, PA, PSIBRA, PSIA, QMAXA, QPDTA,
4 QA, RECANA, RHOA, RMAXA, RPDTA, RA, SIPHIA, SIPSIA, SITHTA,
5 SPECEA, SA, TABA, THETA, THETBA, TIDLEA, TMILA, TPOSA,
6 TRSTA, UA, VELA, Vhora, VA, WEITA, WA, XEDOTA, XEA, YEDOTA,
7 YEA, ZEDOTA, ZEA, FLODSA
COMMON/VARBLT/ACCXET, ACCYET, ACCZET, ALFAT, CBART(3,3), CBT(3,3), CDT,
1 CLALFT, CLT, COPHIT, COPSIT, COTHTT, CST, D(3,3), DRAGT,
2 FFLOST, FLIFTT, FLODMT, FMACHT, FMMAXT, FMINT, INIZT,

```

```

3          PHIBRT,PHIT,PMAXT,PPD TT,PT,PSIBRT,PSIT,QMAXT,QPDTT,
4          QT,RECANT,RHOT,RMAXT,RPD TT,RT,SIPHIT,SIPSIT,SITHTT,
5          SPECET,ST,TABT,THETAT,THETBT,TIDLET,TMILT,TPOST,
6          TRSTT,UT,VELT,VHORT,VT,WEITT,WT,XEDOTT,XET,YEDOTT,
7          YET,ZEDOTT,ZET,FLDST
COMMON/TALLY/ OFTIMA ,OFTIMT ,GUNTMA ,GUNTMT ,ANGOFA,ANGOFT,
1          DEVAA,DEVAT

C
IF (INIZT.LE. 0) GO TO 5000
OFTIMA =0.
OFTIMT =0.
GUNTMA =0.
GUNTMT =0.
SIXTY=60./DEGRD
THIRTY=30./DEGRD
TEN=10./DEGRD
RETURN

C
5000 CONTINUE

C
C*** COMPUTE ATTACKERS OFFENSIVE AND GUN TIME
C
C
C          DEVIATION ANGLE(DEVAA) - BETWEEN LOS AND X-BODY-A
C          OFF-ANGLE(ANGOFA) - BETWEEN LOS AND NEG.X-BODY-T
C
C          DEVAA = ACOS( XTINA/RANGE)
C          ANGOFA= ACOS(-XAIN T/RANGE)
C          IF (ABS(DEVAA) .GT. SIXTY) GO TO 100
C          IF (ABS(ANGOFA) .GT. SIXTY) GO TO 100
C          OFTIMA =OFTIMA +DT
100 CONTINUE

C
C*** CAN ATTACKER FIRE Q
C
C          CALL FIRCON(1,RANGE,DEVAA,ANGOFA,FIT)
C          GUNTMA = GUNTMA +DT*FIT

C
C*** NOW SAME THING FOR TARGET
C
C          DEVAT = PI-ANGOFA
C          ANGOFT= PI-DEVAA
C          IF (ABS(DEVAT) .GT. SIXTY) GO TO 400
C          IF (ABS(ANGOFT) .GT. SIXTY) GO TO 400
C          OFTIMT =OFTIMT + DT
400 CONTINUE
C          CALL FIRCON(2,RANGE,DEVAT,ANGOFT,FIT)
C          GUNTMT = GUNTMT +DT*FIT
C          RETURN
C          END

```

SUBROUTINE NORPLN(X1,Y1,Z1,ROT,A1,B1,C1)

C-----
C

```
V=SQRT(X1**2+Y1**2+Z1**2)
X=X1/V
Y=Y1/V
Z=Z1/V
XQPYQ=X**2+Y**2
FKK=SQRT(XQPYQ)
FK=SQRT(XQPYQ)*COS(ROT)
C1=FKK*SIN(ROT)
B1=(FK*X-C1*Z*Y)/XQPYQ
A1=(-C1*X*Z-FK*Y)/XQPYQ
RETURN
END
```

C

SUBROUTINE OILER(COSIX,PSI,THETA,PHI)

C-----
C

```
DIMENSION COSIX(3,3)
TRANSFORMATION OF DIRECTION-COSINE MATRIX INTO EULER ANGLES
IF(ABS(COSIX(1,3)).GT.1.0)COSIX(1,3)=SIGN(1.0,COSIX(1,3))
THETA =-ASIN(COSIX(1,3))
PSI   = ATAN2(COSIX(1,2),COSIX(1,1))
PHI   = ATAN2(COSIX(2,3),COSIX(3,3))
RETURN
END
```

C

SUBROUTINE PAIRCR

C-----

C

C *** TO PRINT AIRCRAFT DATA

C

```
COMMON/IDS/IDTAG(2)
COMMON/UNITS/LP,CR,TAPE1,TAPE2,TAPE3,TAPE4
INTEGER CR,TAPE1,TAPE2,TAPE3,TAPE4
```

C

```
COMMON/      K4/DT,TBEGN,TIME,PI,PIDV2,PIDV4,TWOPI,DEGRD,RADDG,G
1           ,VAR(20),IVAR(20),TEND
COMMON/ATTTAR/ANGETA,ANGETT,ANGXIA,ANGXIT,CRGR(3,3),DLOSSA,
1           DLOSST,DRGR(3,3),FAZMSA,FAZMST,FELVSA,FELVST,FLOSSA,
2           FLOSST,PSUBSA,PSUBST,RANGE,RRATE,XAINT,XTINA,YAINT,
3           YTINA,ZAINT,ZTINA
COMMON/VARBLA/ACCXEA,ACCYEA,ACCZEA,ALFAA,CBARA(3,3),CBA(3,3),CDA,
1           CLALFA,CLA,COPHIA,COPSIA,COTHTA,CSA,C(3,3),DRAGA,
2           FFLOSA,FLIFTA,FLODMA,FMACHA,FMMAXA,FMINA,INIZA,
3           PHIBRA,PHIA,PMAXA,PPDTA,PA,PSIBRA,PSIA,QMAXA,QPDTA,
4           QA,RECANA,RHOA,RMAXA,RPDTA,RA,SIPHIA,SIPSIA,SITHTA,
5           SPECEA,SA,TABA,THETA,THETBA,TIDLEA,TMILA,TPOSA,
6           TRSTA,UA,VELA,VHORA,VA,WEITA,WA,XEDOTA,XEA,YEDOTA,
```

```

7          YEA, ZEDOTA, ZEA, FLODSA
COMMON/VARBLT/ACCXET, ACCYET, ACCZET, ALFAT, CBART (3, 3), CBT (3, 3), CDT,
1          CLALFT, CLT, COPHIT, COPSIT, COTHTT, CST, D (3, 3), DRAGT,
2          FFLOST, FLIFTT, FLODMT, FMACHT, FMMAXT, FMMINT, INIZT,
3          PHIBRT, PHIT, PMAXT, PPDTT, PT, PSIBRT, PSIT, QMAXT, QPDTT,
4          QT, RECANT, RHOT, RMAXT, RPDTT, RT, SIPHIT, SIPSIT, SITHTT,
5          SPECET, ST, TABT, THETAT, THETBT, TIDLET, TMILT, TPOST,
6          TRSTT, UT, VELT, VHORT, VT, WEITT, WT, XEDOTT, XET, YEDOTT,
7          YET, ZEDOTT, ZET, FLODST

```

C

```

COMMON/CELSTT/AN1T, AN2T, AN3T, DTPRT, GLEVL, ICMNWT, ICOMDT, ICRECT,
1          ISTRGT, JVALUT, LCELLT (40), LVALT (40), LWEGHT (40),
2          NCELLT, NCLSTT, NOWPRT, NTILTT, NTRYT, ROTST, ROTT, TIMEPT

```

C

```

COMMON/CELSTA/AN1A, AN2A, AN3A, DTPRA, GLEVLA, ICMNWA, ICOMDA, ICRECA,
1          ISTRGA, JVALUA, LCELLA (40), LVALA (40), LWEGHA (40),
2          NCELLA, NCLSTA, NOWPRA, NTILTA, NTRYA, ROTSA, ROTA, TIMEPA

```

```

BETA1=ASIN (VA/VELA) *DEGRD
BETA2=ASIN (VT/VELT) *DEGRD
TAS1=VELA*0.5925
CAS1=TAS1*SQRT (RHOA/0.0023768)
TAS2=VELT*0.5925
CAS2=TAS2*SQRT (RHOT/0.0023768)
ALFA1=ALFAA*DEGRD
ALFA2=ALFAT*DEGRD
G1=GLEVLA*FLODMA
G2=GLEVL*FLODMT
THET1=THETAA*DEGRD
THET2=THETAT*DEGRD
PHI1=PHIA*DEGRD
PHI2=PHIT*DEGRD
PSI1=PSIA*DEGRD
PSI2=PSIT*DEGRD
ALT1=-ZEA
ALT2=-ZET
ALT1D=-ZEDOTA
ALT2D=-ZEDOTT
P1=PA*DEGRD
P2=PT*DEGRD
Q1=QA*DEGRD
Q2=QT*DEGRD
R1=RA*DEGRD
R2=RT*DEGRD
SPEC1=SPECEA/1000.
SPEC2=SPECET/1000.
PS1=(TRSTA-DRAGA)*VELA/WEITA
PS2=(TRSTT-DRAGT)*VELT/WEITT

```

C

```

WRITE (46, 190) TIME, IDTAG (1), IDTAG (2), FMACHA, FMACHT, TAS1, TAS2,
1          CAS1, CAS2, ALFA1, ALFA2, BETA1, BETA2, G1, G2, THET1, THET2,
2          PHI1, PHI2, PSI1, PSI2

```

C

```

WRITE (46, 191) XEA, XET, YEA, YET, ALT1, ALT2, XEDOTA, XEDOTT, YEDOTA,
1          YEDOTT, ALT1D, ALT2D, VELA, VELT, P1, P2, Q1, Q2, R1, R2, UA, UT

```

2 ,VA,VT,WA,WT

C WRITE (46,192) FLODMA,FLODMT,FLODSA,FLODST,TPOSA,TPOST,TRSTA,TRSTT,
1 FLIFTA,FLIFTT,DRAGA,DRAGT,CLA,CLT,SPEC1,SPEC2,
2 PS1,PS2

C 190 FORMAT (1H1,///,
*8X,23HAIRCRAFT DATA AT TIME =,F9.4,1X,7HSECONDS,/,
*8X,40 (1H-),///,
1 36X,8HATTACKER,7X,6HTARGET,/,1X,12HAIRCRAFT ID.,
2 25X,A4,10X,A4,///,1X,4HMACH,31X,F8.2,6X,F8.2,/,
3 1X,14HTRUE AIR SPEED,9X,5HKNOTS,7X,F8.2,6X,F8.2,/,
4 1X,19HINDICATED AIR SPEED,4X,5HKNOTS,7X,F8.2,6X,F8.2,///,
5 1X,15HANGLE OF ATTACK,8X,3HDEG,9X,F8.2,6X,F8.2,/,
6 1X,14HSIDESLIP ANGLE,9X,3HDEG,9X,F8.2,6X,F8.2,/,
7 1X,11HLOAD FACTOR,12X,1HG,11X,F8.2,6X,F8.2,///,
8 1X,13HTHETA (PITCH),10X,3HDEG,9X,F8.2,6X,F8.2,/,
9 1X,12HPHI (ROLL),11X,3HDEG,9X,F8.2,6X,F8.2,/,
A 1X,11HPSI (YAW),12X,3HDEG,9X,F8.2,6X,F8.2,/))

C 191 FORMAT (1X,2HXE,21X,4HFEET,8X,F8.1,6X,F8.1,/,
1 1X,2HYE,21X,4HFEET,8X,F8.1,6X,F8.1,/,
2 1X,8HALTITUDE,15X,4HFEET,8X,F8.1,6X,F8.1,///,
3 1X,5HXEDOT,18X,8HFEET/SEC,4X,F8.1,6X,F8.1,/,
4 1X,5HYEDOT,18X,8HFEET/SEC,4X,F8.1,6X,F8.1,/,
5 1X,7HALT DOT,16X,8HFEET/SEC,4X,F8.1,6X,F8.1,/,
6 1X,8HVELOCITY,15X,8HFEET/SEC,4X,F8.1,6X,F8.1,///,
7 1X,1HP,22X,7HDEG/SEC,5X,F8.2,6X,F8.2,/,
8 1X,1HQ,22X,7HDEG/SEC,5X,F8.2,6X,F8.2,/,
9 1X,1HR,22X,7HDEG/SEC,5X,F8.2,6X,F8.2,///,
A 1X,1HU,22X,8HFEET/SEC,4X,F8.1,6X,F8.1,/,
B 1X,1HV,22X,8HFEET/SEC,4X,F8.1,6X,F8.1,/,
C 1X,1HW,22X,8HFEET/SEC,4X,F8.1,6X,F8.1,/))

C 192 FORMAT (1X,19HMAX PERMISS. LDFCT.,4X,1HG,11X,F8.2,6X,F8.2,/,
1 1X,19HSUSTAINED LOAD FCT.,4X,1HG,11X,F8.2,6X,F8.2,///,
2 1X,17HTHROTTLE POSITION,6X,3HN/A,9X,F8.2,6X,F8.2,/,
3 1X,6HTHRUST,17X,6HPOUNDS,6X,F8.1,6X,F8.1,/,
4 1X,4HLIFT,19X,6HPOUNDS,6X,F8.1,6X,F8.1,/,
5 1X,4HDRAG,19X,6HPOUNDS,6X,F8.1,6X,F8.1,///,
6 1X,7HC SUB L,16X,3HN/A,9X,F8.2,6X,F8.2,/,
7 1X,17HSPEC. ENERGY/1000,6X,4HFEET,8X,F8.2,6X,F8.2,/,
8 1X,17HSPEC. ENERGY RATE,6X,8HFEET/SEC,4X,F8.2,6X,F8.2)
END

SUBROUTINE PCELL

C-----
C
C *** TO PRINT TACTICAL SITUATION

C
C
COMMON/IDS/IDTAG(2)
COMMON/UNITS/LP,CR,TAPE1,TAPE2,TAPE3,TAPE4

INTEGER CR,TAPE1,TAPE2,TAPE3,TAPE4

C

```
COMMON/      K4/DT,TBEGN,TIME,PI,PIDV2,PIDV4,TWOPI,DEGRD,RADDG,G
1             ,VAR(20),IVAR(20),TEND
COMMON/ATTTAR/ANGETA,ANGETT,ANGXIA,ANGXIT,CRGR(3,3),DLOSSA,
1             DLOSST,DRGR(3,3),FAZMSA,FAZMST,FELVSA,FELVST,FLOSSA
2             FLOSST,PSUBSA,PSUBST,RANGE,RRATE,XAINT,XTINA,YAINT,
3             YTINA,ZAINT,ZTINA
```

C

```
COMMON/TALLY/ OFTIMA,OFTIMT,GNTIMA,GNTIMT,ANGOFA,ANGOFT,
1             DEVAA,DEVAT
```

C

```
COMMON/CELSTA/AN1A,AN2A,AN3A,DTPRA,GLEVLA,ICMNA,ICOMDA,ICRECA,
1             ISTRGA,JVALUA,LCELLA(40),LVALA(40),LWEGHA(40),
2             NCELLA,NCLSTA,NOWPRA,NTILTA,NTRYA,ROTTA,ROTA,TIMEPA
```

C

```
COMMON/CELSTT/AN1T,AN2T,AN3T,DTPRT,GLEVLT,ICMNT,ICOMDT,ICRECT,
1             ISTRGT,JVALUT,LCELLT(40),LVALT(40),LWEGHT(40),
2             NCELLT,NCLSTT,NOWPRT,NTILTT,NTRYT,ROTTST,ROTT,TIMEPT
```

```
COMMON/VARBLA/ACCXEA,ACCYEA,ACCZEA,ALFAA,CBARA(3,3),CBA(3,3),CDA,
1             CLALFA,CLA,COPHIA,COPSIA,COTHTA,CSA,C(3,3),DRAGA,
2             FFLOSA,FLIFTA,FLODMA,FMACHA,FMMAXA,FMMINA,INIZA,
3             PHIBRA,PHIA,PMAXA,PPDTA,PA,PSIBRA,PSIA,QMAXA,QPDTA,
4             QA,RECANA,RHOA,RMAXA,RPDTA,RA,SIPHIA,SIPSIA,SITHTA
5             SPECEA,SA,TABA,THETAA,THETBA,TIDLEA,TMILA,TPOSA,
6             TRSTA,UA,VELA,VHORA,VA,WEITA,WA,XEDOTA,XEA,YEDOTA,
7             YEA,ZEDOTA,ZEA,FLODSA
```

```
COMMON/VARBLT/ACCXET,ACCYET,ACCZET,ALFAT,CBART(3,3),CBT(3,3),CDT,
1             CLALFT,CLT,COPHIT,COPSIT,COTHTT,CST,D(3,3),DRAGT,
2             FFLOST,FLIFTT,FLODMT,FMACHT,FMMAXT,FMMINT,INIZT,
3             PHIBRT,PHIT,PMAXT,PPDPT,PT,PSIBRT,PSIT,QMAXT,QPDTT,
4             QT,RECANT,RHOT,RMAXT,RPDPT,RT,SIPHIT,SIPSIT,SITHTT
5             SPECET,ST,TABT,THETAT,THETBT,TIDLET,TMILT,TPOST,
6             TRSTT,UT,VELT,VHORT,VT,WEITT,WT,XEDOTT,XET,YEDOTT,
7             YET,ZEDOTT,ZET,FLODST
```

C

```
ANGX1=ANGXIA*DEGRD
ANGX2=ANGXIT*DEGRD
ANGE1=ANGETA*DEGRD
ANGE2=ANGETT*DEGRD
ANGT1=ACOS(COS(ANGXIA)*COS(ANGETA))
ANGT1=ANGT1*DEGRD
ANGT2=ACOS(COS(ANGXIT)*COS(ANGETT))
ANGT2=ANGT2*DEGRD
FLOSS1=FLOSSA*DEGRD
FLOSS2=FLOSST*DEGRD
DLOSS1=DLOSSA*DEGRD
DLOSS2=DLOSST*DEGRD
ANGOF1=ANGOFA*DEGRD
ANGOF2=ANGOFT*DEGRD
```

C

```
TMSL1A=0.
TMSL2A=0.
TMSL1T=0.
```

TMSL2T=0.

```
WRITE (46,190) TIME, IDTAG (1), IDTAG (2), RANGE, RRATE,  
1 ANGT1, ANGT2, ANGX1, ANGX2, ANGE1, ANGE2,  
2 FLOSS1, FLOSS2, DLOSS1, DLOSS2, ANGOF1, ANGOF2,  
3 OPTIMA, OPTIMT, GNTIMA, GNTIMT, TMSL1A, TMSL1T,  
4 TMSL2A, TMSL2T
```

```
WRITE (46,191) (LVALA (J), LVALT (J), J=1,15), NCELLA, NCELLT, JVALUA,  
1 JVALUT
```

```
190 FORMAT (1H1, ///, 6X, 28HTACTICAL SITUATION AT TIME =, F9.4, 1X,  
1 7HSECONDS, /, 6X, 45 (1H-), ///,  
2 44X, 8HATTACKER, 5X, 6HTARGET, /,  
3 1X, 12HAIRCRAFT ID., 33X, A4, 8X, A4, ///,  
4 1X, 5HRANGE, 27X, 4HFEET, 13X, F10.1, /,  
5 1X, 10HRANGE RATE, 22X, 8HFEET/SEC, 9X, F10.1, ///,  
6 1X, 26HLINE OF SIGHT ANGLE (LOS), 6X, 3HDEG, 7X, F9.2, 3X, F9.2, /,  
7 1X, 35HAZIMUTH OF LOS (IN BODY AXES) DEG, 7X, F9.2, 3X, F9.2, /,  
8 1X, 35HELEVATION OF LOS (IN BODY AXES) DEG, 7X, F9.2, 3X, F9.2, /,  
9 1X, 16HDEVIATION ANGLE, 16X, 3HDEG, 7X, F9.2, 3X, F9.2, /,  
A 1X, 21HDEVIATION ANGLE RATE, 11X, 7HDEG/SEC, 3X, F9.2, 3X, F9.2, /,  
B 1X, 9HANGLE OFF, 23X, 3HDEG, 7X, F9.2, 3X, F9.2, /,  
C 1X, 26HACCUMULATED OFFENSIVE TIME, 6X, 3HSEC, 7X, F9.4, 3X, F9.4, /,  
D 1X, 35HACCUMULATED TIME FOR WEAPON 1 SEC, 7X, F9.4, 3X, F9.4, /,  
E 1X, 35HACCUMULATED TIME FOR WEAPON 2 SEC, 7X, F9.4, 3X, F9.4, /,  
F 1X, 35HACCUMULATED TIME FOR WEAPON 3 SEC, 7X, F9.4, 3X, F9.4, ///)
```

```
191 FORMAT (  
1 1X, 29H01 IS OPPONENT IN FRONT OF ME, 16X, I2, 12X, I2, /,  
2 1X, 23H02 AM I BEHIND OPPONENT, 22X, I2, 12X, I2, /,  
3 1X, 21H03 CAN I SEE OPPONENT, 24X, I2, 12X, I2, /,  
4 1X, 26H04 CAN OPPONENT NOT SEE ME, 19X, I2, 12X, I2, ///,  
5 1X, 25H05 CAN I FIRE AT OPPONENT, 20X, I2, 12X, I2, /,  
6 1X, 30H06 CAN OPPONENT NOT FIRE AT ME, 15X, I2, 12X, I2, /,  
7 1X, 40H07 IS OPPONENT WITHIN CERTAIN CONE OF ME, 5X, I2, 12X, I2, /,  
8 1X, 40H08 AM I OUTSIDE CERTAIN CONE OF OPPONENT, 5X, I2, 12X, I2, ///,  
9 1X, 28H09 IS CLOSURE RATE FAVORABLE, 17X, I2, 12X, I2, /,  
A 1X, 39H10 CAN I STAY IN WEAPONS DEL. ENVELOPE, 6X, I2, 12X, I2, /,  
B 1X, 38H11 NOT DEFENSIVE OR OPPONENT CAN'T GET, 7X, I2, 12X, I2, /,  
C 1X, 39H12 NOT DEFENSIVE OR OPPONENT CAN'T STAY, 6X, I2, 12X, I2, ///,  
D 1X, 36H13 LINE OF SIGHT ANGLE WITHIN LIMITS, 9X, I2, 12X, I2, /,  
E 1X, 28H14 RATE OF LOS WITHIN LIMITS, 17X, I2, 12X, I2, /,  
F 1X, 34H15 WILL I HAVE AN ENERGY ADVANTAGE, 11X, I2, 12X, I2, ///,  
G 1X, 11HCELL NUMBER, 29X, I7, 7X, I7, ///,  
H 1X, 10HCELL VALUE, 35X, I2, 12X, I2)  
END
```

SUBROUTINE PCSRO

```
PCSRO PRINTS THE INPUT DATA FOR VELOCITY OF SOUND, CS0, AND  
AIR DENSITY, RHO0
```

```

COMMON/      K4/DT, TBEGN, TIME, PI, PIDV2, PIDV4, TWOPI, DEGRD, RADDG, G
1            ,VAR(20), IVAR(20), TEND
COMMON/TABLES/CS0(121)      ,RHO0(121)
1            ,TIDLE(7,14,2)  ,TMIL(7,14,2)      ,TAB(7,14,2)
2            ,SURF(2)        ,F4VG(5,12,2)     ,F4SG(5,12,2)
3            ,F4MMI(13,2)    ,F4MMA(13,2)     ,RECANG(10,12,2)
4            ,F4CLA(10,2)    ,F4ALP(10,21,2)  ,CLMAX(14,2)
5            ,F5CLA(10,2)    ,F5ALP(10,21,2)
6            ,F4CDR(10,21,2) ,SBCDR(14,2)
7            ,F5CDR(10,21,2)

WRITE(46,132)
132 FORMAT(1H1,20X,'SPEED OF SOUND AND AIR DENSITY TABLES FOR STANDAR
1 ATMOSPHERE'//
2            21X,'-----'
3-----'//
4' SPEED OF SOUND (FEET/SEC)'/
5' -----'//)

WRITE(46,139)
139 FORMAT('      TABLE HAS 121 POINTS, ALTITUDE INCREMENT IS 500 FEE
1, ALTITUDE REGIME IS ZERO TO 60,000 FEET.')
WRITE(46,136) (CS0(I), I=1, 121)
136 FORMAT( / 10F8.2 '      0 TO 4500 FEET' / 10F8.2
1 ' 5000 TO 9500 FEET' / 10F8.2 ' 10000 TO 14500 FEET'/
2 10F8.2' 15000 TO 19500 FEET' / 10F8.2 ' 20000 TO 24500 FEET'
3 //10F8.2 ' 25000 TO 29500 FEET' / 10F8.2 ' 30000 TO 34500 FE
4T' / 10F8.2 ' 35000 TO 39500 FEET' / 10F8.2 ' 40000 TO 4450
5 FEET' / 10F8.2 ' 45000 TO 49500 FEET' //10F8.2 ' 50000 TO 5
6500 FEET' / 10F8.2 ' 55000 TO 59500 FEET' / F8.2 , 72X,
7 ' 60000' 11X 'FEET' )

WRITE(46,138)
138 FORMAT('0AIR DENSITY, RHO0, SLUGS /FT CUBED, AS A FUNCTION OF AL
LITUDE')
WRITE(46,120)
120 FORMAT(' -----' )
WRITE(46,139)
WRITE(46,140) ( RHO0(I), I = 1, 121 )
140 FORMAT( / 10F9.7 '      0 TO 4500 FEET' / 10F9.7
1 ' 5000 TO 9500 FEET' / 10F9.7 ' 10000 TO 14500 FEET'/
2 10F9.7' 15000 TO 19500 FEET' / 10F9.7 ' 20000 TO 24500 FEET'
3 //10F9.7 ' 25000 TO 29500 FEET' / 10F9.7 ' 30000 TO 34500 FE
4T' / 10F9.7 ' 35000 TO 39500 FEET' / 10F9.7 ' 40000 TO 4450
5 FEET' / 10F9.7 ' 45000 TO 49500 FEET' //10F9.7 ' 50000 TO 5
6500 FEET' / 10F9.7 ' 55000 TO 59500 FEET' / F9.7 , 81X,
7 ' 60000' 11X 'FEET' )

C
RETURN
END

SUBROUTINE PRESR(TIMPR, ISOM, V2, XX, YY, ZZ, TSTHD, TCTHD, TSPSD, TCPSD)
C-----
C
C PREDICTION OF OWN FLIGHT PATH FOR STRAIGHT FLIGHT
C TIMPR = PREDICTION TIME ISOM.. 1=UPRIGHT 2= INVERTED

```

```

C   V2 = PRESENT VELOCITY      TSTHD = SIN THETA ETC OF VELOCITY VECT.
COMMON/      K4/DT,TBEGN,TIME,PI,PIDV2,PIDV4,TWOPI,DEGRD,RADDG,G
1      ,VAR(20),IVAR(20),TEND
COMMON/PREDIC/CNEW(3,3),VXNEW,VYNEW,VZNEW,XXNEW,YYNEW,ZZNEW

C
T = TIMPR
V = V2
SITHE = TSTHD
COTHE = TCTHD
COPSI=TCPSD
SIPSI=TSPSD
SIPHI = 0.
IF (ISOM .LT. 0 ) GO TO 90
COPHI = 1.
GO TO 100
90 COPHI = -1.
100 CONTINUE
CNEW(1,1) = COTHE * COPSI
CNEW(1,2) = COTHE * SIPSI
CNEW(1,3) = -SITHE
CNW21 = - COPHI * SIPSI
CNW22 = COPHI * COPSI
CNEW(2,1) = CNW21
CNEW(2,2) = CNW22
CNEW(2,3) = 0.
CNEW(3,1) = CNW22 * SITHE
CNEW(3,2) = - CNW21 * SITHE
CNEW(3,3) = COTHE * COPHI
VXNEW = CNEW(1,1) * V
VYNEW = CNEW(1,2) * V
VZNEW = CNEW(1,3) * V
XXNEW = XX + VXNEW * T
YYNEW = YY + VYNEW * T
ZZNEW = ZZ + VZNEW * T
RETURN
END

```

SUBROUTINE PRTF4 (IDTAG,MANY)

```

C-----
COMMON/SCALES/KALEA (15),KALEB (15),KALEC (25),KALED (25)
1      ,VAKALA ( 7),VAKALB ( 5),VAKALC (14),VAKALD (12)
2      ,DELALT      ,DELACH      ,DELCEL
COMMON/TABLES/CS0 (121)      ,RH00 (121)
1      ,TIDLE (7,14,2)      ,TMIL (7,14,2)      ,TAB (7,14,2)
2      ,SURF (2)      ,F4VG (5,12,2)      ,F4SG (5,12,2)
3      ,F4MMI (13,2)      ,F4MMA (13,2)      ,RECANG (10,12,2)
4      ,F4CLA (10,2)      ,F4ALP (10,21,2)      ,CLMAX (14,2)
5      ,F5CLA (10,2)      ,F5ALP (10,21,2)
6      ,F4CDR (10,21,2)      ,SBCDR (14,2)
7      ,F5CDR (10,21,2)
DIMENSION IDTAG (2),LABEL (2)
DATA LABEL (2)/4HTRGT/,LABEL (1)/4HATKR/
IF (MANY.LE.1) LABEL (1)=4H

```

```

DO 7654 KEE=1,MANY
WRITE(46,901)IDTAG(KEE),LABEL(KEE)
WRITE(46,907)SURF(KEE)
WRITE(46,902)
WRITE(46,903)(VAKALD(J),J=1,12)
WRITE(46,904)
DO 110 I=1,5
WRITE(46,905)VAKALB(I),(F4VG(I,J,KEE),J=1,12)
110 CONTINUE
WRITE(46,906)
WRITE(46,903)(VAKALD(J),J=1,12)
WRITE(46,904)
DO 120 I=1,5
WRITE(46,905)VAKALB(I),(F4SG(I,J,KEE),J=1,12)
120 CONTINUE
WRITE(46,911)
WRITE(46,912)
WRITE(46,913)(VAKALA(J),J=1,7)
WRITE(46,914)
DO 210 J=1,14
WRITE(46,915)VAKALC(J),(TIDLE(I,J,KEE),I=1,7)
210 CONTINUE
WRITE(46,916)
WRITE(46,913)(VAKALA(J),J=1,7)
WRITE(46,914)
DO 220 J=1,14
WRITE(46,915)VAKALC(J),(TMIL(I,J,KEE),I=1,7)
220 CONTINUE
WRITE(46,917)
WRITE(46,913)(VAKALA(J),J=1,7)
WRITE(46,914)
DO 230 J=1,14
WRITE(46,915)VAKALC(J),(TAB(I,J,KEE),I=1,7)
230 CONTINUE
WRITE(46,921)
WRITE(46,922)(VAKALD(J),J=1,10)
WRITE(46,923)
CL = 0.0
DO 310 J=1,21
WRITE(46,924)CL,(F4ALP(I,J,KEE),I=1,10)
CL = CL+DELCEL
310 CONTINUE
WRITE(46,925)
WRITE(46,922)(VAKALD(J),J=1,10)
WRITE(46,926)(F4CLA(J,KEE),J=1,10)
WRITE(46,927)
WRITE(46,922)(VAKALD(J),J=1,10)
WRITE(46,923)
CL = 0.0
DO 320 J=1,21
WRITE(46,924)CL,(F5ALP(I,J,KEE),I=1,10)
CL = CL+DELCEL
320 CONTINUE
WRITE(46,928)

```

```

WRITE(46,922)(VAKALD(J),J=1,10)
WRITE(46,926)(F5CLA(J,KEE),J=1,10)
WRITE(46,931)
WRITE(46,922)(VAKALD(J),J=1,10)
WRITE(46,923)
CL      = 0.0
DO 410 J=1,21
WRITE(46,924)CL,(F4CDR(I,J,KEE),I=1,10)
CL      = CL+DELCEL
410 CONTINUE
WRITE(46,932)
WRITE(46,922)(VAKALD(J),J=1,10)
WRITE(46,923)
CL      = 0.0
DO 420 J=1,21
WRITE(46,924)CL,(F5CDR(I,J,KEE),I=1,10)
CL      = CL+DELCEL
420 CONTINUE
WRITE(46,941)
WRITE(46,942)(VAKALC(J),J=1,14)
WRITE(46,943)(CLMAX(J,KEE),J=1,14)
WRITE(46,944)
WRITE(46,945)
ALT     = 0.0
DO 510 I=1,13
WRITE(46,946)ALT,F4MMI(I,KEE),F4MMA(I,KEE)
ALT     = ALT+DELALT
510 CONTINUE
WRITE(46,951)
WRITE(46,903)(VAKALD(J),J=1,12)
WRITE(46,904)
ALT     = 0.0
DO 610 I=1,10
WRITE(46,952)ALT,(RECANG(I,J,KEE),J=1,12)
ALT     = ALT+DELALT
610 CONTINUE
WRITE(46,953)
WRITE(46,942)(VAKALC(J),J=1,14)
WRITE(46,954)(SBCDR(J,KEE),J=1,14)
7654 CONTINUE
RETURN
901 FORMAT(1H1,40X,30HAERODYNAMIC AND ENGINE DATA ,A4,4X,A4///)
902 FORMAT(///,20X,'MAXIMUM LOAD FACTOR(AFTERBURNER-ON) F4VG( G )',/)
903 FORMAT(10X,' MACH NO',12F 8.2)
904 FORMAT(' ALTITUDE',/)
905 FORMAT(F9.0,9X,12F 8.1)
906 FORMAT(///,20X,'SUSTAINED LOAD FACTOR(AFT BURN-ON) F4SG ( G )',/)
907 FORMAT(41X,'REFERENCE AREA',F8.1,' SQFT')
911 FORMAT(///,20X,'THRUST TABLES FOR 1/2 OF PLANE S ENGINES',/)
912 FORMAT( //,20X,'IDLE THRUST TIDLE (LB)',/)
913 FORMAT(11X,'ALTITUDE',7F12.0)
914 FORMAT(' MACH NO',/)
915 FORMAT(F8.1,11X,7F12.0)
916 FORMAT(1H1,20X,'MILITARY THRUST TMIL (LB)',/)

```

```

917 FORMAT(///,20X,'AFTERBURNER THRUST TAB (LB)',/)
921 FORMAT(1H1,20X,'ANGLE OF ATTACK (NO HI LIFT DEVICES) F4ALP',/)
922 FORMAT(11X,'MACH NO',10F10.1)
923 FORMAT(' LIFT COEFF',/)
924 FORMAT(F11.1,7X,10F10.4)
925 FORMAT(///,20X,'CL SUB ALPHA F4CLA (FOR NEG. CL)',/)
926 FORMAT(/,18X,10F10.4)
927 FORMAT(1H1,20X,'ANGLE OF ATTACK (WITH HI LIFT DEVICES) F5ALP',/)
928 FORMAT(///,20X,' CL SUB ALPHA F5CLA (FOR NEG. CL)',/)
931 FORMAT(1H1,20X,'DRAG COEFFICIENT (NO HI LIFT DEVICES) F4CDR',/)
932 FORMAT( //,20X,'CD (WITH HI LIFT DEVICES) F5CDR',/)
941 FORMAT(1H1,20X,'CLMAX',/)
942 FORMAT(' MACH NO',10X,14F6.1,/)
943 FORMAT(18X,14F6.2)
944 FORMAT(///,20X,'MIN AND MAX MACH NUMBERS F4MMI,F4MMA',/)
945 FORMAT(' ALTITUDE M-MIN M-MAX',/)
946 FORMAT(F9.0,2F10.2)
951 FORMAT(///,20X,'DIVE RECOVERY ANGLE RECANG',/)
952 FORMAT(F9.0,9X,12F 8.4)
953 FORMAT(///,20X,'SPEED BRAKE DRAG COEFF(FULL DEFLECTION) SBCDR',/)
954 FORMAT(18X,14F6.4)
END

```

```

SUBROUTINE PRETNW (AN1,AN2,AN3,CBRTR,ANORG,TIMEP,XEDOT,YEDOT,
1 ZEDOT,XET,YET,ZET,VELT)

```

```

C-----
C
COMMON/ K4/DT,TBEGN,TIME,PI,PIDV2,PIDV4,TWOPI,DEGRD,RADDG,G
1 ,VAR(20),IVAR(20),TEND
COMMON/PREDIC/CNEW(3,3),VXNEW,VYNEW,VZNEW,XXNEW,YYNEW,ZZNEW
DIMENSION XYZPP(3),DELX(3),CBRTR(3,3)
ANORM=ANORG*G

```

```

C
C *** NEW COORDINATES IN TRIAL MANEUVER PLANE SYSTEM
RAN=VELT**2/ANORM
OMEG=VELT/RAN
CAPOM=OMEG*TIMEP
XYZPP(1)=RAN*SIN(CAPOM)
XYZPP(2)=0.
XYZPP(3)=-RAN*(1.-COS(CAPOM))

```

```

C
C *** COORDINATE CHANGES IN INERTIAL SYSTEM
C
DO 10 I=1,3
DELX(I)=0.
DO 10 J=1,3
10 DELX(I)=DELX(I)+CBRTR(J,I)*XYZPP(J)
XXNEW=XET+DELX(1)
YYNEW=YET+DELX(2)
ZZNEW=ZET+DELX(3)

```

```

C
C *** NOW GET VELOCITY AND DIRECTION COSINE MATRIX AT PREDICTED POINT
C

```

```

CSOM=COS (CAPOM)
SINOM=SIN (CAPOM)
VXNEW=VELT* (CBRTR (1,1)*CSOM-CBRTR (3,1)*SINOM)
VYNEW=VELT* (CBRTR (1,2)*CSOM-CBRTR (3,2)*SINOM)
VZNEW=VELT* (CBRTR (1,3)*CSOM-CBRTR (3,3)*SINOM)

```

```

C
C *** NOW OBTAIN PREDICTED DIRECTION COSINE MATRIX
C

```

```

PSIBP=ATAN2 (VYNEW, VXNEW)
VHPR=SQRT (VXNEW**2+VYNEW**2)
THETP=ATAN (-VZNEW/VHPR)
PHIBP=ATAN2 ( (AN3/COS (THETP)) , (AN2*COS (PSIBP)-AN1*SIN (PSIBP)) )
CALL CMTRX (PSIBP, THETP, PHIBP, CNEW)
RETURN
END

```

```

SUBROUTINE QUATEX (B, ATRIX)

```

```

C-----
C

```

```

DIMENSION ATRIX (3,3), B (1)

```

```

C
C TRANSFORMATION OF QUATERNIONS INTO DIRECTION-COSINE MATRIX
C

```

```

ATRIX (1,1) = B (3)*B (4)
ATRIX (2,1) = B (1)*B (2)
ATRIX (1,2) = (ATRIX (1,1)+ATRIX (2,1))*2.0
ATRIX (2,1) = (ATRIX (1,1)-ATRIX (2,1))*2.0
ATRIX (1,1) = B (2)*B (4)
ATRIX (3,1) = -B (1)*B (3)
ATRIX (1,3) = (ATRIX (1,1)+ATRIX (3,1))*2.0
ATRIX (3,1) = (ATRIX (1,1)-ATRIX (3,1))*2.0
ATRIX (1,1) = B (2)*B (3)
ATRIX (3,2) = B (1)*B (4)
ATRIX (2,3) = (ATRIX (1,1)+ATRIX (3,2))*2.0
ATRIX (3,2) = (ATRIX (1,1)-ATRIX (3,2))*2.0
ATRIX (3,3) = (B (3)+B (4)) * (B (3)-B (4))
ATRIX (2,2) = (B (1)+B (2)) * (B (1)-B (2))
ATRIX (1,1) = ATRIX (2,2)-ATRIX (3,3)
ATRIX (2,2) = ATRIX (2,2)+ATRIX (3,3)
ATRIX (3,3) = (B (1)+B (3)) * (B (1)-B (3)) + (B (2)+B (4)) * (B (2)-B (4))
RETURN
END

```

```

SUBROUTINE QUATIN (PSI, THE, PHI, UAT)

```

```

C-----
C

```

```

DIMENSION UAT (1)

```

```

C
C TRANSFORMATION OF EULER ANGLES INTO QUATERNIONS
C

```

```

PSIH = PSI*0.5
THEH = THE*0.5

```



```

PHIH = PHI*0.5
CT = COS (THEH)
ST = SIN (THEH)
CP = COS (PHIH)
SP = SIN (PHIH)
PHIH = CT*CP
CT = CT*SP
CP = ST*CP
THEH = ST*SP
ST = COS (PSIH)
SP = SIN (PSIH)
UAT (1) = ST*PHIH+SP*THEH
UAT (2) = -ST*THEH+SP*PHIH
UAT (3) = ST*CP+SP*CT
UAT (4) = ST*CT-SP*CP
RETURN
END

```

```

SUBROUTINE RELGN (XE1, YE1, ZE1, XEDOT1, YEDOT1, ZEDOT1, XE2, YE2, ZE2,
1 XEDOT2, YEDOT2, ZEDOT2, FLOSM1, FLOSM2, TM, VVEL1, VVEL2)

```

C
C
C
C
C

```

-----
COMPUTES RELATIVE GEOMETRY

```

```

FLOSM IS LOS ANGLE OBSERVED TM AGO (USED FOR LOS RATE)

```

C
1
1
2
3

```

COMMON/      K4/DT, TBEGN, TIME, PI, PIDV2, PIDV4, TWOPI, DEGRD, RADDG, G
1             , VAR (20), IVAR (20), TEND
COMMON/ATTTAR/ANGETA, ANGETT, ANGXIA, ANGXIT, CRGR (3, 3), DLOSSA,
1             DLOSST, DRGR (3, 3), FAZMSA, FAZMST, FELVSA, FELVST, FLOSSA,
2             FLOSST, PSUBSA, PSUBST, RANGE, RRATE, XAINT, XTINA, YAINT,
3             YTINA, ZAINT, ZTINA

```

C

```

TAGO = TM
TAXE=XE2-XE1
TAYE=YE2-YE1
TAZE=ZE2-ZE1
TAXDE=XEDOT2-XEDOT1
TAYDE=YEDOT2-YEDOT1
TAZDE=ZEDOT2-ZEDOT1
RANGE= SQRT ( TAXE * TAXE + TAYE * TAYE + TAZE * TAZE )
RRATE  = ( TAXE * TAXDE + TAYE * TAYDE + TAZE * TAZDE ) / RANGE
XAINT  = - ( DRGR (1,1) * TAXE + DRGR (1,2) * TAYE + DRGR (1,3) * TAZE )
YAINT  = - ( DRGR (2,1) * TAXE + DRGR (2,2) * TAYE + DRGR (2,3) * TAZE )
ZAINT  = - ( DRGR (3,1) * TAXE + DRGR (3,2) * TAYE + DRGR (3,3) * TAZE )

```

C

```

XTINA = CRGR (1,1) * TAXE + CRGR (1,2) * TAYE + CRGR (1,3) * TAZE
YTINA = CRGR (2,1) * TAXE + CRGR (2,2) * TAYE + CRGR (2,3) * TAZE
ZTINA = CRGR (3,1) * TAXE + CRGR (3,2) * TAYE + CRGR (3,3) * TAZE

```

C

```

ANGETA =LOS AZIMUTH          ANGXIA = LOS ELEVATION
ANGETA = ASIN ( -ZTINA / RANGE )
ANGXIA =          ATAN2 ( YTINA ,          XTINA )
ANGETT = ASIN ( - ZAINT /RANGE )

```

```

ANGXIT = ATAN2 ( YAINIT , XAINIT )
CALL CLOSS (XE1, YE1, ZE1, XE2, YE2, ZE2, XEDOT1, YEDOT1, ZEDOT1,
1          VVEL1, FLOSSA)
CALL CLOSS (XE2, YE2, ZE2, XE1, YE1, ZE1, XEDOT2, YEDOT2, ZEDOT2,
1          VVEL2, FLOSST)
DLOSSA= (FLOSSA-FLOSM1)/TAGO
DLOSST= (FLOSST-FLOSM2)/TAGO
RETURN
END

```

```

SUBROUTINE TRAFER (AT, OB, ITMES)

```

```

C-----
C

```

```

DIMENSION AT (1), OB (1)
NN          = ITMES
DO 100 I=1, NN
OB (I) = AT (I)
100 CONTINUE
RETURN
END

```

```

SUBROUTINE TRAPL

```

```

C-----

```

```

COMMON/   TER/NPNT, NPNT, NLAB, NKTIC, KPRT, DHOR, DVER
REWIND 2
WRITE (46, 7543) NPNT, NPNT, NLAB, NKTIC, KPRT, DHOR, DVER
7543 FORMAT (///, ' NPNT =', I5, 5X, ' NPNT =', I5, 5X, ' NLAB =', I5, /,
+          ' NKTIC =', I5, 5X,
+          ' KPRT =', I5, 5X, ' DHOR =', F12.3, 5X, ' DVER =', F12.3//)
RETURN
END

```

SUBROUTINE EQMOTT

C-----
C
C *** THIS SUBROUTINE CALCULATES FOR A GIVEN MANEUVER STATUS AND A GIVEN
C *** VELOCITY VECTOR THE ACCELERATIONS OF THE CG OF THE AIRCRAFT IN
C *** THE INERTIAL AXIS SYSTEM. IT ALSO CALCULATES THE PRESENT ATTITUDE
C *** EXPRESSED BY THE DIRECTION COSINE MATRIX OF THE BODY AXIS
C *** SYSTEM (D(I,J))
C *** IT ALSO CALCULATES APPROXIMATE VALUES OF THE EULER ANGLE RATES
C *** AND THE BODY AXIS VELOCITIES

C
COMMON/AEROUT/TIDLEX ,TMILX ,TABX
1 ,FLODMX ,FLODSX
2 ,FMMINX ,FMMAXX ,RECANX
3 ,ALPHAX
4 ,CDX ,SBCDX
COMMON/ATTTAR/ANGETA,ANGETT,ANGXIA,ANGXIT,CRGR(3,3),DLOSSA,
1 DLOSST,DRGR(3,3),FAZMSA,FAZMST,FELVSA,FELVST,FLOSSA,
2 FLOSST,PSUBSA,PSUBST,RANGE,RRATE,XAINT,XTINA,YAINT,
3 YTINA,ZAINT,ZTINA
COMMON/CELSTT/AN1T,AN2T,AN3T,DTPRT,GLEVLT,ICMNWT,ICOMDT,ICRECT,
1 ISTRGT,JVALUT,LCELLT(40),LVALT(40),LWEGHT(40),
2 NCELLT,NCLSTT,NOWPRT,NTILTT,NTRYT,ROTST,ROTT,TIMEPT
COMMON/ERNION/ AB(4,2)
COMMON/ K4/DT,TBEGN,TIME,PI,PIDV2,PIDV4,TWOPI,DEGRD,RADDG,G
1 ,VAR(20),IVAR(20),TEND
COMMON/HASSLE/IABLEA ,IABLET
COMMON/VARBLT/ACCXET,ACCYET,ACCZET,ALFAT,CBART(3,3),CBT(3,3),CDT,
1 CLALET,CLT,COPHIT,COPSIT,COTHHT ,CST,D(3,3),DRAGT,
2 FFLOST,FLIFTT,FLODMT,FMACHT,FMMAXT,FMMINT,INIZT,
3 PHIBRT,PHIT,PMAXT,PPDTT,PT,PSIBRT,PSIT,QMAXT,QPDTT,
4 QT,RECAN, RHOT,RMAXT,RPDTT,RT,SIPHIT,SIPSIT,SITHTT ,
5 SPECET,ST,TABT,THETAT,THETBT,TIDLET,TMILT,TPOST,
6 TRSTT,UT,VELT,VHORT,VT,WEITT,WT,XEDOTT,XET,YEDOTT,
7 YET,ZEDOTT,ZET,FLODST
COMMON/ENERGY/SPENYA,SPENYT
DIMENSION CDES(3,3)
IF(INIZT)8,8,1

C
C *** INITIALIZATION AT BEGIN OF A RUN
C-----
C

1 CONTINUE
PMAXT=90./DEGRD
QMAXT = 25.0/DEGRD
RMAXT=15./DEGRD
PPDTT=PMAXT*DT
QPDTT=QMAXT*DT
RPDTT=RMAXT*DT
CALL FILTRT (1,1.,1.,1)
NROLL=0
ICOMO=ICOMDT
ISTRO=ISTRGT

```

PHIO=PHIT
PHINST= PHIT
THETO=THETAT
PSIO=PSIT
VHORT=SQRT (XEDOTT**2+YEDOTT**2)
VELT=SQRT (VHORT**2+ZEDOTT**2)
CALL CMTRX (PSIT, THETAT, PHIT, D)
CALL NORPLN (XEDOTT, YEDOTT, ZEDOTT, ROTT, AN1T, AN2T, AN3T)

```

```

C
HT=-ZET
CALL CSRHO (HT, CS, RHO)
RHOT=RHO
CST=CS
FMACHT=VELT/CST
IABLET= TPOST
CALL AERF4 ( HT, FMACHT, -3)
IF (ISTRGT.EQ.1) GLEVL=(VHORT/VELT)/FLODMX
IF (ISTRGT.EQ.-1) GLEVL=- (VHORT/VELT)/FLODMX
GLEVO = GLEVL
RETURN

```

```

C
C *** AT 8 IS NORMAL ENTRY DURING OPERATION
C -----
C

```

```

8 CONTINUE
HT=-ZET
CALL CSRHO (HT, CS, RHO)
RHOT=RHO
CST=CS
FMASST=WEITT/G
VHORS=XEDOTT**2+YEDOTT**2
VHORT=SQRT (VHORS)
VSQ =VHORS+ZEDOTT**2
QBARS=VSQ*RHOT*ST/2.
VELT=SQRT (VSQ )
SPECET= HT+VSQ / (2.*G)
SPENYT= SPECET
FMACHT=VELT/CST
PSIBRT=ATAN2 (YEDOTT, XEDOTT)
THETBT=ATAN (-ZEDOTT/VHORT)
SIPSIT=SIN (PSIBRT)
COPSIT=COS (PSIBRT)
SITHTT =SIN (THETBT)
COTHTT =COS (THETBT)
PHIT = PHINST
ALT=-ZET
IABLET= TPOST
1220 CALL AERF4 (ALT, FMACHT, -3)
RECANT=RECANX
FMMINT=FMMINX
FMMAXT=FMMAXX
TMILT=TMILX
TIDLET=TIDLEX
TABT=TABX

```

```

      FLODMT=FLODMX
      FLODST=FLODSX
800  CONTINUE
      MAXCL = 0
      MAXPHI= 0
802  CONTINUE
      IF(MAXPHI.LE.2)GO TO 804
      CALL ERMSG(14,SIPHS)
C
999  TIME=TEND+2.*DT
      RETURN
804  CONTINUE
C
C      OBTAIN AERO DATA
      FLOADT=GLEVLТ*FLODMT
      FLIFTT=WEITТ*FLOADT
      CLT=FLIFTT/QBARS
      CLTO = CLT
      CALL AERF4(FMACHТ,CLТ,-4)
      IF(CLT.EQ.CLTO)GO TO 805
      IF(MAXCL.EQ.0)GO TO 803
      CALL ERMSG(8,CLТ)
      GO TO 999
803  CONTINUE
      MAXCL = 1
      GLEVLТ= GLEVLТ/CLTO*CLТ
      FLOADT= GLEVLТ*FLODMT
      FLIFTT= WEITТ*FLOADT
805  CONTINUE
      ALFAT = ALPHAX
      SIALF=SIN(ALFAT)
      CSALF=COS(ALFAT)
      CDT=CDX
      DRAGT=QBARS*CDT
C *** OBTAIN THRUST   TPOS=0   IDLE
C                               TPOS=1   MILITARY
C                               TPOS=2   AFTERBURNER
C *** THRUST TABLE OF F-4 IS FOR ONE ENGINE ONLY
C
      IF(TPOST.EQ.0.) TRSTT=TIDLET*2.
      IF(TPOST.EQ.1.) TRSTT=TMILT*2.
      IF(TPOST.EQ.2.) TRSTT=TABT*2.
      GLTFL = FLOADT+TRSTT/WEITТ*SIALF
      IF(GLTFL.EQ.0.0)GLTFL= 1.0E-06
      IF(ICMNWT)30,30,810
810  CONTINUE
C
C *** THIS PORTION WHEN THE COMMAND HAS CHANGED,
C *** CALCULATE DESIRED CHANGE IN ROLL ANGLE
C *** AND HOW MANY DT WE ARE GOING TO ROLL WITH MAX ROLL RATE
C
      IF(PHIT.GE.-PI)GO TO 812
      PHIT = PHIT+TWOPI
      PHIO = PHIO+TWOPI

```

```

      GO TO 814
812 CONTINUE
      IF(PHIT.LE.PI)GO TO 814
      PHIT = PHIT-TWOPI
      PHIO = PHIO-TWOPI
814 CONTINUE
      IF(ISTRGT) 830,850,820
820 DPHI=-PHIT
      GO TO 8700
830 DPHI=PI-PHIT
      GO TO 8700
850 CONTINUE
      CALL CMTRX(PSIBRT,THETBT,ROTT,CDES)
      SIPHS =-CDES(2,3)/GLTFL
      IF(ABS(SIPHS).LE.1.0E-06)SIPHS= 0.0
      SIPHIT= SIN(ROTT)
      IF(ABS(SIPHIT).LE.1.0E-06)SIPHIT= 0.0
      IF(ABS(SIPHS).GT.ABS(SIPHIT))GO TO 225
      PHIS=ASIN(SIPHS)
      PHIDS=ROTT+PHIS
      DPHI=PHIDS-PHIT
8700 CONTINUE
      ADPHI=ABS(DPHI)
      IF(ABS(PI-ADPHI).LE.1.0E-04)GO TO 860
      IF(PI-ADPHI) 855,860,870
870 SIGNR=1.
      IF(DPHI.LT.0.) SIGNR=-1.
      GO TO 880
855 SIGNR=-1.
      IF(DPHI.LT.0.) SIGNR=1.
      DPHI=TWOPI-ABS(DPHI)
      GO TO 880
860 DPHI=PI
      ADPHI= DPHI
      IF(YAINT.EQ.0.0)GO TO 8601
      SIGNR = YAINT/ABS(YAINT)
      GO TO 8602
8601 SIGNR =-1.0
8602 CONTINUE
880 NROLL=(ABS(DPHI)/PPD TT)
C
      ICMNWT=0
C *** CHECK IF LAST TRANSITION IS COMPLETED
      30 CONTINUE
C
      IF(NROLL) 31,32,500
      31 CALL ERMSG(1,DUM)
      NROLL = 0
      32 CONTINUE
C
C *** WE ARE IN THE MANEUVER MODE
C -----
C
      PHIBRT=ATAN2((AN3T/COTH TT ),(AN2T*COPSIT-AN1T*SIPSIT))

```

```

SIPHIT=SIN (PHIBRT)
COPHIT=COS (PHIBRT)
IF (ABS (SIPHIT) .LE.1.0E-06)SIPHIT= 0.0
CALL CMTRX (PSIBRT, THETBT, PHIBRT, CBT)
C
C CHECK FOR STRAIGHT FLIGHT
C
IF (ISTRO) 210, 220, 230
230 PHIT=0.
GO TO 255
210 PHIT=+PI
GO TO 255
220 CONTINUE
SIPHS =-CBT (2, 3) /GLTFL
IF (ABS (SIPHS) .LE.1.0E-06)SIPHS= 0.0
IF (ABS (SIPHS) .LE.ABS (SIPHIT))GO TO 240
225 CONTINUE
MAXPHI= MAXPHI+1
IF (MAXCL.NE.0)GO TO 235
C INCREASE G-LEVEL
GLEVLT= GLEVLT/ABS (SIPHIT) * (ABS (SIPHS) +1.0E-06)
IF (ABS (GLEVLT) .LE.1.0)GO TO 802
GLEVLT= 1.0
235 CONTINUE
C VARY MANEUVER PLANE
IF (ABS (PHIBRT) -PIDV2) 237, 237, 238
237 CONTINUE
PHIBRT= 0.0
GO TO 239
238 CONTINUE
PHIBRT= PI
239 CONTINUE
ROTT = PHIBRT
CALL NORPLN (XEDOTT, YEDOTT, ZEDOTT, PHIBRT, AN1T, AN2T, AN3T)
GO TO 802
240 CONTINUE
PHIS=ASIN (SIPHS)
250 PHIT=PHIBRT+PHIS
255 CONTINUE
C
GO TO 1000
C
C *** 500 FOR CASE WHEN WE ARE IN TRANSITION MODE
C -----
C
500 CONTINUE
C
FLIFTT=FLIFO
ALFAT=ALFO
DRAGT=DRAGO
TRSTT=TRSTO
CSALF=CSALO
SIALF=SIALO
C

```

```

PHIT=PHIT+SIGNR*PPD TT
PHIBRT= PHIT
C
NROLL=NROLL-1
IF(NROLL) 31,910,1000
910 CONTINUE
C
C *** 910 FOR CALCULATION OF NEW MANEUVER PLANE NORMAL ( PLANE THROUGH
C *** PRESENT VEL. VECTOR WITH PREVIOUSLY DETERMINED ANGLE ROTT)
C
CALL NORPLN(XEDOTT,YEDOTT,ZEDOTT,ROTT,AN1T,AN2T,AN3T)
C
C *** 1000 COMBINED PATH FOR ALL MANEUVER MODES
C -----
C
1000 CONTINUE
C
C
C OBTAIN DIRECTION COSINE MATRIX OF INSTANTANEOUS MANEUVER PLANE SYS
C
CALL CMTRX(PSIBRT,THETBT,PHIT,CBART)
C
C CALCULATE FORWARD FORCE (ALONG MANEUVER PLANE X-AXIS)
C AND NORMAL FFORCE (POSITIVE ALONG THE POSITIVE ZM AXIS)
C INCLUDE AERODYNAMIC FORCES AND THRUST, BUT NOT WEIGHT
C
FF=TRSTT*CSALF-DRAGT
FN=-FLIFTT-TRSTT*SIALF
C
C TRANSFORM FORCES INTO INERTIAL SYSTEM AND ADD WEIGHT
C
FXET=FF*CBART(1,1)+FN*CBART(3,1)
FYET=FF*CBART(1,2)+FN*CBART(3,2)
FZET=FF*CBART(1,3)+FN*CBART(3,3) + WEITT
C
ACCXET=FXET/FMASST
ACCYET=FYET/FMASST
ACCZET=FZET/FMASST
PSUBST= FE/WEITT*VELT
C
OBTAIN PROJECTIONS OF X-BODY AXIS ONTO THE THREE INERTIAL AXES
XBX=CSALF*CBART(1,1)-SIALF*CBART(3,1)
XBY=CSALF*CBART(1,2)-SIALF*CBART(3,2)
XBX=CSALF*CBART(1,3)-SIALF*CBART(3,3)
C
C NOW WE CAN GET THE CURRENT EULER ANGLES OF THE BODY AXIS SYSTEM
C
PSIT=ATAN2(XBY,XBX)
THETAT=ATAN(-XBX/SQRT(XBX**2+XBY**2))
C
C *** OBTAIN DIRECTION COSINE MATRIX OF BODY AXIS SYSTEM
C
PHINST= PHIT
CALL NORPLN(XEDOTT,YEDOTT,ZEDOTT,PHIT,AN1,AN2,AN3)
PHIT = ATAN2(AN3/COS(THETAT),AN2*COS(PSIT)-AN1*SIN(PSIT))

```



```

CALL CMTRX(PSIT,THETAT,PHIT,CDES)
C
C
C
OBTAIN APPROXIMATE EULER ANGLE RATES

DELPHI=PHIT-PHIO
IF(DELPHI.LT.(-PI)) DELPHI=TWOPI+DELPHI
IF(DELPHI.GT.PI) DELPHI=-(TWOPI-DELPHI)
DELTHT =THETAT-THETO
IF(DELTHT .LT.(-PI)) DELTHT =TWOPI+DELTHT
IF(DELTHT .GT.PI) DELTHT =-(TWOPI-DELTHT )
DELPSI=PSIT-PSIO
IF(DELPSI.LT.(-PI)) DELPSI=TWOPI+DELPSI
IF(DELPSI.GT.PI) DELPSI=-(TWOPI-DELPSI)
PHIDOT=DELPHI/DT
THETDT =DELTHT /DT
PSIDOT=DELPSI/DT
COTHEO= THETO+DELTHT *0.5
PT = PHIDOT-PSIDOT*SIN(COTHEO)
COTHEO= COS(COTHEO)
COPHIO= PHIO+DELPHI*0.5
SIPHIO= SIN(COPHIO)
COPHIO= COS(COPHIO)
QT = THETDT *COPHIO+PSIDOT*SIPHIO*COTHEO
RT = PSIDOT*COPHIO*COTHEO-THETDT *SIPHIO
C
C
C *** FILTER THE BODY RATES WHICH GO TO THE DMS
C
C
CALL FILTRT (1,PT,PT,0)
CALL FILTRT (2,QT,QT,0)
CALL FILTRT (3,RT,RT,0)
IF(ABS(PT).GT.PMAXT+1.0E-06)CALL ERMSG(21,PT)
IF(ABS(QT).GT.QMAXT+1.0E-06)CALL ERMSG(22,QT)
IF(ABS(RT).GT.RMAXT+1.0E-06)CALL ERMSG(23,RT)
CALL QUATT(PT,QT,RT,AB(1,2))
CALL QUATEX(AB(1,2),D(1,1))
CALL OILER(D(1,1),PSIO,THETO,PHIO)
ANGAX = 1.0
DO 1410 I=1,3
DUMMY = 0.0
DO 1405 J=1,3
DUMMY = DUMMY+CDES(I,J)*D(I,J)
1405 CONTINUE
ANGAX = AMIN1(ANGAX,DUMMY)
1410 CONTINUE
C
C 925 IF(ANGAX.LT.0.988)PRINT 925,PSIT,PSIO,THETAT,THETO,PHIT,PHIO,TIME
C 925 FORMAT(24H T-ATTITUDE DISCREPANCY ,3(4X,2F8.4),10H AT TIME ,F8.3
C
C
C
OBTAIN BODY AXES VELOCITIES

UT=XEDOTT*D(1,1)+YEDOTT*D(1,2)+ZEDOTT*D(1,3)
VT=XEDOTT*D(2,1)+YEDOTT*D(2,2)+ZEDOTT*D(2,3)
WT=XEDOTT*D(3,1)+YEDOTT*D(3,2)+ZEDOTT*D(3,3)
C
C
SAVE PRESENT VALUES AS OLD VALUES

```

C
ICOMO=ICOMDT
ISTRO=ISTRGT
GLEVO=GLEVLT
FLIFO=FLIFTT
ALFO=ALFAT
TRSTO=TRSTT
DRAGO=DRAGT
CSALO=CSALF
SIALO=SIALF

C
RETURN
END

SUBROUTINE EXTRT(C)
C-----

C
C
C
C
C
EXTRAPOLATION OF ATTACKERS POSITION, VELOCITY AND ATTITUDE BY THE
TARGET

COMMON/EXTPOT/TIMEXT, DELTT, XZEROA, YZEROA, ZZEROA, XMIN1A, YMIN1A,
1 ZMIN1A, XMIN2A, YMIN2A, ZMIN2A, XEXPA, YEXPA, ZEXPA,
2 XDEXPA, YDEXPA, ZDEXPA, VELEXA, DRCEXA(3,3)
COMMON/ K4/DT, TBEGN, TIME, PI, PIDV2, PIDV4, TWOPI, DEGRD, RADDG, G
1 , VAR(20), IVAR(20), TEND

DIMENSION C(3,3)

C
C
C
CHECK WHETHER THE THREE POINTS USED FOR EXTRAPOLATION LIE IN
A STRAIGHT LINE

U1 = XMIN2A - XZEROA
U2 = YMIN2A - YZEROA
V1 = XMIN1A - XZEROA
V2 = YMIN1A - YZEROA
U3 = ZMIN2A - ZZEROA
V3 = ZMIN1A - ZZEROA
UXV1 = U2 * V3 - U3 * V2
UXV2 = -U1 * V3 + U3 * V1
UXV3 = U1 * V2 - U2 * V1
ILINEA = 0

UXVMG = UXV1 * UXV1 + UXV2 * UXV2 + UXV3 * UXV3

C
IF U CROSS V EQUALS ZERO, THEN THE THREE POINTS ARE COLINEAR
IF(UXVMG - .001) 10,15,15

10 ILINEA = 1

GO TO 20

15 ANORM = 1./ SQRT(UXVMG)

C
DRC21 = UXV1 * ANORM

DRC22 = UXV2 * ANORM

DRC23 = UXV3 * ANORM

20 CONTINUE

DTSQ=DELTT*DELTT

DXSQ=TIMEXT*TIMEXT

AX = (.5 * (XZEROA + XMIN2A) - XMIN1A)/DTSQ

```

AY = (.5 * ( YZEROA + YMIN2A ) - YMIN1A )/DTSQ
AZ = (.5 * ( ZZEROA + ZMIN2A ) - ZMIN1A )/DTSQ
BX = (3.*XZEROA-4.*XMIN1A+XMIN2A)/(2.*DELTT)
BY = (3.*YZEROA-4.*YMIN1A+YMIN2A)/(2.*DELTT)
BZ = (3.*ZZEROA-4.*ZMIN1A+ZMIN2A)/(2.*DELTT)
XEXPA=AX*DXSQ+BX*TIMEXT+XZEROA
YEXPA=AY*DXSQ+BY*TIMEXT+YZEROA
ZEXPA=AZ*DXSQ+BZ*TIMEXT+ZZEROA
XDEX=2.*AX*TIMEXT+BX
YDEX=2.*AY*TIMEXT+BY
ZDEX=2.*AZ*TIMEXT+BZ
XDEXPA=XDEX
YDEXPA=YDEX
ZDEXPA=ZDEX
VELPA = SQRT(XDEX * XDEX + YDEX * YDEX + ZDEX * ZDEX )
VELEXA=VELPA

```

C
C
C

OBTAIN DIRECTION COSINES

```

IF(ILINEA ) 38,38,39
38 FACT = 1. / VELPA
DRC11 = XDEX * FACT
DRC12 = YDEX * FACT
DRC13 = ZDEX * FACT
DRCEXA(1,1) =DRC11
DRCEXA(1,2) =DRC12
DRCEXA(1,3) =DRC13
DRCEXA(2,1) =DRC21
DRCEXA(2,2) =DRC22
DRCEXA(2,3) =DRC23
DRCEXA(3,1) = DRC12 * DRC23 - DRC13 * DRC22
DRCEXA(3,2) = DRC13 * DRC21 - DRC11 * DRC23
DRCEXA(3,3) = DRC11 * DRC22 - DRC12 * DRC21
GO TO 40

```

C FOR COLINEAR CASE EXTRAPOLATED DIRECTION COSINES ARE EQUAL
C TO THE PRESENT ONES

```

39 DO 3901 I = 1,3
DO 3901 J = 1,3
3901 DRCEXA(I,J) =C(I,J)
40 CONTINUE
RETURN
END

```

SUBROUTINE FILTRT (NPQR,RATED,RATEAC,INIZ)

C-----
C

```

IF(INIZ.EQ.0) GO TO 100
DEGRD=180./(4.*ATAN(1.))
PMAx=90./DEGRD
QMAX = 25.0/DEGRD
RMAX=15./DEGRD
SPP=0.
SPQ=0.
SPR=0.

```

```

      FF=0.6
      RETURN
100 GO TO (110,120,130) NPQR
110 PP=SPP
      PMX=PMAX
      GO TO 200
120 PP=SPQ
      PMX=QMAX
      GO TO 200
130 PP=SPR
      PMX=RMAX
200 CONTINUE
      IF(RATED-PMX) 210,213,211
211 RATEAC=AMIN1(PP+FF*PMX,PMX)
      GO TO 215
210 IF(RATED+PMX) 212,213,213
212 RATEAC=AMAX1(PP-FF*PMX,-PMX)
      GO TO 215
213 RATEAC=PP+FF*(RATED-PP)
215 GO TO(2210,2220,2230) NPQR
2210 SPP=RATEAC
      GO TO 3000
2220 SPQ=RATEAC
      GO TO 3000
2230 SPR=RATEAC
      GO TO 3000
3000 RETURN
      END

```

SUBROUTINE THRTL

C-----
C

```

COMMON/ATTTAR/ANGETA,ANGETT,ANGXIA,ANGXIT,CRGR(3,3),DLOSSA,
1      DLOSST,DRGR(3,3),FAZMSA,FAZMST,FELVSA,FELVST,FLOSSA,
2      FLOSST,PSUBSA,PSUBST,RANGE,RRATE,XAINT,XTINA,YAINT,
3      YTINA,ZAINT,ZTINA
COMMON/      K4/DT,TBEGN,TIME,PI,PIDV2,PIDV4,TWOPI,DEGRD,RADDG,G
1      ,VAR(20),IVAR(20),TEND
COMMON/VARBLT/ACCXET,ACCYET,ACCZET,ALFAT,CBART(3,3),CBT(3,3),CDT,
1      CLALFT,CLT,COPHIT,COPSIT,COTHTT,CST,D(3,3),DRAGT,
2      FFLOST,FLIETT,FLODMT,FMACHT,FMMAXT,FMMINT,INIZT,
3      PHIBRT,PHIT,PMAXT,PPDPT,PT,PSIBRT,PSIT,QMAXT,QPDPT,
4      QT,RECANT,RHOT,RMAXT,RPDPT,RT,SIPHIT,SIPSIT,SITHTT,
5      SPECET,ST,TABT,THETAT,THETBT,TIDLET,TMILT,TPOST,
6      TRSTT,UT,VELT,VHORT,VT,WEITT,WT,XEDOTT,XET,YEDOTT,
7      YET,ZEDOTT,ZET,FLODST
COMMON/VARBLA/ACCXEA,ACCYEA,ACCZEA,ALFAA,CBARA(3,3),CBA(3,3),CDA,
1      CLALFA,CLA,COPHIA,COPSIA,COTHTA,CSA,C(3,3),DRAGA,
2      FFLOSA,FLIFTA,FLODMA,FMACHA,FMMAXA,FMMINA,INIZA,
3      PHIBRA,PHIA,PMAXA,PPDTA,PA,PSIBRA,PSIA,QMAXA,QPDTA,
4      QA,RECANA,RHOA,RMAXA,RPDTA,RA,SIPHIA,SIPSIA,SITHTA,
5      SPECEA,SA,TABA,THETAA,THETBA,TIDLEA,TMILA,TPOSA,
6      TRSTA,UA,VELA,VHORA,VA,WEITA,WA,XEDOTA,XEA,YEDOTA,
7      YEA,ZEDOTA,ZEA,FLODSA

```

COMMON/TALLY/ OPTIMA ,OFTIMT ,GUNTMA ,GUNTMT ,ANGOFA,ANGOFT,
1 DEVA,DEVAT

C

TEN = 10./DEGRD
SIXTY = 60./DEGRD
IF (FMACHT.GT.(FMACHA+0.05)) GO TO 7000
IF(XAINT.LT.0.0)GO TO 4000
IF (ABS (ANGOFT) .GE.SIXTY)RETURN
IF (ABS (DEVAT) .GE.SIXTY)RETURN
IF (RANGE.GE.5000.0)RETURN
IF (RANGE.LT.3000.0)GO TO 7000
IF (RRATE.GE.-300.0)RETURN
IF (THETBT.LT.SIXTY)GO TO 7000
RETURN
4000 CONTINUE
IF (ABS (FLOSSA) .LE.TEN)RETURN
IF (ABS (FLOSSA) .GE.SIXTY)RETURN
IF (RANGE.GE.1500.0)RETURN
IF (RRATE.GE.-300.0)RETURN
7000 CONTINUE
TPOST = 0.0
RETURN
END

SUBROUTINE TRYNXT

C-----

C

COMMON/CELSTT/AN1T,AN2T,AN3T,DTPRT,GLEVLT,ICMNWT,ICOMDT,ICRECT,
1 ISTRGT,JVALUT,LCELLT(40),LVALT(40),LWEGHT(40),
2 NCELLT,NCLSTT,NOWPRT,NTILTT,NTRYT,ROTST,ROTT,TIMEPT
COMMON/ K4/DT,TBEGN,TIME,PI,PIDV2,PIDV4,TWOPI,DEGRD,RADDG,G
1 ,VAR(20),IVAR(20),TEND
COMMON/PREDIC/CNEW(3,3),VXNEW,VYNEW,VZNEW,XXNEW,YYNEW,ZZNEW
COMMON/TRIALT/AN1TRT(10),AN2TRT(10),AN3TRT(10),CPRT(3,3,10),
1 DLOSRT(10),DRATT(10),DRGPRT(10),DXEPRT(10),
2 DYEPRT(10),DZEPRT(10),FLDTRT(10),FLOSRT(10),
3 ICNPRT(10),ICTRYT(10),ISTPRT(20,10),ISTRYT(10),
4 IVALTT,IVPRT(10),KNEWT,KROTST,PSTRT(10),PXEPRT(10),
5 PYEPRT(10),PZEPRT(10),RNGPRT(10),ROTNCT,ROTN2T,
6 ROTRYT(10),THTRT(10),TPOTRT(10),XEXA,YEXA,ZEXA
COMMON/VARBLT/ACCXET,ACCYET,ACCZET,ALFAT,CBART(3,3),CBT(3,3),CDT,
1 CLALFT,CLT,COPHIT,COPSIT,COTHTT,CST,D(3,3),DRAGT,
2 FFLOST,FLIFTT,FLODMT,FMACHT,FMMAXT,FMMINT,INIZT,
3 PHIBRT,PHIT,PMAXT,PPDPT,PT,PSIBRT,PSIT,QMAXT,QPDPT,
4 QT,RECANT,RHOT,RMAXT,RPDPT,RT,SIPHIT,SIPSIT,SITHTT
5 SPECET,ST,TABT,THETAT,THETBT,TIDLET,TMILT,TPOST,
6 TRSTT,UT,VELT,VHORT,VT,WEITT,WT,XEDOTT,XET,YEDOTT,
7 YET,ZEDOTT,ZET,FLODST
DIMENSION CMPL(3,3)
DATA EXLSM/1.57080/
DATA EIGHTY/1.39626/
DATA TAB155/1.0/,TAB156/1.1/,TAB157/0.1/

C

TRFACT=2.

```

FLODES= VELT*VELT*RHOT*ST*0.5
FLODES= CLADES(2,FMACHT)/FLODMT*FLODES/WEITT
FLODES= AMIN1(0.95,FLODES)
FLOSTR=(VHORT/VELT)/FLODMT
DO 7 I=1,10
  ISTRYT(I)= 0
  ICTRYT(I)= 0
  ROTRYT(I)= 0.0
  FLDTRT(I)= TAB155
  TPOTRT(I)= 2.0
7 CONTINUE
C CALCULATE ROTST, THE ANGLE OF ROTATION OF THE MANEUVER PLANE
C THROUGH PRESENT VELOCITY VECTOR AND EXTRAPOLATED OPPONENTS POS.
TAXE = XEXA- XET
TAYE = YEXA- YET
TAZE = ZEXA- ZET
DZ=(XEDOTT*ZEDOTT*TAXE+YEDOTT*ZEDOTT*TAYE-(VHORT**2)*TAZE)/VELT
DY=-YEDOTT*TAXE+XEDOTT*TAYE
IF(DY.EQ.0.) GO TO 10
ROTST=ATAN2(DY,DZ)
GO TO 99
10 IF(DZ.GE.0.) ROTST=0.
  IF(DZ.LT.0.) ROTST=PI
99 CONTINUE
DIVEAN =-THETBT
IF(DIVEAN .LT.0.) GO TO 1002
IF(DIVEAN .GE.RECANT) GO TO 500
IF(DIVEAN .LT.(0.8*RECANT)) GO TO 1002
C
C *** DIVE ANGLE IS BETWEEN 80 AND 100 PERCENT OF RECOVERY ANGLE
C
IF(RECANT.GT.1.2)GO TO 1002
NTRYT=3
KROTST=1
C
C *** FIRST TRIAL IS PULLUP IN VERTICAL PLANE
C
C *** TRIAL 2 AND 3 ARE PULLUPS IN ADJACENT PLANES
C
ICTRYT(2)=1
ROTRYT(2)=ROTNCT
C
ICTRYT(3)=-1
ROTRYT(3)=-ROTNCT
GO TO 600
500 CONTINUE
C
C *** ONLY ONE MANEUVER ALLOWED.. STRAIGHT PULLUP
C
NTRYT=1
KROTST=1
GO TO 600
C
C CHECK FOR LOW LOAD FACTOR

```

C

```
1002 CONTINUE
      IF(FLODMT.GT.1.5)GO TO 1001
      IF(THETBT.LE.0.0)GO TO 3500
      IF(THETBT.GT.EIGHTY)GO TO 3450
      IF(FLODMT.GE.1.0)GO TO 3600
3330 CONTINUE
      NTRYT = 2
      IF(ABS(ROTST).LE.PIDV2)KROTST= 1
      IF(ABS(ROTST).GT.PIDV2)KROTST= 2
      ROTRYT(2) = PI
      ICTRYT(2) = PI/ROTNCT+0.001
      FLDTRT(1)=0.001
      FLDTRT(2)=0.001
      GO TO 600
3450 CONTINUE
      NTRYT = 1
      CALL GETRXN(ROTST,ROTNCT,ROTRYT(1),ICTRYT(1))
      GO TO 600
3500 CONTINUE
3600 CONTINUE
      NTRYT = 6
      DO 3605 I=1,NTRYT
      FLDTRT(I) = FLODES
3605 CONTINUE
      KROTST= 3
      IF(ABS(ROTST).GT.PIDV2)KROTST= 5
      IF(ROTST.LT.0.0)KROTST= KROTST+1
      ROTRYT(2) = PI
      ROTRYT(3) = ROTNCT
      ROTRYT(4) = -ROTNCT
      ROTRYT(5) = PI-ROTNCT
      ROTRYT(6) = ROTNCT-PI
      ICTRYT(2) = PI/ROTNCT
      ICTRYT(3) = 1
      ICTRYT(4) = -1
      ICTRYT(5) = ICTRYT(2)-1
      ICTRYT(6) = 1-ICTRYT(2)
      GO TO 600
```

C

C

```
C *** 1001 FOR ' NORMAL ' CONDITIONS
C *****
```

C

```
1001 CONTINUE
      NTRYT = 4
      KROTST = 1
      NTRY = 4
      ISTRYT(1) = ISTRGT
      ROTRYT(1) = PHIBRT
      IF(ISTRGT) 2010,100,200
100 CONTINUE
      FLDTRT(1) = TAB155+(GLEVLGT-TAB155)*RADIFY(6HGLEVLGT,0.75,0.5,1.0)
      ICTRYT(1) = PHIBRT/ROTNCT
```

```

ROTPM=PHIBRT
CALL GETRXN(ROTPM,ROTNCT,ROTR2,ICTR2)
ROTRYT(2) = ROTR2
ICTRYT(2) = ICTR2
IF(ROTR2-ROTPM)130,123,125
123 CONTINUE
IF(ROTPM)130,130,125
125 ROTRYT(3) = ROTR2 - ROTNCT
ICTRYT(3) = ICTR2 - 1
GO TO 140
130 ROTRYT(3) = ROTR2 + ROTNCT
ICTRYT(3) = ICTR2 +1
140 CONTINUE
ISTRYT(4)= 1
IF( ABS( ROTST ) .LE. PIDV2) GO TO 150
ISTRYT( 4 ) = -1
ROTRYT(4) = PI
150 CONTINUE
C IF NONE OF THE THREE MANEUVER PLANES IN TRIALS 1, 2, OR 3 IS THE
C PLANE CLOSEST TO THE OPPONENT, ADD A FIFTH TRIAL MANEUVER
K = 1
TESROT= PI
DO 250 I=1,3
TEROT = ABS(ROTRYT(I)-ROTST)
IF(TEROT.GT.PI)TEROT= TWOPI-TEROT
IF(TEROT.GT.TESROT)GO TO 250
K = I
TESROT= TEROT
250 CONTINUE
IF(TESROT.LE.ROTN2T)GO TO 275
C NOT INCLUDED
NTRYT = 5
NTRY = 5
KROTST= 5
CALL GETRXN(ROTST,ROTNCT,ROTRYT(5),ICTRYT(5))
GO TO 2000
C INCLUDED
275 CONTINUE
KROTST= K
IF(K.EQ.1.AND.ABS(FLODES-FLDTRT(1)).LE.0.05)GO TO 2000
NTRYT = 5
ICTRYT(5) = ICTRYT(K)
ROTRYT(5) = ROTRYT(K)
FLDTRT(5) = FLODES
GO TO 2000
200 CONTINUE
FLDTRT(1) = FLOSTR
ISTRYT(2) = -1
ROTRYT(2) = PI
GO TO 2020
2010 CONTINUE
FLDTRT(1) = -FLOSTR
ISTRYT(2) = 1
2020 KROTST = 3

```



```

        CALL GETRXN(ROTST,ROTNCT,ROTR3,ICTR3)
        ROTRYT(3) = ROTR3
        ICTRYT(3) = ICTR3
        IF( ROTR3 - ROTST ) 2110,2105,2105
2105 ROTRYT(4) = ROTR3 - ROTNCT
        ICTRYT(4) = ICTR3 - 1
        GO TO 2000
2110 ROTRYT(4) = ROTR3 + ROTNCT
        ICTRYT(4) = ICTR3 + 1
2000 CONTINUE
        IF ( FFLOST .GT. EXLSM ) GO TO 2400
C *** CALCULATE REQUIRED G-LEVEL FOR TURN INTO OPPONENTS EXTRAPOLATED
C *** POSITION
        CALL CMTRX(PSIBRT,THETBT,ROTST,CMPL)
        DIST2=TAXE**2+TAYE**2+TAZE**2
        ZMT=TAXE*CMPL(3,1)+TAYE*CMPL(3,2)+TAZE*CMPL(3,3)
        RADIS=DIST2/(2.*ZMT)
        GL2=(ABS((VELT**2)/RADIS)/G) + CMPL(3,3)
        GL3=ABS(CMPL(2,3))
        GLEVRT=SQRT(GL2**2+GL3**2)/FLODMT
        GLEVRT= GLEVRT*TAB156+TAB157
        IF ( GLEVRT .GT. 1.0 ) GO TO 2400
        NTRYT = NTRYT + 1
        FLDTRT( NTRYT ) = GLEVRT
        CALL GETRXN(ROTST,ROTNCT,ROTRYT(NTRYT),ICTRYT(NTRYT))
        KROTST= NTRYT
        GO TO 2500
2400 CONTINUE
        IF(JVALUT.GT.6)GO TO 2500
C DEFENSIVE MANEUVER
        NTRYT = NTRYT+1
        IF(JVALUT.LE.5)KROTST= NTRYT
        ROTPM = PIDV2-ROTNCT
        IF(ABS(ROTST).GT.PIDV2)ROTPM=-ROTPM
        ROTPM = ABS(ROTST)+ROTPM
        IF(ROTST.LT.0.0)ROTPM=-ROTPM
        CALL GETRXN(ROTPM,ROTNCT,ROTRYT(NTRYT),ICTRYT(NTRYT))
        ISTRYT(NTRYT)= 0
        FLDTRT(NTRYT)= FLODES
2500 CONTINUE
        DO 2555 I=2,NTRY
        IF(ISTRYT(I))2551,2555,2553
2551 FLDTRT(I)=-FLOSTR
        GO TO 2555
2553 FLDTRT(I)= FLOSTR
2555 CONTINUE
        DO 2560 I=1,NTRYT
2560 TPOTRT(I)=TTRACT
        600 CONTINUE
        DO 2600 I=1,NTRYT
        GLEVRT= FLDTRT(I)*FLODMT
        GLEV = GLEVRT
        CALL HUBLO(1,2,DTPRT,GLEVRT)
        IF(GLEV.NE.GLEVRT)FLDTRT(I)= GLEVRT/FLODMT

```

```

ROSS = ROTRYT(I)
IF(ISTRYT(I).NE.0)GO TO 2580
SROSS = SIN(ROSS)
REDUC = ABS(SROSS/FLODMT*COTHTT /FLDTRT(I))
IF(REDUC.LE.ABS(SROSS))GO TO 2580
ROSS = 0.0
IF(ABS(ROTRYT(I)).LE.PIDV2)GO TO 2570
ROSS = PI
2570 CONTINUE
ROTRYT(I)= ROSS
2580 CONTINUE
CALL NORPLN(XEDOTT,YEDOTT,ZEDOTT,ROSS,AN1TRT(I),AN2TRT(I),
1 AN3TRT(I))
2600 CONTINUE
RETURN
END

```

SUBROUTINE QUATT(P,Q,R,UAT)

```

C-----
C
COMMON/      K4/DT,TBEGN,TIME,PI,PIDV2,PIDV4,TWOPI,DEGRD,RADDG,G
1            ,VAR(20),IVAR(20),TEND
SAVE
EQUIVALENCE (DT,H)
DIMENSION UAT(4)
DIMENSION AT(4)
REAL IK,JK,KK
EQUIVALENCE (C1,CRHOK)
DATA OM2ZER /1.E-6/

C
C QUATERNION INTEGRATION
OMEG K2= P*P+Q*Q+R*R
IF(OMEG K2 .LT.OM2ZER ) OMEG K2 =OM2ZER
OMEGAK =SQRT(OMEG K2 )
RHOK =H*OMEGAK *.5
SRHOK =SIN(RHOK )
CRHOK =COS(RHOK )
C2P =SRHOK /OMEGAK
C3P =2.*(1.-CRHOK )/OMEG K2
C4 =4.*(H-2.*C2P )/OMEG K2
IK =-C2P*R
JK = C2P*Q
AT(1) = C1*UAT(1)+IK*UAT(2)-JK*UAT(3)-KK*UAT(4)
KK = C2P*P
AT(2) =-IK*UAT(1)+C1*UAT(2)-KK*UAT(3)+JK*UAT(4)
AT(3) = JK*UAT(1)+KK*UAT(2)+C1*UAT(3)+IK*UAT(4)
AT(4) = KK*UAT(1)-JK*UAT(2)-IK*UAT(3)+C1*UAT(4)

C NORMALIZATION
ORMAL = 1.0/SQRT(AT(1)**2+AT(2)**2+AT(3)**2+AT(4)**2)
DO 75 J=1,4
UAT(J)= AT(J)*ORMAL
75 CONTINUE
RETURN
END

```

SUBROUTINE REACTT (XEA, YEA, ZEA, C)

C-----
C

```

COMMON/AEROUT/TIDLEX          , TMILX          , TABX
1          , FLODMX          , FLODSX
2          , FMMINX          , FMMAXX          , RECANX
3          , ALPHAX
4          , CDX          , SBCDX
COMMON/ATTTAR/ANGETA, ANGETT, ANGXIA, ANGXIT, CRGR (3, 3) , DLOSSA,
1          DLOSST, DRGR (3, 3) , FAZMSA, FAZMST, FELVSA, FELVST, FLOSSA,
2          FLOSST, PSUBSA, PSUBST, RANGE, RRATE, XAINT, XTINA, YAINT,
3          YTINA, ZAINT, ZTINA
COMMON/CELSTT/AN1T, AN2T, AN3T, DTPRT, GLEVL, ICMNWT, ICOMDT, ICRECT,
1          ISTRGT, JVALUT, LCELLT (40) , LVALT (40) , LWEGHT (40) ,
2          NCELLT, NCLSTT, NOWPRT, NTILT, NTRYT, ROTST, ROTT, TIMEPT
COMMON/ENERGY/SPENYA, SPENYT
COMMON/EXTPOT/TIMEXT, DELTT, XZEROA, YZEROA, ZZEROA, XMIN1A, YMIN1A,
1          ZMIN1A, XMIN2A, YMIN2A, ZMIN2A, XEXPA, YEXPA, ZEXPA,
2          XDEXPA, YDEXPA, ZDEXPA, VELEXA, DRCEXA (3, 3)
COMMON/HASSLE/IABLEA          , IABLET
COMMON/      K4/DT, TBEGN, TIME, PI, PIDV2, PIDV4, TWOPI, DEGRD, RADDG, G
1          , VAR (20) , IVAR (20) , TEND
COMMON/PREDIC/CNEW (3, 3) , VXNEW, VYNEW, VZNEW, XXNEW, YYNEW, ZZNEW
COMMON/TRIALT/AN1TRT (10) , AN2TRT (10) , AN3TRT (10) , CPRT (3, 3, 10) ,
1          DLOSRT (10) , DRATT (10) , DRGPR (10) , DXEPRT (10) ,
2          DYEPR (10) , DZEPRT (10) , FLDTRT (10) , FLOSRT (10) ,
3          ICNPR (10) , ICTRYT (10) , ISTPR (20, 10) , ISTRYT (10) ,
4          IVALTT, IVPRT (10) , KNEWT, KROTST, PSTRT (10) , PXEPRT (10) ,
5          PYEPRT (10) , PZEPRT (10) , RNPRT (10) , ROTNCT, ROTN2T,
6          ROTRYT (10) , THTRT (10) , TPOTRT (10) , XEXA, YEXA, ZEXA
COMMON/VARBLT/ACCXET, ACCYET, ACCZET, ALFAT, CBART (3, 3) , CBT (3, 3) , CDT,
1          CLALFT, CLT, CPHIT, COPSIT, COTHTT , CST, D (3, 3) , DRAGT,
2          FFLOST, FLIFTT, FLODMT, FMACHT, FMMAXT, FMINT, INIZT,
3          PHIBRT, PHIT, PMAXT, PPDTT, PT, PSIBRT, PSIT, QMAXT, QPDTT,
4          QT, RECANT, RHOT, RMAXT, RPDTT, RT, SIPHIT, SIPSIT, SITHTT ,
5          SPECET, ST, TABT, THETAT, THETBT, TIDLET, TMILT, TPOST,
6          TRSTT, UT, VELT, VHORT, VT, WEITT, WT, XEDOTT, XET, YEDOTT,
7          YET, ZEDOTT, ZET, FLODST
DIMENSION C (3, 3)
DIMENSION CBRTR (3, 3)
IF(ICRECT .LE. 0) GO TO 10

```

C
C
C
C

*** INITIALIZATION

```

DELTT = DTPRT
DTPRX = DTPRT * RADIFY (5HDTPRT, 1.0, 0.8, 1.2)
XMIN2A=0.
YMIN2A=0.
ZMIN2A=0.

```

```
XMIN1A=0.
YMIN1A=0.
ZMIN1A=0.
XZEROA=0.
YZEROA=0.
ZZEROA=0.
ITPRE=1
STORET=DELTT
TIMCN=0.
NOWPRT=0
GO TO 999
```

```
C
C *** END OF INIZIALIZATION
10 CONTINUE
   STORET=STORET+DT
   IF(STORET.GE.DELTT) GO TO 15
   GO TO 185
15 STORET=0.
   XMIN2A=XMIN1A
   YMIN2A=YMIN1A
   ZMIN2A=ZMIN1A
   XMIN1A=XZEROA
   YMIN1A=YZEROA
   ZMIN1A=ZZEROA
   XZEROA=XEA
   YZEROA=YEA
   ZZEROA=ZEA
   TLSAVE= TIME
185 CONTINUE
   IF(ITPRE.LE.0) GO TO 190
   IF(TIME.LT.TBEGN+DELTT+DELTT)GO TO 999
   ITPRE=0
   GO TO 200
190 CONTINUE
   NOWPRT=0
   TIMCN=TIMCN+DT
   IF(TIMCN.LT.DTPRX)GO TO 999
200 TIMCN=0.00001
   NOWPRT=1
   CALL STATET
   DTPRX = DTPRT*RADIFY(5HDTPRT,1.0,0.8,1.2)
   TIMEX = TIMEPT*RADIFY(6HTIMEPT,1.0,0.8,1.2)
   IF (ABS (RANGE+RRATE*TIMEX) .LT. ABS (RRATE*DTPRX) *0.5)
1     TIMEX = DTPRX
   TIMEXT= TIMEX+TIME-TLSAVE

C
C CALL EXTRT (C)
C -----
C
XEXA=XEXPA
YEXA=YEXPA
ZEXA=ZEXPA
DXEXA=XDEXPA
DYEXA=YDEXPA
```

```

DZEXA=ZDEXPA
VVELA=VELEXA
DO 650 I=1,3
DO 650 J=1,3
650 CRGR(I,J)=DRCEXA(I,J)
C
C *** DETERMINE SUITABLE TRIAL COMMANDS
C *****
C
VDUM = 1. / VELT
VSPS = YEDOTT/VHORT
VCPS = XEDOTT / VHORT
VSTH = - ZEDOTT * VDUM
VCTH = VHORT * VDUM
PSIBRT=ATAN2(YEDOTT,XEDOTT)
THETBT=ATAN(-ZEDOTT/VHORT)
SIPSIT=SIN(PSIBRT)
COPSIT=COS(PSIBRT)
SITHTT =SIN(THETBT)
COTHTT =COS(THETBT)
PHIBRT=ATAN2((AN3T/COTHTT ),(AN2T*COPSIT-AN1T*SIPSIT))
SIPHIT=SIN(PHIBRT)
COPHIT=COS(PHIBRT)
CALL CMTRX(PSIBRT,THETBT,PHIBRT,CBT)
CALL TRYNXT
C
C *** CALCULATE THRUST, DRAG, AND P SUB S FOR ALL TRIAL MANEUVERS
C
QBARS=VELT*VELT*RHOT*ST/2.
IABL = IABLET
DO 2700 I=1,NTRYT
IF(TPOTRT(I).EQ.0.) THTRT(I)=TIDLET*2.
IF(TPOTRT(I).EQ.1.) THTRT(I)=TMILT*2.
IF(TPOTRT(I).EQ.2.) THTRT(I)=TABT*2.
IABLET= TPOTRT(I)
FLIFTR=FLDTRT(I)*FLODMT*WEITT
CLTR=FLIFTR/QBARS
CLTRO = CLTR
CALL AERF4(FMACHT,CLTR,-4)
FLDTRT(I)= FLDTRT(I)/CLTRO*CLTR
ALPH = ALPHAX
DRATT (I)=QBARS*CDX
PSTRT(I)=(THTRT(I)*COS(ALPH)-DRATT (I))*VELT/WEITT
2700 CONTINUE
IABLET= IABL
IF(IVAR(1).GT.0) GO TO 50251
AA4 = ROTST * DEGRD
WRITE(46,50210) KROTST, NTRYT, AA4
50210 FORMAT ( '0 TRYNXT RETURNS KROTST, NTRYT, ROTST = '
1 2I5, F10.3 // ' I ISTRYT ICTRYT G S'
2 ' ROTRYT AN1TRT AN2TRT AN3TRT P SUB S'/)
DO 50250 I = 1, NTRYT
AA1=ROTRYT(I)*DEGRD
GLD = FLDTRT(I) * FLODMT

```

```

WRITE (46,50215) I, ISTRYT(I), ICTRYT(I), GLD, AA1, AN1TRT(I), AN2TRT(I)
1 , AN3TRT(I), PSTRT(I)
50215 FORMAT( 3I10, F10.4, 5F10.3 )
50250 CONTINUE
50251 CONTINUE
C
C *** OBTAIN PREDICTED SITUATION FOR TRIALS 1 THROUGH NTRYT
C *****
DO 600 I=1, NTRYT
C
IF(ISTRYT(I)) 400,380,400
380 CONTINUE
PHIBR=ATAN2((AN3TRT(I)/COTHTT), (AN2TRT(I)*COPSIT-AN1TRT(I)*
1 SIPSIT))
CALL CMTRX(PSIBRT, THETBT, PHIBR, CBRTR)
COMPGN =CBRTR(3,3)
ACCNR = FLDTRT(I) * FLODMT - COMPGN
CALL PRETNW (AN1TRT(I), AN2TRT(I), AN3TRT(I), CBRTR, ACCNR, TIMEX,
1 XEDOTT, YEDOTT, ZEDOTT, XET, YET, ZET, VELT)
GO TO 450
400 CALL PRESR(TIMEX, ISTRYT(I), VELT, XET, YET, ZET, VSTH, VCTH, VSPS, VCPS)
450 PXEPRT(I)=XXNEW
PYEPRT(I)=YYNEW
PZEPRT(I)=ZZNEW
DXEPRT(I)=VXNEW
DYEPRT(I)=VYNEW
DZEPRT(I)=VZNEW
DO 475 II=1,3
DO 475 J=1,3
475 CPRT(II,J,I)= CNEW(II,J)
600 CONTINUE
C
C *** DETERMINE VALUE FOR EACH SITUATION RESULTING FROM THE NTRYT
C *** TRIAL COMMANDS
C
IVALTT= 0
SPEN = SPENYA
SPENYA= SPENYT+(SPEN-SPENYT)*RADIFY(5HENOPT,1.0,0.9,1.1)
DO 850 K = 1, NTRYT
XEPRT = PXEPRT(K)
YEPRT = PYEPRT(K)
ZEPRT = PZEPRT(K)
PDXET = DXEPRT(K)
PDYET = DYEPRT(K)
PDZET = DZEPRT(K)
VVELT = SQRT ( PDXET * PDXET + PDYET * PDYET + PDZET * PDZET )
DO 810 II = 1,3
DO 810 JJ= 1,3
810 DRGR(II, JJ ) = CPRT(II, JJ, K )
DUMMY=1.
CALL RELGN(XEXA, YEXA, ZEXA, DXEXA, DYEXA, DZEXA, XEPRT, YEPRT, ZEPRT,
1 PDXET, PDYET, PDZET, DUMMY, FFLOST, TIMEXT, VVELA, VVELT)
RANGE = RANGE*RADIFY(5HRANGT,1.0,0.9,1.1)
RRATE = RRATE*RADIFY(5HRANGT,1.0,0.9,1.1)

```

```

FLOSST= FLOSST*RADIFY(4HLOST,1.0,0.9,1.1)
DLOSST= DLOSST*RADIFY(4HLOST,1.0,0.9,1.1)
FLOSSA= FLOSSA*RADIFY(4HLOST,1.0,0.8,1.2)
FLOSST= AMIN1(PI,FLOSST)
FLOSSA= AMIN1(PI,FLOSSA)
PSUBST=PSTRT(K)
CALL STATET
DO 830 L = 1, NCLSTT
830 ISTRPT (L,K) = LCELLT( L )
RNGPRT(K) = RANGE
DRGPRT(K) = RRATE
ICNPRT(K) = NCELLT
IVPRT(K) = JVALUT
FLOSRT(K) = FLOSST
DLOSRT(K) = DLOSST
IVALTT= MAX0(IVALTT,JVALUT)
850 CONTINUE
SPENYA= SPEN
C
C *** SELECT THE MOST PROMISING OF THE TRIAL COMMANDS
C *** *****
C
IF(IVALTT.NE.IVPRT(KROTST))GO TO 120
KNEWT = KROTST
GO TO 300
120 CONTINUE
TESROT= PI
KNEWT = 1
DO 130 K=1,NTRYT
IF(IVALTT.NE.IVPRT(K))GO TO 130
TEROT = ABS(ROTRYT(K)-ROTST)
IF(TEROT.GT.PI)TEROT= TWOPI-TEROT
IF(TEROT.GE.TESROT)GO TO 130
TESROT= TEROT
KNEWT = K
130 CONTINUE
300 CONTINUE
ICMNWT = 1
ISTRGT = ISTRYT( KNEWT )
ICOMDT = ICTRYT( KNEWT )
ROTT=ROTRYT(KNEWT)
GLEVLT = FLDTRT( KNEWT )
AN1T= AN1TRT(KNEWT )
AN2T = AN2TRT(KNEWT )
AN3T = AN3TRT(KNEWT )
TPOST=TPOTRT(KNEWT)
IF(IVAR(1).GT.0) GO TO 999
WRITE(46,145)TIMEX,TIME
145 FORMAT(/' TARGET TACTICS PREDICTS IN 'F6.2' SECONDS FROM CURRENT
1IME' F10.4' SECONDS'//
2 ' COMAND RANGE RATE' ' CELL STATE' 43X
3 'STATE STATE' / 9X ' FEET FPS' 58X
4 'NO VALUE' / )
C

```

```

DO 500 K = 1, NTRYT
IRAN = RNGPRT(K)
IRR = DRGPRT(K)
ISM = ISTRYT(K)
ICM = ICTRYT(K)
IF( ISM .NE.0) ICM = 0
WRITE(46,150) ISM, ICM, IRAN, IRR, ( ISTRPT(L,K), L=1, NCLSTT )
1 ICNPRT(K), IVPRT(K)
150 FORMAT( I3, I4, I7, I6, 5I3,3X,5I3,3X,5I3,      3X,      2I6 )
500 CONTINUE

```

C

```

WRITE(46,550)KNEWT
550 FORMAT(/' TARGET SELECTS TRIAL MANEUVER NUMBER' I4)
999 CONTINUE
RETURN
END

```

SUBROUTINE STATET

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

```

STATET COMPUTES THE CELL STATE AND VALUE FOR TARGET CELL STRUCTU
INPUT IS RELATIVE GEOMETRY FROM ROUTINE RELGN
ROUTINE RELGN MUST BE CALLED JUST BEFORE CALL OF STATET
OUTPUT IS
CELL STATE VECTOR, LCELLT( 40)
CELL VALUES FOR THE INDIVIDUAL CELLS, LVALT(40)
CELL NUMBER, NCELLT
VALUE OF THE TOTAL CELL STRUCTURE, JVALVT

```

```

COMMON/ATTTAR/ANGETA, ANGETT, ANGXIA, ANGXIT, CRGR(3,3), DLOSSA,
1 DLOSST, DRGR(3,3), FAZMSA, FAZMST, FELVSA, FELVST, FLOSSA,
2 FLOSST, PSUBSA, PSUBST, RANGE, RRATE, XAINT, XTINA, YAINT,
3 YTINA, ZAINT, ZTINA
COMMON/CELSTT/AN1T, AN2T, AN3T, DTPRT, GLEVL, ICMNWT, ICOMDT, IRECT,
1 ISTRGT, JVALUT, LCELLT(40), LVALT(40), LWEGHT(40),
2 NCELLT, NCLSTT, NOWPRT, NTILTT, NTRYT, ROTST, ROTT, TIMEPT
COMMON/ENERGY/SPENYA, SPENYT
COMMON/ K4/DT, TBEGN, TIME, PI, PIDV2, PIDV4, TWOPI, DEGRD, RADDG, G
1 ,VAR(20), IVAR(20), TEND
DATA CLOS/300.0/
1 ,SEPR/300.0/
2 ,FLOSX/1.0/
3 ,DLOSX/0.1/
DATA RANG1/2000.0/, RANG2/7000.0/
DATA KNOWOP/2/

```

C

C

C

C

C

C

C

C

C

C

C

C

C

C

```

DO 125 I=1, NCLSTT
125 LCELLT(I)=0

```

```

DETERMINE THE TARGET STATE VECTOR, LCELLT

```

```

LCELLT( I) = 1 IF ASSOCIATED STATEMENT I IS TRUE

```



```

C
C VOLUME BEHIND AIRCRAFT
C VOLUME IS A THIRTY DEGREE CONE FOR 3000 FEET BEHIND
C VOLUME IS A CONE CHANGING TO 45 DEGREE AT 3000 FEET BACK
C VOLUME IS LIMITED TO 5000 FEET BEHIND
C
RSQ1 = YAIN T * YAIN T + ZAIN T * ZAIN T
IF(LCELLT(4).EQ.1)GO TO 431
IF(XAIN T)405,431,431
405 CONTINUE
CALL FIRCON(KNOWOP,RANGE,0.0,0.0,FAA)
IF(FAA.EQ.0.0)GO TO 431
C *** RSQ1 IS SQUARE OF ( DISTANCE FROM ATTACKER CG TO TARGET BODY X
410 IF(RANGE- 3000.)411,411,415
411 IF( RSQ1 - .57735 * XAIN T * .57735 * XAIN T) 430,430,431
C
C XSHIFT IS TARGET BODY X AZIS LOCATION 45 DEGREE CONE APEX
C XPRI IS DISTANCE ALONG TARGET BODY X AXIS, FROM 45 DEGREE CONE
C POINT ON X AXIS CLOSEST TO ATTACKER
C
415 XSHIFT = - 3000. * .36603
XPRI = XAIN T - XSHIFT
IF ( RSQ1 - XPRI * XPRI ) 430,430,431
431 LCELLT( 6 ) = 1
430 CONTINUE
C
RSQ2 = YTINA * YTINA + ZTINA * ZTINA
IF(LCELLT(3).EQ.0)GO TO 531
IF(XTINA)505,531,531
505 CONTINUE
CALL FIRCON(2,RANGE,0.0,0.0,FAA)
IF(FAA.EQ.0.0)GO TO 531
510 IF(RANGE-3000.)511,511,515
511 IF( RSQ2 - .57735 * XTINA * .57735 * XTINA ) 530,530,531
515 XSHIFA = -3000. * .36603
XPRIM = XTINA - XSHIFA
IF ( RSQ2 - XPRIM * XPRIM ) 530,530,531
530 LCELLT( 5 ) = 1
531 CONTINUE
C
C ABILITY TO ATTACK
C OPPONENT WITHIN 30 DEGREES OF VELOCITY VECTOR
C
FAA = 0.0
IF(LCELLT(1).EQ.1)GO TO 481
CALL FIRCON(KNOWOP,RANGE,FLOSSA,0.0,FAA)
481 CONTINUE
LCELLT(8)= 1.0-FAA
C
IF(LCELLT(2).EQ.0)GO TO 581
CALL FIRCON(2,RANGE,FLOSST,0.0,FAA)
LCELLT(7)= FAA
581 CONTINUE
C
C *** CLOSING

```

```

C
REFDIS= RRATE*DTPRT
IF (ABS (REFDIS) .LT.0.1)REFDIS= 0.1
IF (LCELLT(5) .EQ.0.AND.LCELLT(7) .EQ.0)GO TO 1070
DECTO1=(RANG1-RANGE)/REFDIS
DECTO2=(RANG2-RANGE)/REFDIS
IF (DECTO1*DECTO2.GE.0.0)GO TO 920
C
WITHIN RANG1 - RANG2
IF (ABS ((RANG2-RANG1)/REFDIS) .LT.5.0)GO TO 950
C
LONG RESIDENCE TIME
LCELLT(10)= 1
GO TO 950
920 CONTINUE
C
APPROACH SPEED
DECTOM= AMAX1 (DECTO1,DECTO2)
IF (DECTOM.LE.0.0)GO TO 1070
IF (DECTOM.GT.10.0)GO TO 1070
IF (ABS ((RANG2-RANG1)/REFDIS) .LT.2.0)GO TO 1070
950 CONTINUE
LCELLT(9)= 1
1070 CONTINUE
C
C
C
SEPARATING
C
LCELLT(11)= 1
LCELLT(12)= 1
IF (LCELLT(6) .NE.0.AND.LCELLT(8) .NE.0)GO TO 1270
DECTO1=(RANG1-RANGE)/REFDIS
DECTO2=(RANG2-RANGE)/REFDIS
IF (DECTO1*DECTO2.GE.0.0)GO TO 1120
C
WITHIN RANG1 - RANG2
IF (ABS ((RANG2-RANG1)/REFDIS) .LT.5.0)GO TO 1150
C
LONG RESIDENCE TIME
LCELLT(12)= 0
GO TO 1150
C
1120 CONTINUE
C
C
SEPARATION SPEED
DECTOM= AMAX1 (DECTO1,DECTO2)
IF (DECTOM.LE.0.0)GO TO 1270
IF (DECTOM.GT.10.0)GO TO 1270
IF (ABS ((RANG2-RANG1)/REFDIS) .LT.2.0)GO TO 1270
1150 CONTINUE
LCELLT(11)= 0
1270 CONTINUE
C
C
C
*** LINE OF SIGHT, LOS
C
IF ( FLOSST .LE. FLOSX ) LCELLT ( 13 ) = 1
C
C
RATE OF CHANGE OF ANGLE LOS
C
IF (DLOSST.GT. (DLOSX-FLOSST)/DTPRT*0.1)GO TO 1450

```

```

IF(DLOSST.LT.(DLOSX+FLOSST)/DTPRT*(-0.5))GO TO 1450
IF(XTINA.GT.0.0.AND.RRATE.LT.-CLOS)GO TO 1450
LCELLT(14)= 1
1450 CONTINUE
IF((PSUBST+100.0)*DTPRT*5.0+SPENYT-SPENYA.GT.0.0)LCELLT(15)= 1
C
C   CALCULATE VALUE OF EACH INDIVIDUAL STATE, VECTOR LVALT
C
DO 10010 I =1, NCLSTT
10010 LVALT(I) = LCELLT(I) * LWEGHT( I)
C
C   CALCULATE TOTAL CELL STRUCTURE VALUE, JVALUT
C   AND CELL NUMBER, NCELLT
C
JVALUT = 0
NCELLT = 0
DO 10050 I = 1, NCLSTT
JVALUT = JVALUT + LVALT(I)
NCELLT = NCELLT + LCELLT(I)*2**(I-1)
10050 CONTINUE
C
C   PASS ELEVATION AND AZIMUTH TO LOS FROM TARGET X BODY AXIS , AS
C   FELVST, FAZMST, TO ROUTINE PRCELT
C
FELVST = ANGETT
FAZMST = ANGXIT
C
RETURN
END

```

SUBROUTINE EQMOTA

C-----
 C
 C *** THIS SUBROUTINE CALCULATES FOR A GIVEN MANEUVER STATUS AND A GIVEN
 C *** VELOCITY VECTOR THE ACCELERATIONS OF THE CG OF THE AIRCRAFT IN
 C *** THE INERTIAL AXIS SYSTEM. IT ALSO CALCULATES THE PRESENT ATTITUDE
 C *** EXPRESSED BY THE DIRECTION COSINE MATRIX OF THE BODY AXIS
 C *** SYSTEM (C(I,J))
 C *** IT ALSO CALCULATES APPROXIMATE VALUES OF THE EULER ANGLE RATES
 C *** AND THE BODY AXIS VELOCITIES

C DIMENSION CDES(3,3)

C
 COMMON/AEROUT/TIDLEX ,TMILX ,TABX
 1 ,FLODMX ,FLODSX
 2 ,FMMINX ,FMMAXX ,RECANX
 3 ,ALPHAX
 4 ,CDX ,SBCDX
 COMMON/ATTTAR/ANGETA, ANGETT, ANGXIA, ANGXIT, CRGR(3,3), DLOSSA,
 1 DLOSST, DRGR(3,3), FAZMSA, FAZMST, FELVSA, FELVST, FLOSSA,
 2 FLOSST, PSUBSA, PSUBST, RANGE, RRATE, XAINT, XTINA, YAINT,
 3 YTINA, ZAINT, ZTINA
 COMMON/CELSTA/AN1A, AN2A, AN3A, DTPRA, GLEVLA, ICMNWA, ICOMDA, ICRECA,
 1 ISTRGA, JVALUA, LCELLA(40), LVALA(40), LWEGHA(40),
 2 NCELLA, NCLSTA, NOWPRA, NTLTA, NTRYA, ROTSA, ROTA, TIMEPA
 COMMON /ERNION/ AB(4,2)
 COMMON/HASSLE/IABLEA ,IABLET
 COMMON/ K4/DT, TBEGN, TIME, PI, PIDV2, PIDV4, TWOPI, DEGRD, RADDG, G
 1 ,VAR(20), IVAR(20), TEND
 COMMON/VARBLA/ACCXEA, ACCYEA, ACCZEA, ALFAA, CBARA(3,3), CBA(3,3), CDA,
 1 CLALFA, CLA, COPHIA, COPSIA, COTHTA ,CSA, C(3,3), DRAGA,
 2 FFLOSA, FLIFTA, FLODMA, FMACHA, FMMAXA, FMMINA, INIZA,
 3 PHIBRA, PHIA, PMAXA, PPDTA, PA, PSIBRA, PSIA, QMAXA, QPDTA,
 4 QA, RECANA, RHOA, RMAXA, RPDTA, RA, SIPHIA, SIPSIA, SITHTA ,
 5 SPECEA, SA, TABA, THETA, THETBA, TIDLEA, TMILA, TPOSA,
 6 TRSTA, UA, VELA, VHORA, VA, WEITA, WA, XEDOTA, XEA, YEDOTA,
 7 YEA, ZEDOTA, ZEA, FLODSA
 COMMON/ENERGY/SPENYA, SPENYT
 IF(INIZA)8,8,1

C
 C *** INITIALIZATION AT BEGIN OF A RUN
 C-----

C
 C
 C
 C
 1 CONTINUE
 PMAXA=90./DEGRD
 QMAXA = 25.0/DEGRD
 RMAXA=15./DEGRD
 PPDTA=PMAXA*DT
 QPDTA=QMAXA*DT
 RPDTA=RMAXA*DT
 CALL FILTRA(1,1.,1.,1)
 NROLL=0
 ICOMO=ICOMDA
 ISTRO=ISTRGA

```

PHIO=PHIA
PHINST= PHIA
THETO=THETAA
PSIO=PSIA
VHORA=SQRT (XEDOTA**2+YEDOTA**2)
VELA=SQRT (VHORA**2+ZEDOTA**2)
CALL CMTRX (PSIA, THETAA, PHIA, C)
CALL NORPLN (XEDOTA, YEDOTA, ZEDOTA, ROTA, AN1A, AN2A, AN3A)

```

C

```

HA=-ZEA
CALL CSRHO (HA, CS, RHO)
RHOA=RHO
CSA=CS
FMACHA=VELA/CSA
IABLEA= TPOSA
CALL AERF4 (HA, FMACHA, 3)
IF (ISTRGA.EQ.1) GLEVLA=(VHORA/VELA)/FLODMX
IF (ISTRGA.EQ.-1) GLEVLA=- (VHORA/VELA)/FLODMX
GLEVO = GLEVLA
RETURN

```

C

C *** AT 8 IS NORMAL ENTRY DURING OPERATION

C

C

8 CONTINUE

```

HA=-ZEA
CALL CSRHO (HA, CS, RHO)
RHOA=RHO
CSA=CS
FMASSA=WEITA/G
VHORS=XEDOTA**2+YEDOTA**2
VHORA=SQRT (VHORS)
VSQ =VHORS+ZEDOTA**2
QBARS=VSQ*RHOA*SA/2.
VELA=SQRT (VSQ )
SPECEA= HA+VSQ / (2.*G)
SPENYA= SPECEA
FMACHA=VELA/CSA
PSIBRA=ATAN2 (YEDOTA, XEDOTA)
THETBA=ATAN (-ZEDOTA/VHORA)
SIPSIA=SIN (PSIBRA)
COPSIA=COS (PSIBRA)
SITHTA =SIN (THETBA)
COTHTA =COS (THETBA)
PHIA = PHINST
ALT = -ZEA
IABLEA= TPOSA
1220 CALL AERF4 (ALT, FMACHA, 3)
RECANA=RECANX
FMMINA=FMMINX
FMMAXA=FMMAXX
TMILA=TMILX
TIDLEA=TIDLEX
TABA=TABX

```

```

      FLODMA=FLODMX
      FLODSA=FLODSX
C
800 CONTINUE
      MAXCL = 0
      MAXPHI= 0
802 CONTINUE
      IF(MAXPHI.LE.2)GO TO 804
      CALL ERMSG(12,SIPHS)
C
999 TIME =TEND+2.*DT
      RETURN
804 CONTINUE
C
      OBTAIN AERO DATA
      FLOADA=GLEVLA*FLODMA
      FLIFTA=WEITA*FLOADA
      CLA=FLIFTA/QBARS
      CLAO = CLA
      CALL AERF4(FMACHA,CLA,4)
      IF(CLA.EQ.CLAO)GO TO 805
      IF(MAXCL.EQ.0)GO TO 803
      CALL ERMSG(4,CLA)
      GO TO 999
803 CONTINUE
      MAXCL = 1
      GLEVLA= GLEVLA/CLAO*CLA
      FLOADA= GLEVLA*FLODMA
      FLIFTA= WEITA*FLOADA
805 CONTINUE
      ALFAA = ALPHAX
      SIALF=SIN(ALFAA)
      CSALF=COS(ALFAA)
      CDA=CDX
      DRAGA=QBARS*CDA
C *** OBTAIN THRUST   TPOS=0   IDLE
C                               TPOS=1   MILITARY
C                               TPOS=2   AFTERBURNER
C *** THRUST TABLE OF F-4 IS FOR ONE ENGINE ONLY
C
      IF(TPOSA.EQ.0.0)TRSTA= TIDLEA*2.0
      IF(TPOSA.EQ.1.0)TRSTA= TMILA*2.0
      IF(TPOSA.EQ.2.0)TRSTA= TABA*2.0
      GLTFL = FLOADA+TRSTA/WEITA*SIALF
      IF(GLTFL.EQ.0.0)GLTFL= 1.0E-06
C
      IF(ICMNA)30,30,810
810 CONTINUE
C
C *** THIS PORTION WHEN THE COMMAND HAS CHANGED,
C *** CALCULATE DESIRED CHANGE IN ROLL ANGLE
C *** AND HOW MANY DT WE ARE GOING TO ROLL WITH MAX ROLL RATE
C
      IF(PHIA.GE.-PI)GO TO 812
      PHIA = PHIA+TWOPI

```

```

      PHIO = PHIO+TWOPI
      GO TO 814
812  CONTINUE
      IF(PHIA.LE.PI)GO TO 814
      PHIA = PHIA-TWOPI
      PHIO = PHIO-TWOPI
814  CONTINUE
      IF(ISTRGA) 830,850,820
820  DPHI=-PHIA
      GO TO 8700
830  DPHI=PI-PHIA
      GO TO 8700
850  CONTINUE
      CALL CMTRX(PSIBRA,THETBA,ROTA,CDES)
      SIPHS =-CDES(2,3)/GLTFL
      IF(ABS(SIPHS).LE.1.0E-06)SIPHS= 0.0
      SIPHIA= SIN(ROTA)
      IF(ABS(SIPHIA).LE.1.0E-06)SIPHIA= 0.0
      IF(ABS(SIPHS).GT.ABS(SIPHIA))GO TO 225
      PHIS=ASIN(SIPHS)
      PHIDS=ROTA+PHIS
      DPHI=PHIDS-PHIA
8700 CONTINUE
      ADPHI=ABS(DPHI)
      IF(ABS(PI-ADPHI).LE.1.0E-04)GO TO 860
      IF(PI-ADPHI) 855,860,870
870  SIGNR=1.
      IF(DPHI.LT.0.) SIGNR=-1.
      GO TO 880
855  SIGNR=-1.
      IF(DPHI.LT.0.) SIGNR=1.
      DPHI=TWOPI-ABS(DPHI)
      GO TO 880
860  DPHI=PI
      ADPHI = DPHI
      IF(YTINA.EQ.0.0)GO TO 8601
      SIGNR = YTINA/ABS(YTINA)
      GO TO 8602
8601 SIGNR = 1.0
8602 CONTINUE
      880 NROLL=(ABS(DPHI)/PPDTA)
      ICMNWA=0
C *** CHECK IF LAST TRANSITION IS COMPLETED
      30 CONTINUE
C
      IF(NROLL) 31,32,500
      31 CALL ERMSG(1,DUM)
      NROLL = 0
      32 CONTINUE
C
C *** WE ARE IN THE MANEUVER MODE
C -----
      PHIBRA=ATAN2((AN3A/COTHTA ),(AN2A*COPSIA-AN1A*SIPSIA))

```



```

COPHIA=COS (PHIBRA)
SIPHIA=SIN (PHIBRA)
IF (ABS (SIPHIA) .LE.1.0E-06)SIPHIA= 0.0
CALL CMTRX (PSIBRA, THETBA, PHIBRA, CBA)
C
C CHECK FOR STRAIGHT FLIGHT
C
IF (ISTRO) 210, 220, 230
230 PHIA=0.
GO TO 255
210 PHIA=-PI
GO TO 255
220 CONTINUE
SIPHS =-CBA (2, 3)/GLTFL
IF (ABS (SIPHS) .LE.1.0E-06)SIPHS= 0.0
IF (ABS (SIPHS) .LE.ABS (SIPHIA))GO TO 240
225 CONTINUE
MAXPHI= MAXPHI+1
IF (MAXCL.NE.0)GO TO 235
C INCREASE G-LEVEL
GLEVLA= GLEVLA/ABS (SIPHIA) * (ABS (SIPHS) +1.0E-06)
IF (ABS (GLEVLA) .LE.1.0)GO TO 802
GLEVLA= 1.0
235 CONTINUE
C VARY MANEUVER PLANE
IF (ABS (PHIBRA) -PIDV2) 237, 237, 238
237 CONTINUE
PHIBRA= 0.0
GO TO 239
238 CONTINUE
PHIBRA= PI
239 CONTINUE
ROTA = PHIBRA
CALL NORPLN (XEDOTA, YEDOTA, ZEDOTA, PHIBRA, AN1A, AN2A, AN3A)
GO TO 802
240 CONTINUE
PHIS=ASIN (SIPHS)
250 PHIA=PHIBRA+PHIS
255 CONTINUE
C
GO TO 1000
C
C *** 500 FOR CASE WHEN WE ARE IN TRANSITION MODE
C -----
C
500 CONTINUE
C
FLIFTA=FLIFO
ALFAA=ALFO
DRAGA=DRAGO
TRSTA=TRSTO
CSALF=CSALO
SIALF=SIALO
C

```

```

PHIA=PHIA+SIGNR*PPDTA
PHIBRA= PHIA
C
NROLL=NROLL-1
IF(NROLL) 31,910,1000
910 CONTINUE
C
C *** 910 FOR CALCULATION OF NEW MANEUVER PLANE NORMAL (PLANE TROUGH
C *** PRESENT VELOCITY VECTOR WITH PREVIOUSLY DETERMINED ANGLE ROTA)
CALL NORPLN(XEDOTA,YEDOTA,ZEDOTA,ROTA,AN1A,AN2A,AN3A)
C
C *** 1000 COMBINED PATH FOR ALL MANEUVER MODES
C -----
C
1000 CONTINUE
C
C OBTAIN DIRECTION COSINE MATRIX OF INSTANTANEOUS MANEUVER PLANE SYS
C
CALL CMTRX(PSIBRA,THETBA,PHIA,CBARA)
C
C CALCULATE FORWARD FORCE (ALONG MANEUVER PLANE X-AXIS)
C AND NORMAL FFORCE (POSITIVE ALONG THE POSITIVE ZM AXIS)
C INCLUDE AERODYNAMIC FORCES AND THRUST, BUT NOT WEIGHT
C
FF=TRSTA*CSALF -DRAGA
FN=-FLIFTA-TRSTA*SIALF
C
C TRANSFORM FORCES INTO INERTIAL SYSTEM AND ADD WEIGHT
C
FXEA=FF*CBARA(1,1)+FN*CBARA(3,1)
FYEA=FF*CBARA(1,2)+FN*CBARA(3,2)
FZEA=FF*CBARA(1,3)+FN*CBARA(3,3) + WEITA
C
ACCXEA=FXEA/FMASSA
ACCYEA=FYEA/FMASSA
ACCZEA=FZEA/FMASSA
PSUBSA= FF/WEITA*VELA
C
C OBTAIN PROJECTIONS OF X-BODY AXIS ONTO THE THREE INERTIAL AXES
XBX=CSALF*CBARA(1,1)-SIALF*CBARA(3,1)
XBY=CSALF*CBARA(1,2)-SIALF*CBARA(3,2)
XBX=CSALF*CBARA(1,3)-SIALF*CBARA(3,3)
C
C NOW WE CAN GET THE CURRENT EULER ANGLES OF THE BODY AXIS SYSTEM
C
PSIA=ATAN2(XBY,XBX)
THETAA=ATAN(-XBX/SQRT(XBX**2+XBY**2))
C
C *** OBTAIN DIRECTION COSINE MATRIX OF BODY AXIS SYSTEM
C
PHINST= PHIA
CALL NORPLN(XEDOTA,YEDOTA,ZEDOTA,PHIA,AN1,AN2,AN3)
PHIA = ATAN2(AN3/COS(THETAA),AN2*COS(PSIA)-AN1*SIN(PSIA))
CALL CMTRX(PSIA,THETAA,PHIA,CDES)

```

C
C
C

OBTAIN APPROXIMATE EULER ANGLE RATES

```
DELPHI=PHIA-PHIO
IF(DELPHI.LT.(-PI)) DELPHI=TWOPI+DELPHI
IF(DELPHI.GT.PI) DELPHI=-(TWOPI-DELPHI)
DELTHT =THETAA-THETO
IF(DELTHT .LT.(-PI)) DELTHT =TWOPI+DELTHT
IF(DELTHT .GT.PI) DELTHT =-(TWOPI-DELTHT )
DELPSI=PSIA-PSIO
IF(DELPSI.LT.(-PI)) DELPSI=TWOPI+DELPSI
IF(DELPSI.GT.PI) DELPSI=-(TWOPI-DELPSI)
PHIDOT=DELPHI/DT
THETDT =DELTHT /DT
PSIDOT=DELPSI/DT
COTHEO= THETO+DELTHT *0.5
PA      = PHIDOT-PSIDOT*SIN(COTHEO)
COTHEO= COS(COTHEO)
COPHIO= PHIO+DELPHI*0.5
SIPHIO= SIN(COPHIO)
COPHIO= COS(COPHIO)
QA      = THETDT *COPHIO+PSIDOT*SIPHIO*COTHEO
RA      = PSIDOT*COPHIO*COTHEO-THETDT *SIPHIO
```

C
C
C

*** FILTER THE BODY RATES WHICH GO TO THE DMS

C

```
CALL FILT RA(1,PA,PA,0)
CALL FILT RA(2,QA,QA,0)
CALL FILT RA(3,RA,RA,0)
IF(ABS(PA).GT.PMAXA+1.0E-06)CALL ERMSG(24,PA)
IF(ABS(QA).GT.QMAXA+1.0E-06)CALL ERMSG(25,QA)
IF(ABS(RA).GT.RMAXA+1.0E-06)CALL ERMSG(26,RA)
CALL QUATA(PA,QA,RA,AB(1,1))
CALL QUATEX(AB(1,1),C(1,1))
CALL OILER(C(1,1),PSIO,THETO,PHIO)
ANGAX = 1.0
DO 1410 I=1,3
DUMMY = 0.0
DO 1405 J=1,3
DUMMY = DUMMY+CDES(I,J)*C(I,J)
```

1405 CONTINUE

```
ANGAX = AMIN1(ANGAX,DUMMY)
```

1410 CONTINUE

C

```
IF(ANGAX.LT.0.988)PRINT 925,PSIA,PSIO,THETAA,THETO,PHIA,PHIO,TIME
925 FORMAT(24H A-ATTITUDE DISCREPANCY ,3(4X,2F8.4),10H AT TIME ,F8.3)
```

C

C

OBTAIN BODY AXES VELOCITIES

C

```
UA=XEDOTA*C(1,1)+YEDOTA*C(1,2)+ZEDOTA*C(1,3)
VA=XEDOTA*C(2,1)+YEDOTA*C(2,2)+ZEDOTA*C(2,3)
WA=XEDOTA*C(3,1)+YEDOTA*C(3,2)+ZEDOTA*C(3,3)
```

C

C

SAVE PRESENT VALUES AS OLD VALUES

C

ICOMO=ICOMDA
ISTRO=ISTRGA
GLEVO=GLEVLA
FLIFO=FLIFTA
ALFO=ALFAA
TRSTO=TRSTA
DRAGO=DRAGA
CSALO=CSALF
SIALO=SIALF

C

RETURN
END

SUBROUTINE EXTRA(D)

C

C

C

C

C

EXTRAPOLATION OF TARGETS POSITION, VELOCITY AND ATTITUDE BY THE
ATTACKER

COMMON/EXTPOA/TIMEXA, DELTA, XZEROT, YZEROT, ZZEROT, XMIN1T, YMIN1T,
2 ZMIN1T, XMIN2T, YMIN2T, ZMIN2T, XEXPT, YEXPT, ZEXPT,
2 XDEXPT, YDEXPT, ZDEXPT, VELEXT, DRCEXT(3,3)
COMMON/ K4/DT, TBEGN, TIME, PI, PIDV2, PIDV4, TWOPI, DEGRD, RADDG, G
1 ,VAR(20), IVAR(20), TEND
DIMENSION D(3,3)

C

C

C

C

CHECK WHETHER THE THREE POINTS USED FOR EXTRAPOLATION LIE IN
A STRAIGHT LINE

U1 = XMIN2T - XZEROT
U2 = YMIN2T - YZEROT
V1 = XMIN1T - XZEROT
V2 = YMIN1T - YZEROT
U3 = ZMIN2T - ZZEROT
V3 = ZMIN1T - ZZEROT
UXV1 = U2 * V3 - U3 * V2
UXV2 = -U1 * V3 + U3 * V1
UXV3 = U1 * V2 - U2 * V1
ILINET = 0
UXVMG = UXV1 * UXV1 + UXV2 * UXV2 + UXV3 * UXV3

C

IF U CROSS V EQUALS ZERO, THEN THE THREE POINTS ARE COLINEAR
IF(UXVMG - .001) 10,15,15

10 ILINET = 1

GO TO 20

15 ANORM = 1./ SQRT(UXVMG)

C

DRC21 = UXV1 * ANORM
DRC22 = UXV2 * ANORM
DRC23 = UXV3 * ANORM

20 CONTINUE

DTSQ = DELTA * DELTA

DXSQ = TIMEXA * TIMEXA

AX = (.5 * (XZEROT + XMIN2T) - XMIN1T) / DTSQ

```

AY = (.5 * ( YZEROT + YMIN2T ) - YMIN1T ) / DTSQ
AZ = (.5 * ( ZZEROT + ZMIN2T ) - ZMIN1T ) / DTSQ
BX = (3.*XZEROT-4.*XMIN1T+XMIN2T) / (2.*DELTA)
BY = (3.*YZEROT-4.*YMIN1T+YMIN2T) / (2.*DELTA)
BZ = (3.*ZZEROT-4.*ZMIN1T+ZMIN2T) / (2.*DELTA)
XEXPT=AX*DXSQ+BX*TIMEXA+XZEROT
YEXPT=AY*DXSQ+BY*TIMEXA+YZEROT
ZEXPT=AZ*DXSQ+BZ*TIMEXA+ZZEROT
XDEX=2.*AX*TIMEXA+BX
YDEX=2.*AY*TIMEXA+BY
ZDEX=2.*AZ*TIMEXA+BZ
XDEXPT=XDEX
YDEXPT=YDEX
ZDEXPT=ZDEX
VELPT = SQRT(XDEX * XDEX + YDEX * YDEX + ZDEX * ZDEX )
VELEXT=VELPT

```

C
C
C

OBTAIN DIRECTION COSINES

```

IF(ILINET ) 38,38,39
38 FACT = 1. / VELPT
DRC11 = XDEX * FACT
DRC12 = YDEX * FACT
DRC13 = ZDEX * FACT
DRCEXT(1,1) = DRC11
DRCEXT(1,2) = DRC12
DRCEXT(1,3) = DRC13
DRCEXT(2,1) = DRC21
DRCEXT(2,2) = DRC22
DRCEXT(2,3) = DRC23
DRCEXT(3,1) = DRC12 * DRC23 - DRC13 * DRC22
DRCEXT(3,2) = DRC13 * DRC21 - DRC11 * DRC23
DRCEXT(3,3) = DRC11 * DRC22 - DRC12 * DRC21
GO TO 40

```

C
C
C
C

FOR COLINEAR CASE EXTRAPOLATED DIRECTION COSINES ARE EQUAL TO THE PRESENT ONES

```

39 DO 3901 I = 1,3
    DO 3901 J = 1,3
3901 DRCEXT(I,J) = D ( I,J )
40 CONTINUE
RETURN
END

```

SUBROUTINE FILTRA (NPQR,RATED,RATEAC,INIZ)

C-----
C

```

IF(INIZ.EQ.0) GO TO 100
DEGRD=180./(4.*ATAN(1.))
PMAx=90./DEGRD
QMAX = 25.0/DEGRD
RMAX=15./DEGRD

```

```

SPP=0.
SPQ=0.
SPR=0.
FF=0.6
RETURN
100 GO TO (110,120,130) NPQR
110 PP=SPP
    PMX=PMAX
    GO TO 200
120 PP=SPQ
    PMX=QMAX
    GO TO 200
130 PP=SPR
    PMX=RMAX
200 CONTINUE
    IF(RATED-PMX) 210,213,211
211 RATEAC=AMIN1 (PP+FF*PMX,PMX)
    GO TO 215
210 IF(RATED+PMX) 212,213,213
212 RATEAC=AMAX1 (PP-FF*PMX,-PMX)
    GO TO 215
213 RATEAC=PP+FF*(RATED-PP)
215 GO TO(2210,2220,2230) NPQR
2210 SPP=RATEAC
    GO TO 3000
2220 SPQ=RATEAC
    GO TO 3000
2230 SPR=RATEAC
    GO TO 3000
3000 RETURN
    END

```

SUBROUTINE QUATA(P,Q,R,UAT)

```

C-----
C
COMMON/      K4/DT,TBEGN,TIME,PI,PIDV2,PIDV4,TWOPI,DEGRD,RADDG,G
1            ,VAR(20),IVAR(20),TEND
SAVE
EQUIVALENCE (DT,H)
DIMENSION  UAT(4)
DIMENSION  AT(4)
REAL IK,JK,KK
EQUIVALENCE (C1,CRHOK)
DATA OM2ZER /1.E-6/

C
C QUATERNION INTEGRATION
OMEG K2= P*P+Q*Q+R*R
IF(OMEG K2 .LT.OM2ZER ) OMEG K2  =OM2ZER
OMEGAK   =SQRT(OMEG K2  )
RHOK    =H*OMEGAK   *.5
SRHOK   =SIN(RHOK   )
CRHOK   =COS(RHOK   )

```

```

C2P  =SRHOK /OMEGAK
C3P  =2.*(1.-CRHOK )/OMEG K2
C4   =4.*(H-2.*C2P )/OMEG K2
IK   =-C2P*R
JK   = C2P*Q
AT(1) = C1*UAT(1)+IK*UAT(2)-JK*UAT(3)-KK*UAT(4)
KK   = C2P*P
AT(2) =-IK*UAT(1)+C1*UAT(2)-KK*UAT(3)+JK*UAT(4)
AT(3) = JK*UAT(1)+KK*UAT(2)+C1*UAT(3)+IK*UAT(4)
AT(4) = KK*UAT(1)-JK*UAT(2)-IK*UAT(3)+C1*UAT(4)
C    NORMALIZATION
ORMAL = 1.0/SQRT(AT(1)**2+AT(2)**2+AT(3)**2+AT(4)**2)
DO 75 J=1,4
UAT(J) = AT(J)*ORMAL
75 CONTINUE
RETURN
END

```

SUBROUTINE REACTA(XET,YET,ZET,D)

```

C-----
C
COMMON/AEROUT/TIDLEX ,TMILX ,TABX
1 ,FLODMX ,FLODSX
2 ,FMMINX ,FMMAXX ,RECANX
3 ,ALPHAX
4 ,CDX ,SBCDX
COMMON/ATTTAR/ANGETA,ANGETT,ANGXIA,ANGXIT,CRGR(3,3),DLOSSA,
1 DLOSST,DRGR(3,3),FAZMSA,FAZMST,FELVSA,FELVST,FLOSSA,
2 FLOSST,PSUBSA,PSUBST,RANGE,RRATE,XAINT,XTINA,YAINT,
3 YTINA,ZAINT,ZTINA
COMMON/CELSTA/AN1A,AN2A,AN3A,DTPRA,GLEVLA,ICMNA,ICOMDA,ICRECA,
1 ISTRGA,JVALUA,LCELLA(40),LVALA(40),LWEGHA(40),
2 NCELLA,NCLSTA,NOWPRA,NTILTA,NTRYA,ROTTA,ROTA,TIMEPA
COMMON/ENERGY/SPENYA,SPENYT
COMMON/EXTPOA/TIMEXA,DELTA,XZEROT,YZEROT,ZZEROT,XMIN1T,YMIN1T,
2 ZMIN1T,XMIN2T,YMIN2T,ZMIN2T,XEXPT,YEXPT,ZEXPT,
2 XDEXPT,YDEXPT,ZDEXPT,VELEXT,DRCEXT(3,3)
COMMON/HASSLE/IABLEA ,IABLET
COMMON/ K4/DT,TBEGN,TIME,PI,PIDV2,PIDV4,TWOPI,DEGRD,RADDG,G
1 ,VAR(20),IVAR(20),TEND
COMMON/PREDIC/CNEW(3,3),VXNEW,VYNEW,VZNEW,XXNEW,YYNEW,ZZNEW
COMMON/TRIALA/AN1TRA(10),AN2TRA(10),AN3TRA(10),CPRA(3,3,10),
1 DLOSRA(10),DRAGTA(10),DRGPRA(10),DXEPRA(10),
2 DYEPR(10),DZEPRA(10),FLDTRA(10),FLOSRA(10),
3 ICNPRA(10),ICTRYA(10),ISTPRA(20,10),ISTRYA(10),
4 IVALTA,IVPRA(10),KNEWA,KROTTA,PSTRA(10),PXEPRA(10),
5 PYEPRA(10),PZEPRA(10),RNGPRA(10),ROTTCA,ROTTN2A,
6 ROTRYA(10),THTRA(10),TPOTRA(10),XEXT,YEXT,ZEXT
COMMON/VARBLA/ACCXEA,ACCYEA,ACCZEA,ALFAA,CBARA(3,3),CBA(3,3),CDA,
1 CLALFA,CLA,COPHIA,COPSIA,COTHTA ,CSA,C(3,3),DRAGA,
2 FFLOSA,FLIFTA,FLODMA,FMACHA,FMMAXA,FMMINA,INIZA,
3 PHIBRA,PHIA,PMAXA,PPDTA,PA,PSIBRA,PSIA,QMAXA,QPDTA,

```

```

4          QA,RECANA,RHOA,RMAXA,RPDTA,RA,SIPHIA,SIPSIA,SITHTA ,
5          SPECEA,SA,TABA,THETAA,THETBA,TIDLEA,TMILA,TPOSA,
6          TRSTA,UA,VELA,VHORA,VA,WEITA,WA,XEDOTA,XEA,YEDOTA,
7          YEA,ZEDOTA,ZEA,FLODSA

```

```

      DIMENSION D(3,3)
      DIMENSION CBRTR(3,3)
      IF(ICRECA .LE. 0) GO TO 10

```

```

C
C ***  INITIALIZATION
C
C *****

```

```

      DELTA = DTPRA
      XMIN2T=0.
      YMIN2T=0.
      ZMIN2T=0.
      XMIN1T=0.
      YMIN1T=0.
      ZMIN1T=0.
      XZEROT=0.
      YZEROT=0.
      ZZEROT=0.
      ITPRE=1
      STORET=DELTA
      TIMCN=0.
      NOWPRA=0
      DTPRX = DTPRA*RADIFY(5HDTPRA,1.0,0.8,1.2)
      GO TO 999

```

```

C
C ***  END OF INITIALIZATION

```

```

10 CONTINUE
      STORET=STORET+DT
      IF(STORET.GE.DELTA) GO TO 15
      GO TO 185
15 STORET=0.
      XMIN2T=XMIN1T
      YMIN2T=YMIN1T
      ZMIN2T=ZMIN1T
      XMIN1T=XZEROT
      YMIN1T=YZEROT
      ZMIN1T=ZZEROT
      XZEROT=XET
      YZEROT=YET
      ZZEROT=ZET
      TLSAVE= TIME
185 CONTINUE
      IF(ITPRE.LE.0) GO TO 190
      IF(TIME.LT.TBEGN+DELTA*2.0)GO TO 999
      ITPRE=0
      GO TO 200
190 CONTINUE
      NOWPRA=0
      TIMCN=TIMCN+DT
      IF(TIMCN.LT.DTPRX)GO TO 999
200 TIMCN=0.00001

```



```

NOWPRA=1
CALL STATEA
DTPRX = DTPRA*RADIFY(5HDTPRA,1.0,0.8,1.2)
TIMEX = TIMEPA*RADIFY(6HTIMEPA,1.0,0.8,1.2)
IF (ABS (RANGE+RRATE*TIMEX) .LT. ABS (RRATE*DTPRX) *0.5)
1   TIMEX = DTPRX
   TIMEXA= TIMEX+TIME-TLSAVE

C
CALL EXTRA(D)
-----

C
C
XEXT=XEXPT
YEXT=YEXPT
ZEXT=ZEXPT
DXEXT=XDEXPT
DYEXT=YDEXPT
DZEXT=ZDEXPT
VVELT=VELEXT
DO 650 I=1,3
DO 650 J=1,3
650 DRGR(I,J)=DRCEXT(I,J)

C
C *** DETERMINE SUITABLE TRIAL COMMANDS
C *****
C
VDUM = 1. / VELA
VSPS = YEDOTA/VHORA
VCPS = XEDOTA / VHORA
VSTH = - ZEDOTA * VDUM
VCTH = VHORA * VDUM
PSIBRA=ATAN2(YEDOTA,XEDOTA)
THETBA=ATAN(-ZEDOTA/VHORA)
SIPSIA=SIN(PSIBRA)
COPSIA=COS(PSIBRA)
SITHTA =SIN(THETBA)
COTHTA =COS(THETBA)
PHIBRA=ATAN2((AN3A/COTHTA ),(AN2A*COPSIA-AN1A*SIPSIA))
SIPHIA=SIN(PHIBRA)
COPHIA=COS(PHIBRA)
CALL CMTRX(PSIBRA,THETBA,PHIBRA,CBA)
CALL TRYNXA

C
C *** CALCULATE THRUST, DRAG AND P SUB S FOR ALL TRIAL MANEUVERS
C
QBARS=VELA*VELA*RHOA*SA/2.
IABL = IABLEA
DO 2700 I=1,NTRYA
IF(TPOTRA(I).EQ.0.) THTRA(I)=TIDLEA*2.
IF(TPOTRA(I).EQ.1.) THTRA(I)=TMILA*2.
IF(TPOTRA(I).EQ.2.) THTRA(I)=TABA*2.
IABLEA= TPOTRA(I)
FLIFTR=FLDTRA(I)*FLODMA*WEITA
CLTR=FLIFTR/QBARS
CLTRO = CLTR

```

```

CALL AERF4(FMACHA,CLTR,4)
FLDTRA(I)= FLDTRA(I)/CLTRO*CLTR
ALPH = ALPHAX
DRAGTA (I)=QBARS*CDX
PSTRA(I)=(THTRA(I)*COS(ALPH)-DRAGTA (I))*VELA/WEITA
2700 CONTINUE
IABLEA= IABL
IF(IVAR(1).GT.0) GO TO 50251
AA4 = ROTSA * DEGRD
WRITE(46,50210) KRO TSA, NTRYA, AA4
50210 FORMAT ( '0 TRYNXA RETURNS KRO TSA, NTRYA, ROTSA = '
1 2I5, F10.3 // ' I ISTRYA ICTRYA G S'
2 ' ROTRYA AN1TRA AN2TRA AN3TRA P SUB S'/)
DO 50250 I = 1, NTRYA
AA1=ROTRYA(I)*DEGRD
GLD = FLDTRA(I) * FLODMA
WRITE(46,50215)I, ISTRYA(I), ICTRYA(I), GLD, AA1, AN1TRA(I), AN2TRA(I),
1 AN3TRA(I), PSTRA(I)
50215 FORMAT( 3I10, F10.4, 5F10.3 )
50250 CONTINUE
50251 CONTINUE
C
C *** OBTAIN PREDICTED SITUATION FOR TRIALS 1 THROUGH NTRYT
C *** *****
C
DO 600 I=1,NTRYA
IF(ISTRYA(I)) 400,380,400
380 CONTINUE
PHIBR=ATAN2((AN3TRA(I)/COTHTA ), (AN2TRA(I)*COPSIA-AN1TRA(I)*
1 SIPSIA))
CALL CMTRX(PSIBRA,THETBA,PHIBR,CBRTR)
COMPGN =CBRTR(3,3)
ACCNR = FLDTRA(I) * FLODMA - COMPGN
CALL PRETNW (AN1TRA(I),AN2TRA(I),AN3TRA(I),CBRTR,ACCNR,TIMEX,
1 XEDOTA,YEDOTA,ZEDOTA,XEA,YEA,ZEA,VELA)
GO TO 450
400 CALL PRESR(TIMEX, ISTRYA(I),VELA,XEA,YEA,ZEA,VSTH,VCTH,VSPS,VCPS)
450 PXEPRA(I) = XXNEW
PYEPRA(I) = YYNEW
PZEPRA(I) = ZZNEW
DXEPRA(I) = VXNEW
DYEPRA(I) = VYNEW
DZEPRA(I)=VZNEW
DO 475 II=1,3
DO 475 J=1,3
475 CPRA(II,J,I) = CNEW(II,J)
600 CONTINUE
C
C *** DETERMINE VALUE FOR EACH SITUATION RESULTING FROM THE NTRYA
C *** TRIAL COMMANDS
C
IVALTA= 0
SPEN = SPENYT
SPENYT= SPENYA+(SPEN-SPENYA)*RADIFY(5HENOPA,1.0,0.9,1.1)

```

```

DO 850 K = 1, NTRYA
XEPRA = PXEPRA(K)
YEPRA = PYEPRA(K)
ZEPRA = PZEPRA(K)
PDXEA = DXEPRA(K)
PDYEA = DYEPRA(K)
PDZEA = DZEPRA(K)
VVELA = SQRT ( PDXEA * PDXEA + PDYEA * PDYEA + PDZEA * PDZEA )
DO 810 II = 1,3
DO 810 JJ= 1,3
810 CRGR(II,JJ)=CPRA(II,JJ,K)
DUMMY=1.
CALL RELGN(XEPRA,YEPRA,ZEPRA,PDXEA,PDYEA,PDZEA,XEXT,YEXT,ZEXT,
1 DXEXT,DYEXT,DZEXT,FFLOSA,DUMMY,TIMEXA,VVELA,VVELT)
RANGE = RANGE*RADIFY(5HRANGA,1.0,0.9,1.1)
RRATE = RRATE*RADIFY(5HRANGA,1.0,0.9,1.1)
FLOSSA= FLOSSA*RADIFY(4HLOSA,1.0,0.9,1.1)
DLOSSA= DLOSSA*RADIFY(4HLOSA,1.0,0.9,1.1)
FLOSST= FLOSST*RADIFY(4HLOSA,1.0,0.8,1.2)
FLOSSA= AMIN1(PI,FLOSSA)
FLOSST= AMIN1(PI,FLOSST)
PSUBSA=PSTRA(K)
CALL STATEA
DO 830 L = 1, NCLSTA
830 ISTPRA (L,K) = LCELLA( L )
RNGPRA(K) = RANGE
DRGPRA(K) = RRATE
ICNPRA(K) = NCELLA
IVPRA(K) = JVALUA
FLOSRA(K) = FLOSSA
DLOSRA(K) = DLOSSA
IVALTA= MAX0(IVALTA,JVALUA)
850 CONTINUE
SPENYT= SPEN
C
C *** SELECT THE MOST PROMISING OF THE TRIAL COMMANDS
C
IF(IVALTA.NE.IVPRA(KRO TSA))GO TO 120
KNEWA = KRO TSA
GO TO 300
120 CONTINUE
TESROT= PI
KNEWA = 1
DO 130 K=1,NTRYA
IF(IVALTA.NE.IVPRA(K))GO TO 130
TEROT = ABS(RO TRYA(K)-RO TSA)
IF(TEROT.GT.PI)TEROT= TWOPI-TEROT
IF(TEROT.GE.TESROT)GO TO 130
TESROT= TEROT
KNEWA = K
130 CONTINUE
300 CONTINUE
ICM NWA = 1
ISTRGA = ISTRYA( KNEWA )

```



```

C
C TARGET AHEAD / ATTACKER BEHIND
C
  IF( XTINA .GE. ( 0. ) ) LCELLA(1) = 1
  IF( XAINT .LE. ( 0. ) ) LCELLA(2) = 1
C
  VISIBILITY
  VISIBILITY DEFFINED BY A PLANE PASSING THROUGH AIRCRAFT WING,
  PITCHED DOWN THIRTY DEGREES
C
  DEFFINE Z VISIBILITY AXIS POSITIVE UPWARDS
C
  ZVISA = .5 * XTINA - .86603 * ZTINA
  IF( ZVISA . GE. ( 0. ) ) LCELLA(3) = 1
C
  ZVIST = .5 * XAINT - .86606 * ZAINT
  IF( ZVIST .LE. ( 0. ) ) LCELLA( 4) = 1
C
  VOLUME BEHIND AIRCRAFT
  VOLUME IS A THIRTY DEGREE CONE FOR 3000 FEET BEHIND
  VOLUME IS A CONE CHANGING TO 45 DEGREE AT 3000 FEET BACK
  VOLUME IS LIMITED TO 5000 FEET BEHIND
C
  RSQ1 = YAINT * YAINT + ZAINT * ZAINT
  IF(LCELLA(3).EQ.0)GO TO 431
  IF(XAINT)405,431,431
405 CONTINUE
  CALL FIRCON(1,RANGE,0.0,0.0,FAT)
  IF(FAT.EQ.0.0)GO TO 431
C
  RSQ1 IS SQUARE OF ( DISTANCE FROM ATTACKER CG TO TARGET BODY X AXIS
C
410 IF(RANGE- 3000.)411,411,415
411 IF( RSQ1 - .57735 * XAINT * .57735 * XAINT) 430,430,431
C
  XSHIFT IS TARGET BODY X AZIS LOCATION 45 DEGREE CONE APEX
  XPRI IS DISTANCE ALONG TARGET BODY X AXIS, FROM 45 DEGREE CONE
  POINT ON X AXIS CLOSEST TO ATTACKER
C
415 XSHIFT = - 3000. * .36603
  XPRI = XAINT - XSHIFT
  IF ( RSQ1 - XPRI * XPRI ) 430,430,431
430 LCELLA( 5 ) = 1
431 CONTINUE
C
  RSQ2 = YTINA * YTINA + ZTINA * ZTINA
  IF(LCELLA(4).EQ.1)GO TO 531
  IF(XTINA)505,531,531
505 CONTINUE
  CALL FIRCON(KNOWOP,RANGE,0.0,0.0,FAT)
  IF(FAT.EQ.0.0)GO TO 531
510 IF(RANGE-3000.)511,511,515
511 IF( RSQ2 - .57735 * XTINA * .57735 * XTINA ) 530,530,531
515 XSHIFA = -3000. * .36603

```

```

XPRIM = XTINA - XSHIFA
IF ( RSQ2 - XPRIM * XPRIM ) 530,530,531
531 LCELLA( 6) = 1
530 CONTINUE
C
C ABILITY TO ATTACK
C OPPONENT WITHIN 30 DEGREES OF VELOCITY VECTOR
C
IF(LCELLA(2).EQ.0)GO TO 481
CALL FIRCON(1,RANGE,FLOSSA,0.0,FAT)
LCELLA(7)= FAT
481 CONTINUE
C
FAT = 0.0
IF(LCELLA(1).EQ.1)GO TO 581
CALL FIRCON(KNOWOP,RANGE,FLOSST,0.0,FAT)
581 CONTINUE
LCELLA(8)= 1.0-FAT
C
C CLOSING
C
REFDIS= RRATE*DTPRA
IF(ABS(REFDIS).LT.0.1)REFDIS= 0.1
IF(LCELLA(5).EQ.0.AND.LCELLA(7).EQ.0)GO TO 1070
DECTO1=(RANG1-RANGE)/REFDIS
DECTO2=(RANG2-RANGE)/REFDIS
IF(DECTO1*DECTO2.GE.0.0)GO TO 920
C WITHIN RANG1 - RANG2
IF(ABS((RANG2-RANG1)/REFDIS).LT.5.0)GO TO 950
C LONG RESIDENCE TIME
LCELLA(10)= 1
GO TO 950
920 CONTINUE
C APPROACH SPEED
DECTOM= AMAX1(DECTO1,DECTO2)
IF(DECTOM.LE.0.0)GO TO 1070
IF(DECTOM.GT.10.0)GO TO 1070
IF(ABS((RANG2-RANG1)/REFDIS).LT.2.0)GO TO 1070
950 CONTINUE
LCELLA(9)= 1
1070 CONTINUE
C
C SEPARATING
C
LCELLA(11)= 1
LCELLA(12)= 1
IF(LCELLA(6).NE.0.AND.LCELLA(8).NE.0)GO TO 1270
DECTO1=(RANG1-RANGE)/REFDIS
DECTO2=(RANG2-RANGE)/REFDIS
IF(DECTO1*DECTO2.GE.0.0)GO TO 1120
C WITHIN RANG1 - RANG2
IF(ABS((RANG2-RANG1)/REFDIS).LT.5.0)GO TO 1150
C LONG RESIDENCE TIME
LCELLA(12)= 0

```

```

GO TO 1150
1120 CONTINUE
C SEPARATION SPEED
DECTOM= AMAX1(DECTO1,DECTO2)
IF(DECTOM.LE.0.0)GO TO 1270
IF(DECTOM.GT.10.0)GO TO 1270
IF(ABS((RANG2-RANG1)/REFDIS).LT.2.0)GO TO 1270
1150 CONTINUE
LCELLA(11)= 0
1270 CONTINUE
C
C LINE OF SIGHT, LOS
C
IF ( FLOSSA .LE. FLOSX ) LCELLA ( 13 ) = 1
C
C RATE OF CHANGE OF ANGLE LOS
IF(DLOSSA.GT.(DLOSX-FLOSSA)/DTPRA*0.1)GO TO 1450
IF(DLOSSA.LT.(DLOSX+FLOSSA)/DTPRA*(-0.5))GO TO 1450
IF(XAINT.GT.0.0.AND.RRATE.LT.-CLOS)GO TO 1450
LCELLA(14)= 1
1450 CONTINUE
IF((PSUBSA+100.0)*DTPRA*5.0+SPENYA-SPENYT.GT.0.0)LCELLA(15)= 1
C
C CALCULATE VALUE OF EACH INDIVIDUAL STATE, VECTOR LVALA
C
DO 10010 I =1, NCLSTA
10010 LVALA(I) = LCELLA(I) * LWEGHA( I)
C
C CALCULATE TOTAL CELL STRUCTURE VALUE, JVALUA
C AND CELL NUMBER, NCELLA
C
JVALUA = 0
NCELLA = 0
DO 10050 I = 1, NCLSTA
JVALUA = JVALUA + LVALA(I)
NCELLA = NCELLA + LCELLA(I)*2**(I-1)
10050 CONTINUE
C
C PASS ELEVATION AND AZIMUTH TO LOS FROM ATTACKER X BODY AXIS , AS
C FELVSA, FAZMSA, TO ROUTINE PRCELA
C
FELVSA = ANGETA
FAZMSA = ANGZIA
C
RETURN
END

SUBROUTINE THRTLA
C-----
C
COMMON/ATTTAR/ANGETA,ANGETT,ANGZIA,ANGXIT,CRGR(3,3),DLOSSA,
1 DLOSST,DRGR(3,3),FAZMSA,FAZMST,FELVSA,FELVST,FLOSSA,
2 FLOSST,PSUBSA,PSUBST,RANGE,RRATE,XAINT,XTINA,YAINT,

```



```

3          YTINA,ZAINT,ZTINA
COMMON/   K4/DT,TBEGN,TIME,PI,PIDV2,PIDV4,TWOPI,DEGRD,RADDG,G
1          ,VAR(20),IVAR(20),TEND
COMMON/VARBLA/ACCXEA,ACCYEA,ACCZEA,ALFAA,CBARA(3,3),CBA(3,3),CDA,
1          CLALFA,CLA,COPHIA,COPSIA,COTHTA,CSA,C(3,3),DRAGA,
2          FFLOSA,FLIFTA,FLODMA,FMACHA,FMMAXA,FMMINA,INIZA,
3          PHIBRA,PHIA,PMAXA,PPDTA,PA,PSIBRA,PSIA,QMAXA,QPDTA,
4          QA,RECANA,RHOA,RMAXA,RPDTA,RA,SIPHIA,SIPSIA,SITHTA,
5          SPECEA,SA,TABA,THETAA,THETBA,TIDLEA,TMILA,TPOSA,
6          TRSTA,UA,VELA,VHORA,VA,WEITA,WA,XEDOTA,XEA,YEDOTA,
7          YEA,ZEDOTA,ZEA,FLODSA
COMMON/TALLY/OFTIMA,OFTIMT,GUNTMA,GUNTMT,ANGOFA,ANGOFT,
1          DEVAA,DEVAT

```

C

```

TEN = 10./DEGRD
SIXTY = 60./DEGRD
IF(XTINA.LT.0.0)GO TO 4000
IF (ABS (ANGOFA) .GE.SIXTY) RETURN
IF (ABS (DEVAA) .GE.SIXTY) RETURN
IF (RANGE.GE.5000.0) RETURN
IF (RANGE.LT.3000.0) GO TO 7000
IF (RRATE.GE.-300.0) RETURN
IF (THETBA.LT.SIXTY) GO TO 7000
RETURN
4000 CONTINUE
IF (ABS (FLOSST) .LE.TEN) RETURN
IF (ABS (FLOSST) .GE.SIXTY) RETURN
IF (RANGE.GE.1500.0) RETURN
IF (RRATE.GE.-300.0) RETURN
7000 CONTINUE
TPOSA = 0.0
RETURN
END

```

SUBROUTINE TRYNXA

C-----
C

```

COMMON/CELSTA/AN1A,AN2A,AN3A,DTPRA,GLEVL,ICMNA,ICOMDA,ICRECA,
1          ISTRGA,JVALUA,LCELLA(40),LVALA(40),LWEGHA(40),
2          NCELLA,NCLSTA,NOWPRA,NTILTA,NTRYA,ROSTA,ROTA,TIMEPA
COMMON/   K4/DT,TBEGN,TIME,PI,PIDV2,PIDV4,TWOPI,DEGRD,RADDG,G
1          ,VAR(20),IVAR(20),TEND
COMMON/PREDIC/CNEW(3,3),VXNEW,VYNEW,VZNEW,XXNEW,YYNEW,ZZNEW
COMMON/TRIALA/AN1TRA(10),AN2TRA(10),AN3TRA(10),CPRA(3,3,10),
1          DLOSRA(10),DRAGTA(10),DRGPRA(10),DXEPRA(10),
2          DYEPR(10),DZEPRA(10),FLDTRA(10),FLOSRA(10),
3          ICNTRA(10),ICTRYA(10),ISTPRA(20,10),ISTRYA(10),
4          IVALTA,IVPRA(10),KNEWA,KROSTA,PSTRA(10),PXEPR(10),
5          PYEPRA(10),PZEPRA(10),RNGPRA(10),ROTNCA,ROTN2A,
6          ROTRYA(10),THTRA(10),TPOTRA(10),XEXT,YEXT,ZEXT
COMMON/VARBLA/ACCXEA,ACCYEA,ACCZEA,ALFAA,CBARA(3,3),CBA(3,3),CDA,
1          CLALFA,CLA,COPHIA,COPSIA,COTHTA,CSA,C(3,3),DRAGA,
2          FFLOSA,FLIFTA,FLODMA,FMACHA,FMMAXA,FMMINA,INIZA,

```

3 PHIBRA, PHIA, PMAXA, PPDTA, PA, PSIBRA, PSIA, QMAXA, QPDTA,
 4 QA, RECANA, RHOA, RMAXA, RPDTA, RA, SIPHIA, SIPSIA, SITHTA ,
 5 SPECEA, SA, TABA, THETAA, THETBA, TIDLEA, TMILA, TPOSA,
 6 TRSTA, UA, VELA, VHORA, VA, WEITA, WA, XEDOTA, XEA, YEDOTA,
 7 YEA, ZEDOTA, ZEA, FLODSA

DIMENSION CMPL(3,3)
 DATA EXLSM/1.57080/
 DATA EIGHTY/1.39626/
 DATA TAB155/1.0/, TAB156/1.1/, TAB157/0.1/
 TRFACT=2.
 FLODES = VELA*VELA*RHOA*SA*0.5
 FLODES= CLADES(1, FMACHA)/FLODMA*FLODES/WEITA
 FLODES= AMIN1(0.95, FLODES)
 FLOSTR=(VHORA/VELA)/FLODMA
 DO 7 I=1,10
 ISTRYA(I)= 0
 ICTRYA(I)= 0
 ROTRYA(I)= 0.0
 FLDTRA(I)= TAB155
 TPOTRA(I)= 2.0

7 CONTINUE

TAXE=XEXT-XEA
 TAYE=YEXT-YEA
 TAZE=ZEXT-ZEA
 DZ=(XEDOTA*ZEDOTA*TAXE+YEDOTA*ZEDOTA*TAYE-(VHORA**2)*TAZE)/VELA
 DY=-YEDOTA*TAXE+XEDOTA*TAYE
 IF(DY.EQ.0.) GO TO 10
 ROTSA=ATAN2(DY, DZ)
 GO TO 99

10 IF(DZ.GE.0.) ROTSA=0.
 IF(DZ.LT.0.) ROTSA=PI

99 CONTINUE

DIVEAN =-THETBA
 IF(DIVEAN .LT.0.) GO TO 1002
 IF(DIVEAN .GE.RECANA) GO TO 500
 IF(DIVEAN .LT.(0.8*RECANA)) GO TO 1002

C
 C *** DIVEANGLE IS BETWEEN 80 AND 100 PERCENT OF RECOVERY ANGLE
 C

IF(RECANA.GT.1.2)GO TO 1002
 NTRYA=3
 KRO TSA=1

C
 C *** FIRST TRIAL IS PULLUP IN VERTICAL PLANE
 C

C *** TRIAL 2 AN 3 ARE PULLUPS IN ADJACENT PLANES
 C

ICTRYA(2)=1
 ROTRYA(2)=ROTNCA

C
 ICTRYA(3)=-1
 ROTRYA(3)=-ROTNCA
 GO TO 600

500 CONTINUE

```

C
C *** ONLY ONE MANEUVER ALLOWED   STRAIGHT PULLUP
C
      NTRYA=1
      KRO TSA=1
      GO TO 600
C
C   CHECK FOR LOW LOAD FACTOR
C
1002 CONTINUE
      IF(FLODMA.GT.1.5)GO TO 1001
      IF(THETBA.LE.0.0)GO TO 3500
      IF(THETBA.GT.EIGHTY)GO TO 3450
      IF(FLODMA.GE.1.0)GO TO 3600
3330 CONTINUE
      NTRYA = 2
      IF(ABS(RO TSA).LE.PIDV2)KRO TSA= 1
      IF(ABS(RO TSA).GT.PIDV2)KRO TSA= 2
      ROTRYA(2)= PI
      ICTRYA(2)= PI/RO TNCA+0.001
      FLDTRA(1)=0.001
      FLDTRA(2)=0.001
      GO TO 600
3450 CONTINUE
      NTRYA = 1
      CALL GETRXN(RO TSA,RO TNCA,RO TRYA(1),ICTRYA(1))
      GO TO 600
3500 CONTINUE
3600 CONTINUE
      NTRYA = 6
      DO 3605 I=1,NTRYA
      FLDTRA(I)= FLODES
3605 CONTINUE
      KRO TSA= 3
      IF(ABS(RO TSA).GT.PIDV2)KRO TSA= 5
      IF(RO TSA.LT.0.0)KRO TSA= KRO TSA+1
      ROTRYA(2)= PI
      ROTRYA(3)= RO TNCA
      ROTRYA(4)=-RO TNCA
      ROTRYA(5)= PI-RO TNCA
      ROTRYA(6)= RO TNCA-PI
      ICTRYA(2)= PI/RO TNCA
      ICTRYA(3)= 1
      ICTRYA(4)=-1
      ICTRYA(5)= ICTRYA(2)-1
      ICTRYA(6)= 1-ICTRYA(2)
      GO TO 600
C
C *** 1001 FOR 'NORMAL'  CONDITIONS
C *****
C
1001 CONTINUE
      NTRYA = 4
      KRO TSA = 1

```

```

NTRY = 4
ISTRYA(1)= ISTRGA
ROTRYA(1)=PHIBRA
IF(ISTRGA) 2010,100,200
100 CONTINUE
FLDTRA(1)= TAB155+(GLEVLA-TAB155)*RADIFY(6HGLEVLA,0.75,0.5,1.0)
ICTRYA(1)= PHIBRA/ROTNCA
ROTPM=PHIBRA
CALL GETRXN(ROTPM,ROTNCA,ROTR2,ICTR2)
ROTRYA(2) = ROTR2
ICTRYA(2) = ICTR2
IF(ROTR2-ROTPM)130,123,125
123 CONTINUE
IF(ROTPM)130,130,125
125 ROTRYA(3) = ROTR2 - ROTNCA
ICTRYA(3) = ICTR2 - 1
GO TO 140
130 ROTRYA(3) = ROTR2 + ROTNCA
ICTRYA(3) = ICTR2 +1
140 CONTINUE
ISTRYA(4)= 1
IF( ABS( ROTSA ) .LE. PIDV2) GO TO 150
ISTRYA( 4 ) = -1
ROTRYA(4) = PI
150 CONTINUE
C
C IF NONE OF THE THREE MANEUVER PLANES IN TRIALS 1, 2, OR 3 IS THE
C PLANE CLOSEST TO THE OPPONENT, ADD A FIFTH TRIAL MANEUVER
C
K = 1
TESROT= PI
DO 250 I=1,3
TEROT = ABS(ROTRYA(I)-RO TSA)
IF(TEROT.GT.PI)TEROT= TWOPI-TEROT
IF(TEROT.GT.TESROT)GO TO 250
K = I
TESROT= TEROT
250 CONTINUE
IF(TESROT.LE.ROTN2A)GO TO 275
C NOT INCLUDED
NTRYA = 5
NTRY = 5
KRO TSA= 5
CALL GETRXN(RO TSA,RO TNCA,RO TRY A(5) ,IC TRY A(5))
GO TO 2000
C INCLUDED
275 CONTINUE
KRO TSA= K
IF(K.EQ.1.AND.ABS(FLODES-FLDTRA(1)).LE.0.05)GO TO 2000
NTRYA = 5
IC TRY A(5)= IC TRY A(K)
RO TRY A(5)= RO TRY A(K)
FLDTRA(5)= FLODES
GO TO 2000

```

```

200 CONTINUE
   FLDTRA(1) = FLOSTR
   ISTRYA(2) = -1
   ROTRYA(2) = PI
   GO TO 2020
2010 CONTINUE
   FLDTRA(1) = -FLOSTR
   ISTRYA(2) = 1
2020 KRO TSA = 3
   CALL GETRXN(RO TSA, ROTNCA, ROTR3, ICTR3)
   ROTRYA(3) = ROTR3
   ICTRYA(3) = ICTR3
   IF( ROTR3 - RO TSA ) 2110, 2105, 2105
2105 ROTRYA(4) = ROTR3 - ROTNCA
   ICTRYA(4) = ICTR3 - 1
   GO TO 2000
2110 ROTRYA(4) = ROTR3 + ROTNCA
   ICTRYA(4) = ICTR3 + 1
2000 CONTINUE
   IF ( FFLOSA .GT. EXLSM ) GO TO 2400
C *** CALCULATE REQUIRED G-LEVEL FOR TURN INTO OPPONENTS EXTRAPOLATED
C POSITION
   CALL CMTRX(PSIBRA, THETBA, RO TSA, CMPL)
   DIST2 = TAXE**2 + TAYE**2 + TAZE**2
   ZMT = TAXE*CMPL(3,1) + TAYE*CMPL(3,2) + TAZE*CMPL(3,3)
   RADIS = DIST2 / (2.*ZMT)
   GL2 = (ABS((VELA**2)/RADIS)/G) + CMPL(3,3)
   GL3 = ABS(CMPL(2,3))
   GLEVRA = SQRT(GL2**2 + GL3**2) / FLODMA
   GLEVRA = GLEVRA * TAB156 + TAB157
   IF ( GLEVRA .GT. 1.0 ) GO TO 2400
   NTRYA = NTRYA + 1
   FLDTRA( NTRYA ) = GLEVRA
   CALL GETRXN(RO TSA, ROTNCA, ROTRYA(NTRYA), ICTRYA(NTRYA))
   KRO TSA = NTRYA
   GO TO 2500
2400 CONTINUE
   IF(JVALUA.GT.6)GO TO 2500
C DEFENSIVE MANEUVER
   NTRYA = NTRYA + 1
   IF(JVALUA.LE.5)KRO TSA = NTRYA
   ROTPM = PIDV2 - ROTNCA
   IF(ABS(RO TSA) .GT. PIDV2) ROTPM = -ROTPM
   ROTPM = ABS(RO TSA) + ROTPM
   IF(RO TSA.LT.0.0) ROTPM = -ROTPM
   CALL GETRXN(ROTPM, ROTNCA, ROTRYA(NTRYA), ICTRYA(NTRYA))
   ISTRYA(NTRYA) = 0
   FLDTRA(NTRYA) = FLODES
2500 CONTINUE
   DO 2555 I=2, NTRY
   IF(ISTRYA(I)) 2551, 2555, 2553
2551 FLDTRA(I) = -FLOSTR
   GO TO 2555
2553 FLDTRA(I) = FLOSTR

```

```

2555 CONTINUE
      DO 2560 I=1,NTRYA
2560 TPOTRA(I)=TTRACT
      600 CONTINUE
          DO 2600 I=1,NTRYA
              GLEVRA= FLDTRA(I)*FLODMA
              GLEV  = GLEVRA
              CALL HUBLO(1,1,DTPRA,GLEVRA)
              IF(GLEV.NE.GLEVRA)FLDTRA(I)= GLEVRA/FLODMA
              ROSS = ROTRYA(I)
              IF(ISTRYA(I).NE.0)GO TO 2580
              SROSS = SIN(ROSS)
              REDUC = ABS(SROSS/FLODMA*COTHTA /FLDTRA(I))
              IF(REDUC.LE.ABS(SROSS))GO TO 2580
              ROSS  = 0.0
              IF(ABS(ROTRYA(I)).LE.PIDV2)GO TO 2570
              ROSS  = PI
2570 CONTINUE
              ROTRYA(I)= ROSS
2580 CONTINUE
              CALL NORPLN(XEDOTA,YEDOTA,ZEDOTA,ROSS,AN1TRA(I),AN2TRA(I),
1              AN3TRA(I))
2600 CONTINUE
      RETURN
      END

```

FUNCTION CLADES (KWHICH,ACHNO)

```
C-----  
C  
C CLADES DEFINES DESIRABLE MANEUVER LIFT COEFFICIENT  
C KWHICH= 1 ATTACKER  
C KWHICH= 2 TARGET  
C ACHNO MACH NUMBER  
C CLADES= 0.5  
C RETURN  
C END
```

FUNCTION GG (A, TU)

```
C-----  
C INTERPOLATION AID FOR AERF4  
C DIMENSION A(2)  
C GG = A(1)+(A(2)-A(1))*TU  
C RETURN  
C END
```

FUNCTION GGG (A, GH, TV, TW)

```
C-----  
C INTERPOLATION AID FOR AERF4  
C DIMENSION A(2)  
C GGG = GH+(A(1)+(A(2)-A(1))*TV-GH)*TW  
C RETURN  
C END
```

FUNCTION GGGG (A, BB)

```
C-----  
C  
C INTERPOLATION AID FOR AERF4  
C DIMENSION A(2)  
C GGGG = (BB-A(1))/(A(2)-A(1))  
C RETURN  
C END
```

FUNCTION RADIFY (NAME, VALNOM, VALO, VALU)

```
C-----  
C  
C *** THIS IS A DUMMY SUBROUTINE RADIFY (NO RANDOMIZATION)  
C  
C RADIFY=VALNOM  
C RETURN  
C END
```

FUNCTION SBDEFA(ALT,ACHNO,KRAFT)

C
C
C
C
C
CALCULATION OF THE LARGEST PERMISSIBLE SPEED BRAKE DEFLECTION
FRACTION OF MAX. DEFLECTION

DIMENSION ACH(3),COEF1(4,2),COEF2(4,2,2),BRKALT(2)
DIMENSION ACHAX(2),DEFLAX(2)
DATA BRKALT(1)/30000.0/,BRKALT(2)/30000.0/
DATA (COEF1(I,1),I=1,4)/1.95,1.27,1.18,30.5/
DATA (COEF1(I,2),I=1,4)/1.95,1.27,1.18,30.5/
DATA (COEF2(I,1,1),I=1,4)/1.75E-5,1.30E-5,1.6E-5,3.33E-4/
DATA (COEF2(I,2,1),I=1,4)/4.00E-5,4.25E-5,4.35E-5,1.00E-3/
DATA (COEF2(I,1,2),I=1,4)/1.75E-5,1.30E-5,1.6E-5,3.33E-4/
DATA (COEF2(I,2,2),I=1,4)/4.00E-5,4.25E-5,4.35E-5,1.00E-3/
DATA ACHAX(1)/2.37/,ACHAX(2)/2.37/
DATA DEFLAX(1)/60.0/,DEFLAX(2)/60.0/

C
C
C
C
KRAFT = 1 - ATTACKER
KRAFT = 2 - TARGET

KR = KRAFT
ANGL = 0.0
IF(ACHNO.GT.ACHAX(KR))GO TO 700
XEE = ALT-BRKALT(KR)
KEE = 1
IF(XEE.GT.0.0)KEE= 2
DO 100 KREG=1,3
ACH(KREG) = COEF1(KREG,KR)+COEF2(KREG,KEE,KR)*XEE
IF(ACHNO.GE.ACH(KREG))IF(KREG-2)700,300,400
IF(KREG.EQ.1)BPANGL= COEF1(4,KR)+COEF2(4,KEE,KR)*XEE

100 CONTINUE
ANGL = DEFLAX(KR)
GO TO 700
300 CONTINUE
ANGL = (ACH(1)-ACHNO)/(ACH(1)-ACH(2))*BPANGL
GO TO 700
400 CONTINUE
ANGL = (ACH(2)-ACHNO)/(ACH(2)-ACH(3))*(DEFLAX(KR)-BPANGL)+BPANGL
700 CONTINUE
SBDEFA= ANGL/DEFLAX(KR)
RETURN
END

FUNCTION SLAPDF(ALT,ACHNO,KRAFT)

C
C
C
C
C
CALCULATION OF LARGEST PERMISSIBLE SLAT/FLAP DEFLECTION ANGLE
AS FRACTION OF FULL DEFLECTION

TRANSFER VECTOR ALT ALTITUDE
ACHNO MACH NUMBER

C
C
C

KRAFT

1 - ATTACKER
2 - TARGET

```
DEFL = 0.0
IF(ACHNO.GE.1.0)GO TO 700
ACHHI = 1.0
IF(ALT.GE.16500.0)GO TO 300
IF(ALT-13000.0)100,100,150
100 CONTINUE
ACHHI = 0.7200+1.384E-05*ALT
GO TO 200
150 CONTINUE
ACHHI = 0.5286+2.857E-05*ALT
200 CONTINUE
IF(ACHNO.GE.ACHHI)GO TO 700
300 CONTINUE
DEFL = 1.0
ACHLO = 0.95
IF(ALT.GE.32000.0)GO TO 500
IF(ALT-25000.0)400,400,450
400 CONTINUE
ACHLO = 0.5400+1.080E-05*ALT
GO TO 500
450 CONTINUE
ACHLO = 0.3100+2.000E-05*ALT
500 CONTINUE
IF(ACHNO.LE.ACHLO)GO TO 700
DEFL = (ACHHI-ACHNO)/(ACHHI-ACHLO)
700 CONTINUE
SLAPDF= DEFL
RETURN
END
```

APPENDIX B

LISTING OF THE FORTRAN PROGRAMS TO PREPARE
AERODYNAMIC TABLES FOR AIRCRAFT A

```

PROGRAM CLINVR
c
c *** this program reads the raw data fo cl sub alpha
c from table fl01
c it puts the entire table into array clatab and then
c inverts the function cl=f(alpha) into alpha=f(cl)
c for the minus five degree tail deflection data
c
DIMENSION CLATAB(10,19,5),CLRAW(950),
. XTAB(14),YTAB(14),
. xprime(15),yprime(15)
. ,tail(5),fmach(10)
data ytab/-12,-8,-4,0,4,8,12,16,20,24,28,32,36,40/
data xprime/-1.2,-1.0,-0.8,-0.6,-0.4,-0.2,0,0.2,0.4,0.6,
. 0.8,1.0,1.2,1.4,1.6/

data tail/-25.,-15.,-5.,5.,15./
data fmach /0.2,0.4,0.6,0.8,0.9,1.0,1.1,1.2,1.4,1.6/
c
open(11,file='clraw.dat',status='old')
open(12,file='alfcl.dat',status='new')
open(13,file='claprt',status='new')

do 10 m=1,135
    index=(m-1)*7
    read(11,*)(clraw(index+j),j=1,7)
10 continue
    read(11,*)(clraw(947-j),j=1,5)

do 100 k=1,5
    do 80 j=1,19
        do 70 i=1,10
            index=(k-1)*190+(j-1)*10+i
            clatab(i,j,k)=clraw(index)
70 continue
80 continue
100 continue

print 989
989 format(///)
do 120 k=3,3
do 110 i=1,10
    print 91,fmach(i),(clatab(i,j,k),j=1,19)
    write(13,91)fmach(i),(clatab(i,j,k),j=1,19)
91 format(f6.2,3x,19f6.2)
110 continue
120 continue

```

LISTING OF PROGRAM 'CLINVR'. (See Notes at end of listing).

```

C
C *** print entire table f101
C
    print 192
    write(13,192)
192 format(1h1,////)
    do 220 k=1,5
        print 93,tail(k)
        write(13,93)tail(k)
    93  format(//,' Tail Deflection = ', f6.1,' degrees',//)
        do 210 i=1,10
            print 91,fmach(i),(clatab(i,j,k),j=1,19)
            write(13,91)fmach(i),(clatab(i,j,k),j=1,19)
210 continue
220 continue

C
C *** OUTER DO-LOOP FOR 10 MACH NUMBERS
C
    DO 1000 K=1,10

C
C *** FILL ARRAY OF INDEPENDENT VARIABLE WITH CL VALUES
C
        DO 1010 KK=1,14
            XTAB(KK)=CLATAB(K,KK,3)
1010  CONTINUE
            IF(K.EQ.1) XTAB(14)=1.5
            IF(K.EQ.2) XTAB(14)=1.47

C
C *** INNER DO-LOOP FOR 15 VALUES OF cl (-1.2 ... 1.6)
C
        DO 1020 J=1,15
            CALL TLU(XPRIME(J),XTAB,YTAB,14,LX,YPRIME(J),IC)
            PRINT 97,J,XPRIME(J),YPRIME(J),IC
    97  FORMAT(I5,2F15.5,I5)
            WRITE(12,92)YPRIME(J)
    92  FORMAT(F12.6)
1020  CONTINUE
1000 CONTINUE

        stop
        end

```

LISTING OF PROGRAM CLINVR (Continued)

```

SUBROUTINE TLU (ARG,XTAB,YTAB,NX,LX,ANS,IC)
C-----
  DIMENSION XTAB(NX),YTAB(NX)
  IC=0
  IF(LX.LT.1)LX=1
  IF(LX.GT.NX-1)LX=NX-1
10  IF(ARG-XTAB(LX))30,40,15
15  IF(ARG-XTAB(LX+1))40,40,20
20  LX=LX+1
  IF(LX.GE.NX) GO TO 99
  GO TO 10
30  LX=LX-1
  IF(LX.LT.1) GO TO 98
  GO TO 10
40  CONTINUE
  FRACT=(ARG-XTAB(LX))/(XTAB(LX+1)-XTAB(LX))
  ANS=YTAB(LX)+(YTAB(LX+1)-YTAB(LX))*FRACT
  RETURN
99  ANS=YTAB(NX)

C  PRINT8
  8  FORMAT(' ARGUMENT EXCEEDS TABLE, ARG AND X-TABLE FOLLOW')
C  PRINT100,ARG,(XTAB(I),I=1,NX)
100 FORMAT(1X,8F9.3)
  IC=90
  RETURN
98  ANS=YTAB(1)
C  PRINT 9
  9  FORMAT(' ARGUMENT LESS THAN FIRST X-TABLE-VALUE')
C  PRINT 100,ARG,(XTAB(I),I=1,NX)
  IC=80
  RETURN
  END

```

LISTING OF PROGRAM 'CLINVR' (Concluded)

The input-file 'clraw.dat' contains the 135 lines (without header-line) of function F101 in F15AC1.DAT.

The program tabulates the C_L values as function of Mach Number, angle of attack, and horizontal tail deflection. It also inverts the function $C_L = f(\text{Alpha})$.

The subroutine TLU used here is a slight modification of the AML version of TLU. If the argument is smaller than the lowest value of the independent variable, TLU returns the lowest y-value and correspondingly when the upper limit of the table is exceeded.

PROGRAM CDINVR

```

C-----
  dimension cdtab(14,10),cdprim(9,10),
  .           xtab(9),ytab(9),xprime(19),yprime(19)
  data ytab/0.,.2,.4,.6,.8,1.,1.2,1.4,1.6/
  data xprime/0.02,0.04,0.06,0.08,0.10,0.20,0.30,0.40,
  .           0.50,0.60,0.70,0.80,0.90,1.0,1.2,1.4,1.6,
  .           1.80,2.0/
  lx=1
  open(11,file='cdtab.dat',status='old')
  OPEN(12,FILE='CLFCD.DAT')
  REWIND 12
  read(11,*)((cdtab(i,j),j=1,10),i=1,14)
C
C.... remove negative cl values from cd table
C
  do 10 i=1,10
  do 10 j=6,14
    jj=j-5
  10   cdprim(jj,i)=cdtab(j,i)
C
C ... big do-loop for 10 mach numbers
C
  do 1000 k=1,10
    do 100 kk=1,9
  100   xtab(kk)=cdprim(kk,k)
C
C ... small do loop for 19 cd values
C
  do 200 j=1,19
    call tlu(xprime(j),xtab,ytab,9,lx,yprime(j),ic)
    print 91,j,xprime(j),yprime(j),ic
  91   format(i5,2f8.5,i5)
    WRITE(12,92)YPRIME(J)
  92  FORMAT(F12.6)
  200  continue
  1000 continue
  stop
  end

```

Listing of Program 'CDINVR'

```

SUBROUTINE TLU (ARG,XTAB,YTAB,NX,LX,ANS,IC)
C-----
  DIMENSION XTAB(NX),YTAB(NX)
  ic=0
  IF(LX.LT.1)LX=1
  IF(LX.GT.NX-1)LX=NX-1
10 IF(ARG-XTAB(LX))30,40,15
15 IF(ARG-XTAB(LX+1))40,40,20
20 LX=LX+1
  IF(LX.GE.NX) GO TO 99
  GO TO 10
30 LX=LX-1
  IF(LX.LT.1) GO TO 98
  GO TO 10
40 CONTINUE
  FRACT=(ARG-XTAB(LX))/(XTAB(LX+1)-XTAB(LX))
  ANS=YTAB(LX)+(YTAB(LX+1)-YTAB(LX))*FRACT
  RETURN
99 ANS=YTAB(NX)

C PRINT8
  8 FORMAT(' ARGUMENT EXCEEDS TABLE, ARG AND X-TABLE FOLLOW')
C PRINT100,ARG,(XTAB(I),I=1,NX)
100 FORMAT(1X,8F9.3)
  IC=90
  RETURN
98 ANS=YTAB(1)
C PRINT 9
  9 FORMAT(' ARGUMENT LESS THAN FIRST X-TABLE-VALUE')
C PRINT 100,ARG,(XTAB(I),I=1,NX)
  IC=80
  RETURN
  END

```

Listing of program 'CDINVR' (Concluded)

This program inverts the function $C_D = f(C_L)$.

The input file 'cdtab.dat' contains the table $C_D = f(C_L)$ as it was generated by the program DRAG.FOR.

APPENDIX C

AERODYNAMICS AND PERFORMANCE DATA FOR
AIRCRAFT B AS USED IN THE AML PROGRAM

AERODYNAMIC AND ENGINE DATA

REFERENCE AREA 565.0 SQFT

MAXIMUM LOAD FACTOR(AFTERBURNER-ON) (G)

ALTITUDE	MACH NO	0.20	0.50	0.80	0.90	1.00	1.10	1.20	1.50	1.80	2.00	2.20	2.40
	0.		1.3	7.8	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0
15000.		0.7	4.4	9.5	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0
30000.		0.4	2.3	5.0	5.8	6.4	6.8	7.8	9.6	10.0	10.0	10.0	10.0
45000.		0.2	1.1	2.5	2.8	3.1	3.4	3.8	4.7	6.3	8.2	10.0	10.0
55000.		0.1	0.7	1.5	1.8	1.9	2.1	2.4	2.9	3.9	5.1	6.5	8.5

SUSTAINED LOAD FACTOR(AFT BURN-ON) (G)

ALTITUDE	MACH NO	0.20	0.50	0.80	0.90	1.00	1.10	1.20	1.50	1.80	2.00	2.20	2.40
	0.		1.3	4.6	7.8	7.6	5.9	2.9	0.0	0.0	0.0	0.0	0.0
15000.		0.7	2.8	4.7	4.9	4.5	4.4	4.4	2.6	0.0	0.0	0.0	0.0
30000.		0.4	1.5	2.6	2.7	2.7	2.8	3.0	3.6	3.6	3.0	1.0	0.0
45000.		0.2	0.7	1.3	1.3	1.3	1.4	1.5	1.8	2.0	2.1	1.9	0.5
55000.		0.1	0.4	0.7	0.8	0.8	0.8	0.9	1.1	1.1	1.2	1.1	0.1

IDLE THRUST TIDLE (LB)

MACH NO	ALTITUDE	0.	10000.	20000.	30000.	40000.	50000.	60000.
	0.2		388.	233.	126.	300.	520.	615.
0.4		260.	150.	76.	118.	270.	340.	340.
0.6		1014.	651.	403.	261.	137.	120.	140.
0.8		1049.	660.	400.	262.	145.	69.	50.
0.9		430.	229.	108.	74.	26.	-4.	40.
1.0		-693.	-547.	-412.	-263.	-185.	-131.	-15.
1.1		-2978.	-2129.	-1476.	-953.	-620.	-401.	-150.
1.2		-3501.	-2498.	-1731.	-1121.	-723.	-452.	-282.
1.4		-5359.	798.	1339.	-239.	-193.	-121.	-70.
1.6		-10000.	-1887.	914.	711.	120.	-73.	55.
1.8		-10000.	-8750.	-305.	40.	-57.	-41.	-3.
2.0		-10000.	-10000.	-1457.	-186.	-197.	-138.	-66.
2.2		-10000.	-10000.	-5000.	-1296.	-886.	-573.	-342.
2.4		-10000.	-10000.	-7200.	-4800.	-3750.	-2400.	-1600.

DATA SET FOR AIRCRAFT B

MILITARY THRUST THIL (LB)

MACH NO	ALTITUDE	0.	10000.	20000.	30000.	40000.	50000.	60000.
	0.2	10350.	7400.	5150.	3500.	1940.	1050.	650.
0.4	10350.	7500.	5200.	3450.	2050.	1170.	725.	
0.6	10650.	7800.	5450.	3650.	2300.	1300.	820.	
0.8	10650.	8350.	5950.	4050.	2550.	1500.	885.	
0.9	10300.	8250.	6100.	4150.	2650.	1550.	905.	
1.0	8750.	7850.	6150.	4250.	2700.	1600.	925.	
1.1	6300.	7550.	6200.	4450.	2800.	1650.	950.	
1.2	2350.	5500.	4850.	3900.	2450.	1400.	750.	
1.4	400.	2500.	3250.	3000.	2000.	1030.	420.	
1.6	0.	500.	1300.	1580.	1020.	200.	0.	
1.8	0.	0.	320.	750.	260.	0.	0.	
2.0	0.	0.	0.	200.	0.	0.	0.	
2.2	0.	0.	0.	0.	0.	0.	0.	
2.4	0.	0.	0.	0.	0.	0.	0.	

AFTERBURNER THRUST TAB (LB)

MACH NO	ALTITUDE	0.	10000.	20000.	30000.	40000.	50000.	60000.
	0.2	18400.	14100.	9830.	6250.	3600.	1940.	650.
0.4	19900.	15000.	10880.	7090.	4180.	2400.	910.	
0.6	22300.	16550.	12450.	8340.	5050.	3000.	1280.	
0.8	25000.	19780.	14700.	10200.	6250.	3700.	1730.	
0.9	25200.	21500.	16000.	11250.	7050.	4100.	2000.	
1.0	24900.	22800.	17380.	12450.	7900.	4550.	2300.	
1.1	24200.	24000.	18800.	13750.	8700.	5050.	2640.	
1.2	23190.	25000.	20150.	15080.	9700.	5550.	3000.	
1.4	17090.	24950.	22900.	18100.	11650.	6700.	3780.	
1.6	11500.	23500.	24680.	21000.	13940.	7950.	4620.	
1.8	8550.	21700.	25750.	23450.	16500.	9300.	5550.	
2.0	6730.	19500.	26200.	24950.	19000.	10650.	6600.	
2.2	5430.	17100.	25700.	25700.	20650.	12000.	7130.	
2.4	4440.	14700.	24000.	24500.	19050.	11350.	6700.	

DATA SET FOR AIRCRAFT B (Continued)

ANGLE OF ATTACK (WITH HI LIFT DEVICES)

MACH NO \ LIFT COEFF	0.2	0.5	0.8	0.9	1.0	1.1	1.2	1.5	1.8	2.0
0.0	-0.0485	-0.0401	-0.0216	-0.0244	-0.0222	-0.0290	-0.0370	-0.0433	-0.0401	-0.0379
0.1	-0.0251	-0.0201	-0.0077	-0.0030	-0.0014	-0.0066	-0.0080	-0.0101	-0.0063	-0.0012
0.2	-0.0017	0.0000	0.0070	0.0183	0.0218	0.0195	0.0257	0.0346	0.0449	0.0630
0.3	0.0164	0.0164	0.0229	0.0394	0.0452	0.0473	0.0513	0.0843	0.0958	0.1094
0.4	0.0342	0.0326	0.0386	0.0507	0.0688	0.0749	0.0946	0.1194	0.1328	0.1449
0.5	0.0518	0.0490	0.0545	0.0820	0.0916	0.1009	0.1213	0.1536	0.1698	0.1817
0.6	0.0696	0.0654	0.0702	0.1037	0.1124	0.1255	0.1480	0.1869	0.2080	0.2258
0.7	0.0873	0.0817	0.0860	0.1255	0.1332	0.1501	0.1749	0.2189	0.2464	0.2672
0.8	0.1005	0.0988	0.1105	0.1473	0.1539	0.1749	0.2072	0.2508	0.2768	0.2964
0.9	0.1136	0.1162	0.1358	0.1691	0.1749	0.2117	0.2396	0.2777	0.3019	0.3255
1.0	0.1269	0.1339	0.1609	0.1911	0.2091	0.2484	0.2702	0.3019	0.3271	0.3761
1.1	0.1400	0.1513	0.1855	0.2131	0.2433	0.2803	0.2971	0.3260	0.3698	0.4189
1.2	0.1531	0.1688	0.2094	0.2349	0.2779	0.3096	0.3239	0.3538	0.3976	0.4466
1.3	0.1663	0.1897	0.2334	0.2569	0.3128	0.3388	0.3527	0.3826	0.4264	0.4754
1.4	0.1854	0.2122	0.2573	0.3016	0.3477	0.3950	0.4105	0.4403	0.4842	0.5332
1.5	0.2143	0.2349	0.3590	0.3527	0.4248	0.5093	0.5248	0.5547	0.5985	0.6475
1.6	0.2433	0.2574	0.4227	0.4037	0.4758	0.5603	0.5758	0.6056	0.6494	0.6985
1.7	0.2836	0.3005	0.4658	0.4468	0.5189	0.6034	0.6189	0.6487	0.6925	0.7416
1.8	0.3435	0.3487	0.5140	0.4950	0.5671	0.6515	0.6671	0.6969	0.7407	0.7898
1.9	0.4000	0.4122	0.5775	0.5585	0.6306	0.7151	0.7306	0.7604	0.8042	0.8533
2.0	0.5304	0.5426	0.7079	0.6889	0.7610	0.8454	0.8610	0.8908	0.9346	0.9837

CL SUB ALPHA (FOR NEG. CL)

MACH NO	0.2	0.5	0.8	0.9	1.0	1.1	1.2	1.5	1.8	2.0
	4.8148	5.4828	6.5481	4.7157	4.2918	3.7085	2.8864	2.1181	1.9588	1.8074

ANGLE OF ATTACK (NO HI LIFT DEVICES)

MACH NO \ LIFT COEFF	0.2	0.5	0.8	0.9	1.0	1.1	1.2	1.5	1.8	2.0
0.0	-0.0168	-0.0180	-0.0126	-0.0225	-0.0188	-0.0251	-0.0319	-0.0372	-0.0340	-0.0314
0.1	-0.0007	-0.0010	0.0021	-0.0019	0.0014	-0.0037	-0.0040	-0.0054	-0.0016	0.0063
0.2	0.0185	0.0148	0.0175	0.0215	0.0237	0.0220	0.0284	0.0375	0.0475	0.0641
0.3	0.0377	0.0305	0.0328	0.0452	0.0463	0.0483	0.0618	0.0829	0.0963	0.1126
0.4	0.0571	0.0463	0.0482	0.0689	0.0688	0.0747	0.0944	0.1241	0.1407	0.1546
0.5	0.0764	0.0620	0.0634	0.0922	0.0913	0.1018	0.1250	0.1640	0.1841	0.1991
0.6	0.0962	0.0777	0.0787	0.1129	0.1145	0.1297	0.1555	0.1998	0.2246	0.2461
0.7	0.1166	0.0967	0.0995	0.1337	0.1375	0.1574	0.1873	0.2330	0.2641	0.2840
0.8	0.1370	0.1208	0.1267	0.1546	0.1607	0.1895	0.2211	0.2653	0.2922	0.3173
0.9	0.1573	0.1449	0.1539	0.1756	0.1892	0.2285	0.2550	0.2923	0.3203	0.3566
1.0	0.1785	0.1688	0.1810	0.2005	0.2255	0.2667	0.2861	0.3194	0.3485	0.3848
1.1	0.2053	0.1950	0.2077	0.2255	0.2616	0.2998	0.3163	0.3463	0.3768	0.4131
1.2	0.2318	0.2217	0.2342	0.2503	0.3026	0.3330	0.3464	0.3732	0.4051	0.4414
1.3	0.2583	0.2485	0.2608	0.2974	0.3437	0.3871	0.4037	0.4304	0.4623	0.4986
1.4	0.3217	0.2906	0.3581	0.3618	0.4210	0.4800	0.4965	0.5285	0.5604	0.5967
1.5	0.3845	0.3477	0.4426	0.4208	0.4983	0.5728	0.5903	0.6222	0.6541	0.6905
1.6	0.4465	0.4168	0.5117	0.4899	0.5674	0.6419	0.6594	0.6913	0.7233	0.7596
1.7	0.5208	0.4911	0.5861	0.6081	0.6854	0.7599	0.7774	0.8093	0.8412	0.8776
1.8	0.5952	0.5655	0.6604	0.6822	0.7597	0.8343	0.8517	0.8837	0.9156	0.9519
1.9	0.6695	0.6398	0.7348	0.7566	0.8341	0.9086	0.9261	0.9580	0.9900	1.0263
2.0	0.7439	0.7142	0.8091	0.8310	0.9084	0.9830	1.0004	1.0324	1.0643	1.1006

CL SUB ALPHA F4CLA (FOR NEG. CL)

MACH NO	0.2	0.5	0.8	0.9	1.0	1.1	1.2	1.5	1.8	2.0
	5.2087	6.3310	6.5109	4.2441	4.4588	3.8454	3.0396	2.2647	2.0426	1.8816

DATA SET FOR AIRCRAFT B (Continued)

DRAG COEFFICIENT (NO HI LIFT DEVICES)

MACH NO LIFT COEFF	0.2	0.5	0.8	0.9	1.0	1.1	1.2	1.5	1.8	2.0
0.0	0.0250	0.0250	0.0260	0.0276	0.0335	0.0455	0.0460	0.0430	0.0409	0.0411
0.1	0.0250	0.0250	0.0270	0.0280	0.0346	0.0456	0.0465	0.0437	0.0415	0.0434
0.2	0.0279	0.0287	0.0342	0.0343	0.0471	0.0553	0.0550	0.0528	0.0558	0.0585
0.3	0.0310	0.0327	0.0414	0.0411	0.0596	0.0669	0.0647	0.0636	0.0740	0.0836
0.4	0.0341	0.0366	0.0486	0.0480	0.0720	0.0785	0.0783	0.0903	0.1051	0.1153
0.5	0.0372	0.0405	0.0559	0.0610	0.0884	0.1005	0.1039	0.1187	0.1422	0.1595
0.6	0.0450	0.0445	0.0631	0.0945	0.1234	0.1320	0.1296	0.1620	0.1989	0.2149
0.7	0.0587	0.0588	0.0854	0.1280	0.1583	0.1636	0.1624	0.2099	0.2574	0.2824
0.8	0.0724	0.0891	0.1263	0.1616	0.1932	0.2023	0.2070	0.2609	0.3370	0.3561
0.9	0.0861	0.1194	0.1673	0.1959	0.2357	0.2523	0.2517	0.3314	0.4165	0.4300
1.0	0.1018	0.1497	0.2107	0.2488	0.2893	0.3036	0.3070	0.4019	0.4961	0.5100
1.1	0.1287	0.1899	0.2616	0.3018	0.3430	0.3623	0.3650	0.4724	0.5757	0.5900
1.2	0.1556	0.2332	0.3125	0.3547	0.4037	0.4211	0.4229	0.5434	0.6553	0.6700
1.3	0.1825	0.2766	0.3634	0.4117	0.4643	0.4823	0.4910	0.6144	0.7349	0.7500
1.4	0.2161	0.3215	0.4165	0.4721	0.5264	0.5466	0.5600	0.6854	0.8145	0.8300
1.5	0.2553	0.3679	0.4733	0.5330	0.5924	0.6110	0.6290	0.7564	0.8941	0.9100
1.6	0.2977	0.4152	0.5307	0.5940	0.6584	0.6754	0.6980	0.8274	0.9747	0.9900
1.7	0.3384	0.4675	0.5886	0.6550	0.7244	0.7398	0.7670	0.8984	1.0543	1.0700
1.8	0.3783	0.5267	0.6466	0.7160	0.7904	0.8042	0.8360	0.9694	1.1339	1.1500
1.9	0.4182	0.5859	0.7046	0.7770	0.8564	0.8686	0.9050	1.0404	1.2135	1.2300
2.0	0.4582	0.6451	0.7626	0.8380	0.9224	0.9330	0.9740	1.1114	1.2931	1.3100

CD (WITH HI LIFT DEVICES)

MACH NO LIFT COEFF	0.2	0.5	0.8	0.9	1.0	1.1	1.2	1.5	1.8	2.0
0.0	0.0420	0.0420	0.0420	0.0383	0.0457	0.0461	0.0458	0.0456	0.0455	0.0452
0.1	0.0420	0.0420	0.0420	0.0384	0.0462	0.0469	0.0468	0.0468	0.0465	0.0461
0.2	0.0420	0.0420	0.0461	0.0462	0.0555	0.0552	0.0543	0.0527	0.0524	0.0516
0.3	0.0473	0.0497	0.0553	0.0553	0.0655	0.0665	0.0644	0.0605	0.0618	0.0637
0.4	0.0529	0.0573	0.0645	0.0644	0.0754	0.0779	0.0788	0.0806	0.0797	0.0795
0.5	0.0586	0.0650	0.0737	0.0735	0.0903	0.1009	0.1052	0.1015	0.0975	0.0976
0.6	0.0634	0.0727	0.0829	0.0950	0.1241	0.1332	0.1316	0.1289	0.1285	0.1276
0.7	0.0700	0.0804	0.0922	0.1205	0.1579	0.1656	0.1581	0.1663	0.1613	0.1607
0.8	0.0950	0.1036	0.1197	0.1461	0.1917	0.1981	0.2056	0.2038	0.2041	0.2074
0.9	0.1200	0.1348	0.1489	0.1717	0.2257	0.2504	0.2530	0.2541	0.2537	0.2541
1.0	0.1450	0.1659	0.1780	0.2062	0.2809	0.3027	0.3057	0.3111	0.3032	0.3017
1.1	0.1699	0.1971	0.2121	0.2435	0.3361	0.3638	0.3700	0.3682	0.3550	0.3592
1.2	0.1949	0.2283	0.2521	0.2808	0.3988	0.4301	0.4342	0.4260	0.4070	0.4167
1.3	0.2199	0.2651	0.2920	0.3181	0.4704	0.4963	0.4986	0.4840	0.4590	0.4742
1.4	0.2468	0.3048	0.3319	0.3624	0.5420	0.5604	0.5656	0.5420	0.5110	0.5317
1.5	0.2768	0.3444	0.3800	0.4090	0.5910	0.6200	0.6326	0.6000	0.5630	0.5892
1.6	0.3068	0.3841	0.4300	0.4590	0.6390	0.6765	0.6996	0.6580	0.6150	0.6467
1.7	0.3389	0.4285	0.4803	0.5103	0.6870	0.7330	0.7666	0.7160	0.6670	0.7042
1.8	0.3748	0.4742	0.5317	0.5616	0.7350	0.7895	0.8336	0.7740	0.7190	0.7617
1.9	0.4123	0.5208	0.5832	0.6131	0.7830	0.8460	0.9006	0.8320	0.7710	0.8192
2.0	0.4388	0.5504	0.6347	0.6646	0.8310	0.9025	0.9676	0.8900	0.8230	0.8767

DATA SET FOR AIRCRAFT B (Continued)

CLMAX

MACH NO	0.2	0.4	0.6	0.8	0.9	1.0	1.1	1.2	1.4	1.6	1.8	2.0	2.2	2.4
	1.92	1.90	1.84	1.62	1.46	1.30	1.16	1.11	0.96	0.83	0.81	0.86	0.90	1.00

MIN AND MAX MACH NUMBERS

ALTITUDE	M-MIN	M-MAX
0.	0.17	1.20
5000.	0.20	1.30
10000.	0.21	1.40
15000.	0.23	1.60
20000.	0.26	1.79
25000.	0.29	2.00
30000.	0.34	2.20
35000.	0.40	2.29
40000.	0.54	2.36
45000.	0.65	2.35
50000.	0.90	2.29
55000.	1.40	2.23
60000.	2.16	2.16

DIVE RECOVERY ANGLE

ALTITUDE	MACH NO	0.20	0.50	0.80	0.90	1.00	1.10	1.20	1.50	1.80	2.00	2.20	2.40
	0.		0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
5000.		90.0000	90.0000	78.0000	66.0000	57.5000	51.2000	46.1000	35.9000	29.4000	26.3000	23.8000	21.8000
10000.		90.0000	90.0000	90.0000	90.0000	90.0000	83.8000	73.5000	54.7000	44.0000	39.1000	35.2000	32.0000
15000.		90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	73.3000	57.4000	50.4000	45.1000	40.8000
20000.		90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	71.4000	61.9000	54.8000	49.3000
25000.		90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	87.2000	74.1000	64.9000	58.0000
30000.		90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	88.2000	76.0000	67.2000
35000.		90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	88.7000	77.4000
40000.		90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	86.6000
45000.		90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000	90.0000

SPEED BRAKE DRAG COEFF(FULL DEFLECTION)

MACH NO	0.2	0.4	0.6	0.8	0.9	1.0	1.1	1.2	1.4	1.6	1.8	2.0	2.2	2.4
	0.04800	0.04800	0.04800	0.04900	0.05300	0.05200	0.04800	0.04200	0.03600	0.03500	0.03500	0.03400	0.03200	0.03000

DATA SET FOR AIRCRAFT B (Concluded)

1. Report No. NASA CR-3985	2. Government Accession No.	3. Recipient's Catalog No.	
4. Title and Subtitle Improvements to the Adaptive Maneuvering Logic Program		5. Report Date June 1986	6. Performing Organization Code
		8. Performing Organization Report No. H-1303	
7. Author(s) George H. Burgin		10. Work Unit No. RTOP 55-66-11	11. Contract or Grant No. NAS2-11421
9. Performing Organization Name and Address TITAN Systems, Inc. P.O. Box 12139 La Jolla, California 92037		13. Type of Report and Period Covered Contractor Report -- Final	
		14. Sponsoring Agency Code	
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D.C. 20546			
15. Supplementary Notes NASA Technical Monitor: Eugene L. Duke, NASA Ames Research Center, Dryden Flight Research Facility, Edwards, California 93523-5000			
16. Abstract <p>The Adaptive Maneuvering Logic (AML) computer program simulates close-in, one-on-one air-to-air combat between two fighter aircraft. This final report prepared under Contract NAS2-11421 describes three important improvements made to the AML program.</p> <p>First, the previously available versions of AML were examined for their suitability as a "baseline" program. The selected program was then revised to eliminate some programming bugs which were uncovered over the years. The report contains a listing of this baseline program in Appendix A.</p> <p>Second, the equations governing the motion of the aircraft were completely revised. This resulted in a model with substantially higher fidelity than the original equations of motion provided. It also completely eliminated a problem by which the AML was plagued since its inception: the "Over-the-Top" problem, which occurred in the older versions when the AML-driven aircraft attempted a vertical or near vertical loop.</p> <p>Third, the requirements for a versatile generic, yet realistic, aircraft model were studied and implemented in the program. The report contains detailed tables which make the generic aircraft to be either a modern, high performance aircraft (Aircraft A), or an older high performance aircraft (Aircraft B) or a previous generation jet fighter (Aircraft C).</p>			
17. Key Words (Suggested by Author(s)) Air combat maneuvering Computer programs Guidance		18. Distribution Statement Unclassified -- Unlimited STAR category 05	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 188	22. Price* A09

*For sale by the National Technical Information Service, Springfield, Virginia 22161.

NASA-Langley, 1986