NOAA Technical Memorandum ERL ARL-87

# THE TRAPS SONIC BOOM PROGRAM

Albion D. Taylor

Air Resources Laboratories
Silver Spring, Maryland
July 1980

noaa NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION / Environmental Research Laboratories

NOAA Technical Memorandum ERL ARL-87

# THE TRAPS SONIC BOOM PROGRAM

Albion D. Taylor

Air Resources Laboratories
Silver Spring, Maryland
July 1980


**UNITED STATES DEPARTMENT OF COMMERCE**
**Philip M. Klutznick, Secretary**

NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION
Richard A. Frank, Administrator

Environmental Research Laboratories
Wilmot N. Hess, Director

NOTICE

The Environmental Research Laboratories do not approve, recommend, or endorse any proprietary product or proprietary material mentioned in this publication. No reference shall be made to the Environmental Research Laboratories or to this publication furnished by the Environmental Research Laboratories in any advertising or sales promotion which would indicate or imply that the Environmental Research Laboratories approve, recommend, or endorse any proprietary product or proprietary material mentioned herein, or which has as its purpose an intent to cause directly or indirectly the advertised product to be used or purchased because of this Environmental Research Laboratories publication.

NOTE FOR THE ELECTRONIC VERSION

The original tech memo included a microfiche giving a copy of the FORTRAN CODE for TRAPS, due to the size of the printouts. For the electronic (pdf) version, this code is now included as an additional appendix.

TABLE OF CONTENTS

## Abstract

A new program called TRAPS has been written having the capability of modeling an aircraft-created sonic boom. Like an earlier program (ARAP), this program allows the aircraft to perform an arbitrary sequence of maneuvers, accelerations and deaccelerations, and it uses a stratified atmospheric model of either a standard or user specified composition. The new program introduces the new feature of accounting for sonic booms which travel upward initially, but are subsequently refracted from the stratopause (~50,000 meters) or the thermosphere (~100,000 meters). Overpressures and shocks are computed from an initial aircraft F-function on the basis of Aging and Hilbert Transforms applied according to the travel paths (rays) of the acoustic energy. In addition, input procedures are simplified and information is made available as to what proportion of the aircraft sonic boom can intercept the ground.

# THE TRAPS SONIC BOOM PROGRAM

Albion D. Taylor
National Oceanic and Atmospheric Administration
Air Resources Laboratories
Silver Spring, MD 20910

July 1980

## 1. Introduction

In 1969, members of the Aeronautical Research Association of Princeton, Inc. wrote a computer program (referred to in this report as the ARAP program) which investigates the propagation of pressure disturbances (sonic booms) from a supersonic aircraft. The program traces such a disturbance as it follows an acoustic ray path and adjusts the form of the disturbance according to weak non-linear interaction (aging) until it strikes the ground.

The program was fully documented by its authors, Hayes, Haefeli, and Kulsrud in their report [Hayes et al:1969], referred to in this report as the ARAP report.

The ARAP program has been frequently used in the forecast of the sonic boom, as an adjunct to experiment planning, and as a planning tool for the regulation of supersonic traffic, and it has enjoyed a notable agreement with observation [Herbert & Hass:1971], [Haglund & Kane:1974], [Maglieri et al:1972]. However, its applicability has always been restricted to the sonic boom carpet directly below the aircraft (direct sonic boom). It was never designed to treat the situation near the edges of the carpet, or where the ray encounters a folding surface or caustic, as may occur during acceleration or in certain maneuvers (the so-called "superboom"). It cannot follow the boom on reflection from the ground, or when refraction causes it to curve upward, or handle an upward moving disturbance in any way.

Furthermore, the preparation of input data for the program has often proven awkward. The maneuver data for the aircraft were required in terms of Mach numbers and of elevation and azimuth angles of the motion relative to the surrounding air, whereas the available data were often in terms of position relative to the ground. Not only did velocities have to be derived, but a windage correction had to be applied and the sound speed at the aircraft had to be considered. Again, the ARAP program calls for temperature and pressure of the atmosphere to be entered as a function of height. In practice, by contrast, meteorological upper air soundings do not measure or report height but require height to be computed from the pressure and temperature. Thus, much precomputation must be done before the ARAP program can be used.

In its original form, the ARAP program even requires some post processing to be performed on its output before an actual pressure signature results, since it stops at obtaining the information required to fit shock waves to the signature and does not go on to produce the signature itself. Rather, instructions are provided in the documentation for the graphical completion of the task. (This fact has led several users, such as NOAA [Herbert & Hass:1971] to modify the program to perform this task).

From 1976 through 1978 there were reports of audible and infrasonic disturbances in the eastern U.S. and Canada [Balachandran et al:1977], [NRL:1978], [DOT:1978] and analysis showed that at least some of these could be correlated with the scheduled flights of the Concorde. These disturbances could not have been the direct sonic boom that the ARAP program was designed to handle, since the Concorde flights were adequately slowed at sufficient distance from the coastline, but they could have been shock waves which had reflected from the water, and then traveled to a height of 50 to 100 kilometers before refracting back toward the ground. Alternately, the boom might have been initially headed upward.

Since these propagation modes are precisely those which the existing ARAP program could not handle, it was decided to rewrite the program to introduce this capability. In 1979, the Air Resources Laboratories of NOAA undertook this task.

A review of the ARAP program and its physical foundations (geometric acoustic theory) was undertaken. It showed that there was nothing in geometric acoustics that prevents its application to rays that reflect from a surface or refract through the horizontal to begin moving downward. Nevertheless, the mathematical analysis on which the program was based began with a choice of coordinates which introduced an artificial infinity in the ray tube area calculations. Further, since height was used as the independent variable, there were logical difficulties in handling rays which alternately traveled upward and downward. The consequences of these choices permeated the program so thoroughly that it was impractical to simply modify the existing program.

Accordingly, it was decided to write a completely new program, using the same fundamental physical concepts and generally the same terminology, but for which the mathematical analysis, program logic, and some definitions were completely revised. The new program has been designated TRAPS (Tracing Rays and Aging Pressure Signatures). In the course of this, the data input routines and the output routines were also rewritten to remove some of the pre- and post-processing burdens from the user.

This report outlines the features of the new program and describes how to prepare the data, run the program, and interpret the results.

## 2. **Ray Characteristics**

The pressure waves traced by the ARAP program can follow only one path - from the aircraft directly to the ground. The new TRAPS program, on the other hand, must deal with a wide variety of paths, including possibly multiple strikes of the ground by the same ray. To avoid confusion in interpreting the results, it is essential to have a systematic classification of the results according to the pathway over which the ray traveled. Such a classification may be provided by considering the mechanism of the ray propagation and the nature of our atmosphere.

Like the ARAP program, the TRAPS program assumes the atmosphere (pressures, temperatures and winds) to be stratified in the vertical but uniform in the horizontal direction and steady in time. These assumptions impose stringent conditions on the possible paths of motion (rays) of the wave. This motion is governed by a variant of Snell's law, which by virtue of the stratification of the atmosphere, requires the horizontal components of wave number, the frequency, and hence the horizontal velocity of the phase surfaces of the wave to be constant with respect to the ground. This constant differs from one ray to another. When combined with the requirement that the net speed be that of sound relative to the air, it determines the size of the vertical component of motion, and thus the motion itself. The result is that, for each ray, there are combinations of wind velocity and temperature at which it cannot exist (see Admittance Criteria, below). Where the ray can exist, its path curves toward regions more favorable to it; i.e. toward levels where the sound speed is lower and/or where the wind component in its direction is greater. Since the sound speed is proportional to the square root of the temperature, for each ray there is a critical combination of temperature and wind velocity that will cause its vertical motion to slow, stop, and reverse.

It should be noted that a downward moving ray which meets such a reversal layer and turns away from the ground will never, because of our stratification assumptions, reach the ground no matter what path it subsequently follows, but will always reverse again at the same height.

If an upward moving ray is to be of concern to us on the ground, it must meet a reversal level in the upper atmosphere. Neglecting the wind for the moment, the conditions for reversal depend solely on the temperature. The graph of molecular-scale temperatures in Figure 1, adapted from the U.S. Standard Atmosphere, 1976 [COESA:1976], demonstrates the typical temperature behavior of the atmosphere for temperate latitudes.

The temperature drops from that at ground level to a minimum at a layer between 10 and 20 km, or the level at which the Concorde flies on its approach to the U.S. Above this level, the temperature again rises and attains a peak at the stratopause (around 50km) before dropping again at heights in the mesosphere. In the thermosphere,

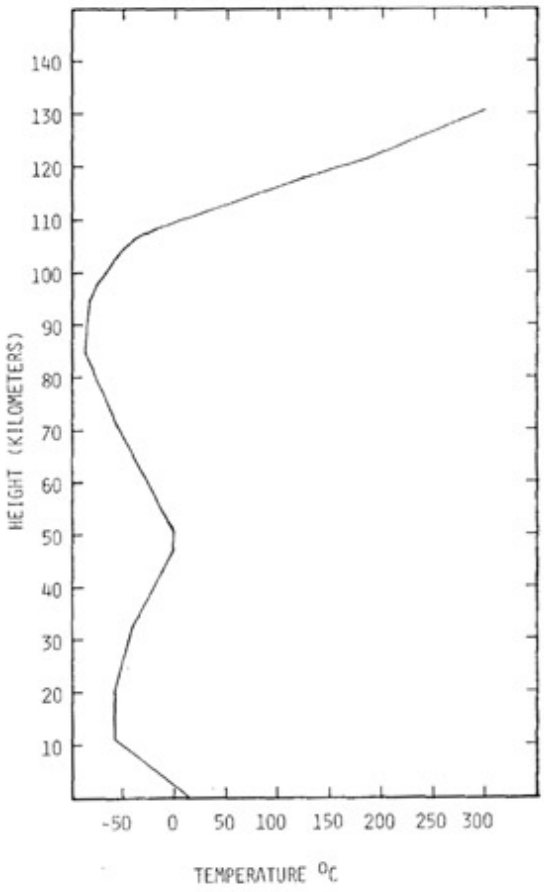



Figure 1
Vertical Profile of Molecular-Scale Temperature
Adapted from U.S. Standard Atmosphere, 1976

above approximately 100km, the temperature again rises sharply.

Because the temperature is so low at the Concorde flight level, there will generally be rays (e.g. those which leave the aircraft horizontally) which cannot reach the ground. If the aircraft speed is sufficiently low, even though supersonic, all rays will he unable to reach the ground, Such a speed is said to be below the "Threshold Mach number", see [Haglund & Kane:1974] and "Admittance Criteria" below.

At higher speeds, there will be rays, especially those directed forward and up or forward and down, for which the ground is attainable. Those moving upward must first encounter a reversal layer. Neglecting wind, this reversal layer must be at a temperature exceeding the ground temperature, or another reversal will take place above the ground.

The temperature peak at the stratopause is generally not high enough to meet this condition. Rays which are reversed here will not come within a few hundred meters of the ground before reversing again. The next opportunity for reversal is above 110 km, and the air at this altitude is so rarefied that any acoustic energy is quickly dissipated through the non-linear "aging" process, as well as linear viscosity and heat conduction. Indeed, the molecular mean free path is so long as to cast doubt on the very concept of sound propagation at these altitudes!

Referring again to the peak at 50km, it is true that seldom if ever will the temperature exceed the ground temperature. However, the winds at this level are not negligible and may attain speeds of 50 to 100 meters per second. If such a wind is in the proper direction, then in concert with the high sound speed, it can cause the reversal of a ray for which the ground is attainable.

Thus, a downward reversal is possible at two levels: one at around 50km which we will designate the M (middle) level, and one above 100 km which we will designate the H (high) level. Of course, it is also possible to reflect from the ground (G) level.

We will designate a ray path which runs directly from the aircraft to the ground as a G ray. If the path then rebounds to the M level and is reversed, when it again meets the ground it is a GMG ray. If the ray moved directly upward from the aircraft to the M layer without an initial ground reflection, we will designate it as an MG ray. If such rays pass the M level and continue to the H level before reversal, they are GHG or HG rays. If a ray reflects twice from the ground, it becomes a GMGMG or GHGHG ray. These codes will be recognized by the program, which actually makes the distinction according to whether reversal occurs above or below 70 km.

It should be noted that our stratification assumptions ensure that reversals always take place at the same altitudes, so that mixed codes such as GMGHG or GHGMG cannot occur. The overpressure signatures, overpressures, sonic boom footprints, etc. associated with a given class of ray will also be designated by the same code.

The primary sonic boom carpet below the aircraft, which is treated by the ARAP program, is the G carpet. The principal source of the audible indirect sonic boom events seems to be the GMG or the MG carpet [Gardner & Rogers:1980], [George & Kim:1979].

## 3. **The Model Atmosphere**

The ARAP program allowed the user his choice of using the built-in atmosphere, taken from the U.S. Standard Atmosphere, 1962 [COESA:1962], or of entering data consisting of a selection of pressures and temperatures at significant heights. The latter had to be computed on the basis of the available upper air radiosonde

soundings, collected on a twice-daily basis at many sites in the U.S. There was no attempt in the program to check the input data for physical consistency. The unwary user could experience some difficulty if he used a radiosonde report from a station having a higher elevation than the site in which he was interested; if he began his data entry at the station elevation (say 300m) and wished his output at sea level, the program would "blow up" for lack of data as the rays went down past 300m. To avoid this, it was important to introduce dummy data points below the elevation of the station. Another idiosyncrasy arises from the fact that input data and the Standard Atmosphere are treated differently. Thus, slight differences in the output occur if, instead of specifying directly the use of the Standard Atmosphere, an input atmosphere is used with the same significant heights, pressures and temperatures!

In the TRAPS program, it was decided to ensure consistency by requiring the identical handling of data in either case. It was also decided to write the input reader routines keeping in mind the type of data actually available, the necessity for ensuring the hydrostatic consistency of the model, and the desirability of reducing the pre-processing tasks of the user.

Direct sounding data for the atmosphere are operationally of two types: balloon-borne radiosondes and rocket-borne rocketsondes. Some indirect data are also available through processing infrared satellite imagery.

Radiosonde data are available on a twice-daily basis (0000 and 1200 GMT) from a number of stations in the U.S. The instruments measure temperatures and dewpoints at a variety of pressure levels, and winds are derived by tracking devices. In general, the resulting data apply from ground to an altitude of about 30 km, which as noted above seldom includes the stratopause so that these reports alone will not suffice for tracking the GMG rays.

Rocketsondes, being considerably more expensive, are generally available only on a weekly basis from a limited number of stations. Temperatures (but usually not pressures) are returned from a selection of heights ranging from 20 km to between 55 and 70 km. By tracking the instrument, winds are also determined.

The supplemental data from satellite infrared analysis yield heights and temperatures of the 5mb, 2mb, 1mb, and 0.4mb pressure surfaces on a once-daily basis (1200GMT). Wind information must be derived on the basis of a geostrophic assumption. These levels correspond to the range of around 35 to 55 km, so they do not extend the range of the rocketsondes, but are rather a substitute when the latter are not available. Like rocketsondes, there is generally a delay of a few days before they are available.

From the above discussion, it is apparent that the H layer is higher than all the operationally available data. If the program is to treat the H type paths at all, data at such heights must be supplied artificially except in very special circumstances.

It was decided to write the input reader subroutines to accept a composite of radiosonde type data (pressure - temperature - dewpoint) and of rocketsonde data (temperature-height), and to use data from the U.S. Standard Atmosphere, 1976 [COESA:1976] to extend the data down to -5 km and up to 130 km*. In the event that no data were provided, the result would be a copy of the Standard Atmosphere.

Where pressures and temperatures are available, the program computes heights using the hydrostatic assumption. If heights but not pressures are available, the program computes pressures under the same assumption. For consistency, the variation of temperature between levels was assumed to be the same as that in the Standard Atmosphere, namely linearly dependent on geopotential height, rather than on pressure. Slight variations may be expected between the resulting heights and those computed by other programs.

Above 86 km, the Standard Atmosphere, 1976 itself changes from a segmented linear temperature variation satisfying the hydrostatic law to a smoothly varying profile with the hydrostatic law relaxed. To avoid problems of logic in our program, we have reinstated the lower conditions in our version, forcing a segmented linear profile with hydrostatic law. The resulting model differs from the standard by no more than one degree C in temperature, and seldom by more than 1% in pressure.

Also above 86 km, the chemical composition of the Standard Atmosphere begins to change, incorporating a proportion of monatomic species which increases with altitude. This produces a significant increase in the ratio of specific heats denoted $\gamma$, an important factor in the sound speed. The ARAP program assumed $\gamma$ to be constant, but TRAPS incorporates a table of $\gamma$ values and assumes linear interpolation with height. Humidity also affects $\gamma$, although to a far lesser extent; this variation is also accounted for.

It is important when analyzing results for the user to remember that the TRAPS program does extend the input data using artificial data from the Standard Atmosphere, that this will generally affect the placement of GHG sonic boom carpets, and may affect the placement of GMG carpets.

## 4. **The Aircraft Track**

The ARAP program requires data on the motion of the aircraft in the form of Mach number and the azimuth and elevation of its motion relative to the air, and the bank angle of the aircraft. Whereas these coordinates are very natural to use in determining the initial

---

* The 1976 version of the Standard Atmosphere differs from the 1962 version only above 51km, and 'this does not affect the ARAP program for flights below this height.

conditions of the rays, they are not the terms of the data generally available.

The data which are available, from ground based radar tracking or from readouts of the Inertial Navigation System, consist of a set of positions (which may be related to an X-Y coordinate system on the ground) and heights sampled at time intervals typically on the order of six seconds. The TRAPS program was written to use this type of data. A natural impulse would be to compute velocities and accelerations from these data, combine them with the wind and sound speeds from the model atmosphere, and convert them to the ARAP coordinates. There is a major drawback to that procedure, however, in that the data are typically reported only to the nearest tenth of a nautical mile in the horizontal, or 100 ft vertically. The uncertainty of as much as 600 ft in position every six seconds can yield an uncertainty in speed of 100 ft/sec or 33 ft/sec/sec in acceleration. Fluctuations of 1g in acceleration every six seconds are not only unlikely in a passenger carrying aircraft, but if accepted by the program will lead to spurious strong focusing and defocusing effects on the rays, and the supposed lift required to induce them will be reflected in extra strong lift components in the overpressures.

For these reasons, it is essential to have the input reader routines smooth the position data before using. Not only must the accelerations be smoothed, but the positions and velocities must be adjusted to correspond.

The technique which was incorporated was derived from the procedure for cubic spline interpolation [Ahlberg et al:1967]. A cubic spline is an interpolation curve composed of a chain of cubic polynomials which pass through the given positions and are required to have matching velocities and accelerations. A given set of positions uniquely defines the set of accelerations at the corresponding time nodes; between those time nodes the accelerations are linearly interpolated. In the present instance, actual accelerations appear as persistent values over several time nodes while accelerations due to round-off appear as short-period oscillations with a near zero mean.

The smoothing procedure continues from this point by applying a low-pass filter to the computed accelerations. Specifically, the accelerations are interpolated to the mid-points between the nodes, and the results are interpolated back to the nodes. The resulting accelerations no longer correspond to a cubic spline through the original data points, but they do correspond to a spline through a new set of positions which can be calculated by reversing the usual spline computations. Experience with actual radar data shows that the shift in position is seldom greater than the round-off error associated with the input data.

The smoothed accelerations and adjusted positions are stored as spline parameters. When requests are issued by the ray tracing subroutines for positions, velocities, or accelerations at specific

times, they are satisfied by cubic interpolation using these parameters.

## 5. **Ray Origins and Admittance Criteria**

In a reference frame at rest in the air at the altitude of the aircraft (airborne reference frame), the normals to the phase surfaces of the wave can be taken to have vector components ($p,q,r$) in the X-, Y- and vertical (Z-) directions, respectively.These components represent the wave numbers in their respective directions; the magnitude of this vector times the sound speed is the frequency, (scaled by the aircraft length) which in the airborne reference system we take as equal to the airspeed of the aircraft.

The tips of these vectors in the airborne system must lie on a sphere whose radius is the aircraft Mach number. In addition, it can be shown that the component of the vector in the direction of the aircraft trajectory must be unity. This means that the tips of vectors must lie in the intersection of the Mach-number radius sphere with a plane normal to the aircraft motion; i.e. on a circle which we call the Mach circle (see Figure 2). The cone formed by the vectors from the origin to the Mach circle represents all the possible ray directions (in the airborne reference system) from the aircraft at any instant; we call it the ray cone and its half-angle whose cosine is the inverse of the Mach number, is the co-Mach angle. An individual ray in the cone is specified by an angle $\phi$, which is measured along the Mach circle from the lowermost ray clockwise as seen by the aircraft pilot.

In transferring from the airborne reference frame to one fixed in the ground the wave numbers $p$, $q$, and $r$ do not change at all. The frequency $\omega$ changes according to the rule

$$\Delta\omega = \Delta u\, p + \Delta v\, q$$

where $\Delta u$ and $\Delta v$ denote the components of the velocity difference between the two frames (i.e. the wind components at aircraft altitude).

Because of our stratification assumptions, it may be shown that in any unaccelerated reference frame, the parameters $\omega$, $p$ and $q$ do not change as the wave propagates along a ray. This is the acoustic version of Snell's law. In addition, the following relation, known as the Eiconal equation, holds:

$$c^2 (p^2 + q^2 + r^2) = (\omega + up + vq)^2$$

at any altitude, where $u$, $v$, and $c$ are the wind components, and the speed of sound, respectively, at that altitude.

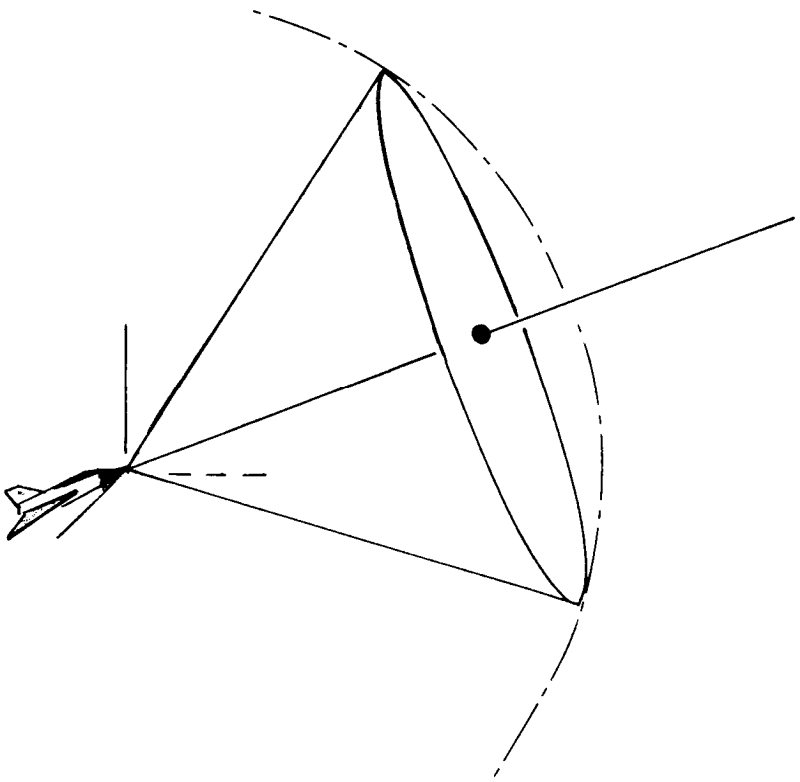


Figure 2
The Mach Sphere, the Mach Circle, and the Ray Cone

Solving for the vertical wave number $r$ yields

$$c^2 r^2 = u_A^2 - 2u_A ( (u - u_0) p + (v - v_0) q)$$

$$-(c^2 - (u-u_0)^2) p^2 + 2(u-u_0) (v-v_0) pq - (c^2 - (v-v_0)^2) q^2$$

where $u_A$, $u_0$ and $v_0$ denote the aircraft airspeed and the horizontal components of wind at the aircraft altitude, respectively. For $r$ to be real-valued, the right side must be positive, which requires the point $(p,q)$ to lie within an elliptical region (the admittance ellipse) in the $p$-$q$ plane. The size and orientation of this ellipse depends on the difference in wind components between the two regions, and the ratio of the sound speeds as well as the aircraft Mach number.

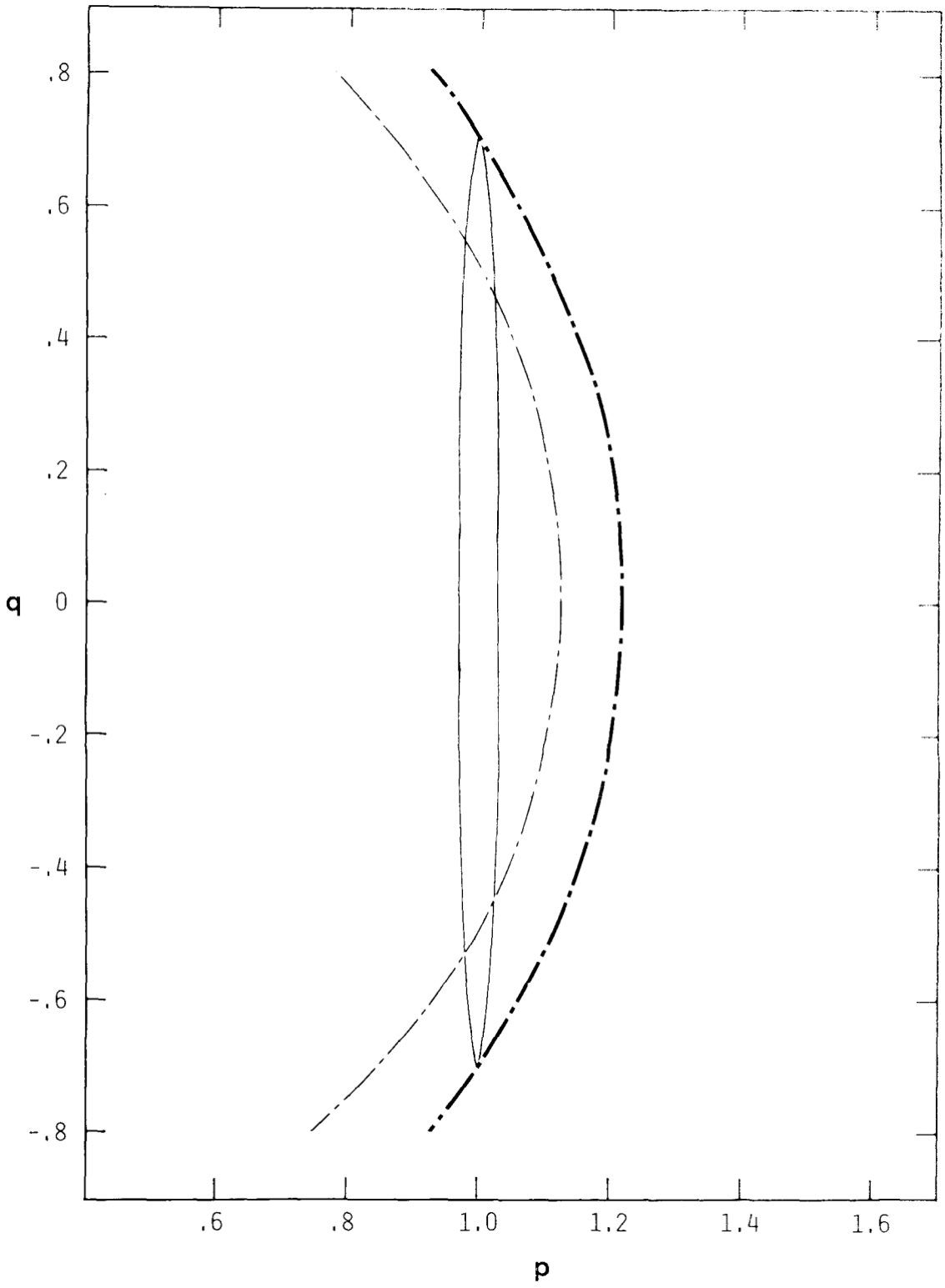



Figure 3
Solid line . . . . . Projection of Mach circle
Bold dash line . . . Projection of Mach Sphere
Dashed line . . . . Admittance ellipse

In Figure 3, we consider the admittance region together with a projection of the Mach circle, which is also an ellipse. In general, the admittance ellipse intersects this projection at four points, splitting the Mach circle into four arcs, two lying inside the admittance ellipse and two outside. The arcs within the admittance region correspond to rays which can penetrate to the altitude in question, those without cannot. One of the two admitted arcs will consist predominantly or exclusively of rays in the upper part of the Mach circle (i.e. rising rays), the other of rays in the lower part (descending rays).

We have introduced a routine into the TRAPS program to carry out the graphical construction of Figure 3 for the admittance ellipse of the ground level, and to determine between which values of $\phi$ the rays will lie inside the ellipse. There are three benefits from this:

(1) Since our interest is in the sonic boom on the ground, the program will decline to trace rays outside the admittance ellipse. This will save computer time and printout compared to the ARAP program, which traced all rays until they either struck ground or a reversal layer.

(2) For an investigation of the margins of the sonic boom carpets, the user is enabled to request the program to trace rays beginning at an edge and working inward. it is notable when this is done that occasionally some rays traced still do not reach the ground, because they lie outside admittance ellipses for altitudes between the aircraft and the ground. In practice, however, these rays reverse at altitudes only a few meters above ground and are still of interest.

(3) The program can be used to efficiently answer the question at which times on a proposed or actual flight track will the sonic boom cease to be able to strike the ground. This corresponds to the situation where the Mach circle no longer projects onto the admittance ellipse because the Mach number is too small. Indeed, this question of threshold or cutoff Mach number is answered separately for ascending and for descending rays. It may be seen from our construction that for a descending aircraft, the upper part of the Mach circle projects further from the origin of the $p$-$q$ plane and therefore ascending rays have a higher cutoff Mach number; the converse is true for a climbing aircraft. If the aircraft is flying level, the two parts project onto the same straight line and are affected symmetrically.

## 6. **Ray Tube Areas and Sonic Boom Amplitude**

At each instant of supersonic flight, the aircraft emits a cone of rays, each ray of which is singled out by specifying an angle $\phi$. The set of rays which leave the aircraft at neighboring times, between $t_e$ and $t_e + \Delta t$, say, and at neighboring angles, between $\phi$ and $\phi + \Delta\phi$, say, form a ray tube.

The total acoustic energy in a ray tube has been shown to be constant (for linear, inviscid processes) by [Blokhintzev:1946] who formulated an invariant relating the ray tube area to the inverse square of the amplitude. This invariant was used in the ARAP program and it is also used in the TRAPS program.

The definition of precisely what is the area of a ray tube differs significantly between the two programs, however. Because the ARAP program utilizes height as the independent variable along the ray, and because of certain computational advantages, the ARAP program defines ray tube area as the area of a horizontal section of the tube. This is larger than the area of a cross-section of the tube. Indeed, when the ray tube is at a reversal level, and is itself horizontal, the horizontal section is parallel to the tube axis and is infinitely greater than the cross-sectional area. Accordingly, the ARAP ray tube area is infinite here, even though the cross-sectional area is not.

By contrast, the TRAPS program defines the ray tube area as neither a horizontal section nor a cross-section, but as a section cut by the wave phase surfaces, i.e. a section normal to the wave normals.

Defined in this way, the ray tube area is always finite, but this definition of ray tube area has the additional benefit, not shared by either of the other two sections, of being a Galilean invariant. That is, it is a quantity whose value does not change when measured by an observer moving at any constant velocity. Since the amplitude of the sonic boom is clearly a Galilean invariant, as are the pressures, temperatures, densities, sound speeds and other physical quantities in the Blokhintzev invariant, it would seem that this definition is the most appropriate. This is reflected in the form of Blokhintzev invariant used in the two programs; the ARAP program requires an additional term designated $c_0$ which cancels the Galilean variance of their ray tube area. This term also becomes zero as the ray becomes horizontal so that the sonic boom amplitude takes on the indefinite form of 0 times infinity. The TRAPS program requires no such additional term and encounters no such form.

Both the ARAP and TRAPS program compute the area as a determinant (called a Jacobian) formed from partial derivatives of coordinates with respect to the ray parameters $\phi$ and $t_e$ These partial derivatives are found by integrating equations similar to the ones used to track the rays, and in fact derived from them.

This technique is superior to the alternative of actually tracking neighboring rays and computing the area of the figure formed by the endpoints, since over the distances with which we are concerned, even rays which are initially very close can spread over considerable distances, and furthermore area computations of that type are so sensitive to round-off errors in position that the error may be many times the actual area.

The Jacobian technique leads to a ray tube area that varies in a continuous manner as the ray is traced, and even the rate of change of area with position along the ray is continuous so long as the gradients of wind and sound speed are continuous in the atmosphere model. Where the gradients are discontinuous (and this occurs at each height at which either temperature or wind is input or taken from the Standard Atmosphere), the rate of change of area (but not the area) undergoes a jump. The amount of this jump is a continuous function of the ray normals, which are themselves continuous.

The result of this is that for the G carpet, the ray tube areas on the ground and the amplitudes are continuous functions of the ray parameters. Except when the ray tube area is zero, or at the edge of the carpet, they are also continuous functions of position on the ground.

For the other carpets, there may arise discontinuities in the ray tube area between rays where the reversal layer of a ray is also a surface of discontinuity of temperature or wind gradients. In such a case, a jump in areal rate of change occurs for some nearby rays which pass through the surface in both directions, but not for other neighbors which do not. Subsequently, the ray tube area becomes discontinuous between rays, but remains continuous along each individual ray.

Consequently, for carpets other than the G, there will be jumps in the ray tube area and the sonic boom amplitude. This fact is clearly due to the choice of atmospheric model, but it is arguable whether it is artificial, in that rapid changes in temperature gradients and in wind shear do occur in the atmosphere. It does suggest the sensitivity of the sonic boom model to variabilities in the atmosphere that are difficult to determine precisely.

## 7. **Folding Surfaces and Caustics**

A noteworthy feature of ray tube analysis is that ray tube areas calculated in the above manner may be either positive or negative. This apparently surprising result is understood by considering the paths of neighboring rays, as shown in the diagram in Figure 4.

In this figure, assume the airplane is flying at supersonic speed directly into the paper, and follow the progress of four neighboring rays from its ray cone. The rays indicated depart the aircraft toward the right and upward, the uppermost being labelled U and the

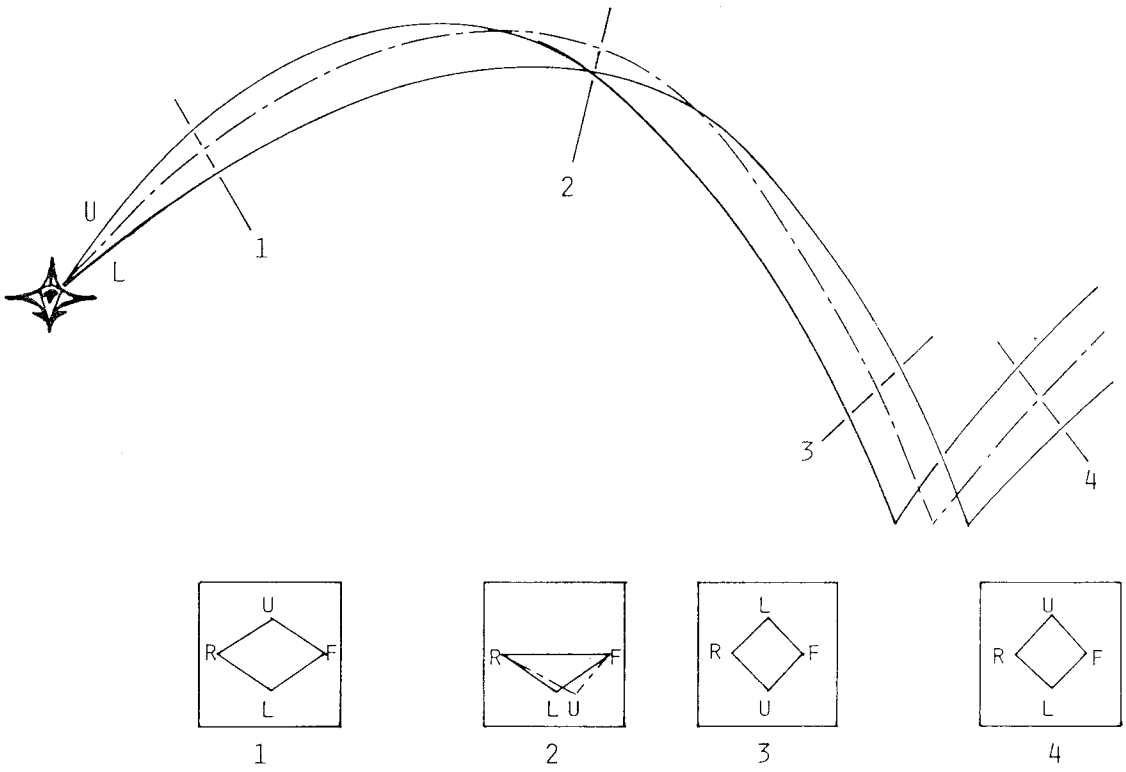



Figure 4
Sections of a Ray Tube

lowermost L. The other two rays have the same $\phi$ angle, intermediate between the U and the L; one (F) leaving from the forward part of the aircraft, the other (R) from the rearward part. Since these two have the same $\phi$ angle, they project into the same curve (dashed in Figure 4).

Early in the tracing of these rays, at the site where section 1 is taken, the TRAPS program calculates a positive ray tube area. Later, at the section 3 site, it finds a negative ray tube area. Still later, after reflection from the ground, where section 4 is located, the ray tube area is again positive.

On examining section 3, it appears that the labeled rays are positioned in a mirror image fashion from that of sections 1 or 4. This occurs because the U ray traveled to a higher reversal layer with a slower horizontal velocity than F and R, which in turn went higher and more slowly than L, so that at section 3, the U ray is lowermost and the L ray uppermost.

In general, a negative ray tube area indicates that the section being used to define the ray tube area is a mirror image (apart from scaling considerations) of the section of the same tube when emitted from the aircraft. This mirror transformation can occur either by rays crossing over each other or through reflection from the ground.

Suppose two adjoining ray tubes with a common interface leave the aircraft, and subsequently one but not the other acquires a negative area. The section of the one will be a mirror image of the other, but their interface will remain in common; perforce the section of the one will overlay the section of the other as if the common interface were the mirror in question. Alternatively, it is as if a sheet of gridded paper were folded over itself where the area changed sign, so that one side was covered twice with rays and the other not at all (c.f. section 2 from Figure 4). For this reason, a surface where the ray tube area changes sign between neighboring rays, or neighboring points of the same ray, is called a "folding surface".

This folding is clear in the case of reflection, since the region above ground is covered by two sets of rays (incident and reflected) while that below ground is covered by none; it also takes place in areas above ground determined by refraction patterns and aircraft maneuvers.

In reflection, the change from positive to negative (or vice versa) is accomplished by changing sign of a non-zero quantity. But on the ray itself, ray tube area is continuous; between any two points with area of opposite sign must lie a point where the area is zero. Such a point is called a caustic* point since the sonic boom amplitude (inversely proportional to the square root of the area) is nominally infinite. Strictly, the amplitude is not infinite, of course, since ray theory is an approximation to a more general theory. At folding and caustic surfaces, that theory takes precedence and governs the amplitude. See the discussion in "The Caustic Passage" below.

Because it is possible for the ray tube area under certain circumstances to be discontinuous across a ray, it is possible for folding surfaces to exist which are not caustics. They still separate a doubly covered region of air or ground from an uncovered region, and are prominent aspects of the carpet or footprint. But the computed amplitudes are not infinite, and any attempt to "locate the caustic" by looking for the zero of the ray tube area is likely to fail.

The finite amplitudes are no more "real" than the infinite amplitudes which they replace, however. The ray theory is based on

---

* The term "caustic" in ray theory derives from the Latin causticus (burning) and the fact that a burning glass concentrates solar energy at such points of focusing.

an assumption that the variability of amplitude is small over distances of the order of a wavelength, an assumption that fails at a folding surface. Although the amplitudes computed for this region are indicative of the concentration of acoustic energy here, exact forecasts must await a more sophisticated theory than either TRAPS or ARAP supplies, as well as more precise upper air data.

Another region where the results of the programs are problematic is at the margin of the carpet, where the reversal layer for a ray and the ground coincide. Beyond that point is a shadow zone to which rays do not penetrate, either because they are not admitted at ground level or because they were reflected from the ground short of the zone. A diffraction theory is required for a fully satisfactory treatment of wave phenomena within a few wavelengths of a shadow zone, and neither ARAP nor TRAPS provides it.

The width of these regions of doubt can be calculated in terms of the curvature of the rays and of the appropriate surface (caustic, folding, or shadow), together with the wavelength of the boom at this point. The curvature of caustic surfaces cannot be calculated by either TRAPS or ARAP, but the widths can be estimated as a few wavelengths. The wavelength in question is the length of the signature as printed out by the program and depends on the aging of the waveform (see below); it is initially a few aircraft lengths, for M-type carpets may be tens and for H-type carpets may be hundreds of aircraft lengths.

## 8. **Aging of Signatures**

In the linearized acoustic theory, the wave form of the pressure travels along the ray unchanged except for amplitude changes governed by the Blokhintzev invariant. At least below the mesopause, effects of viscosity and heat conduction are too small to seriously affect this concept.

But pressure waves of this amplitude are governed by a non-linear theory, and although the non-linear effects are small over any given region up to some tens of wavelengths in size, they do accumulate and are responsible for the typical N-wave profile of the direct sonic booms and the bulk of dissipation of acoustic energy between the aircraft and ground.

In terms of supersonic flow, the sonic boom is "weak", and both the ARAP and TRAPS programs apply a weak shock tube theory due to G.B. Whitham [Whitham:1956] to the propagation of the sonic boom in ray tubes. The details may be found in the ARAP report; in general an overpressure at a given point in the wave form so increases the air speed and sound speed at its location that it seems to overtake a lesser overpressure located ahead of it (see Figure 5). The amount of the overtaking is governed by a quantity termed the age, which increases along a ray at a rate proportional to the amplitude, and inversely proportional, among other terms, to the square root of

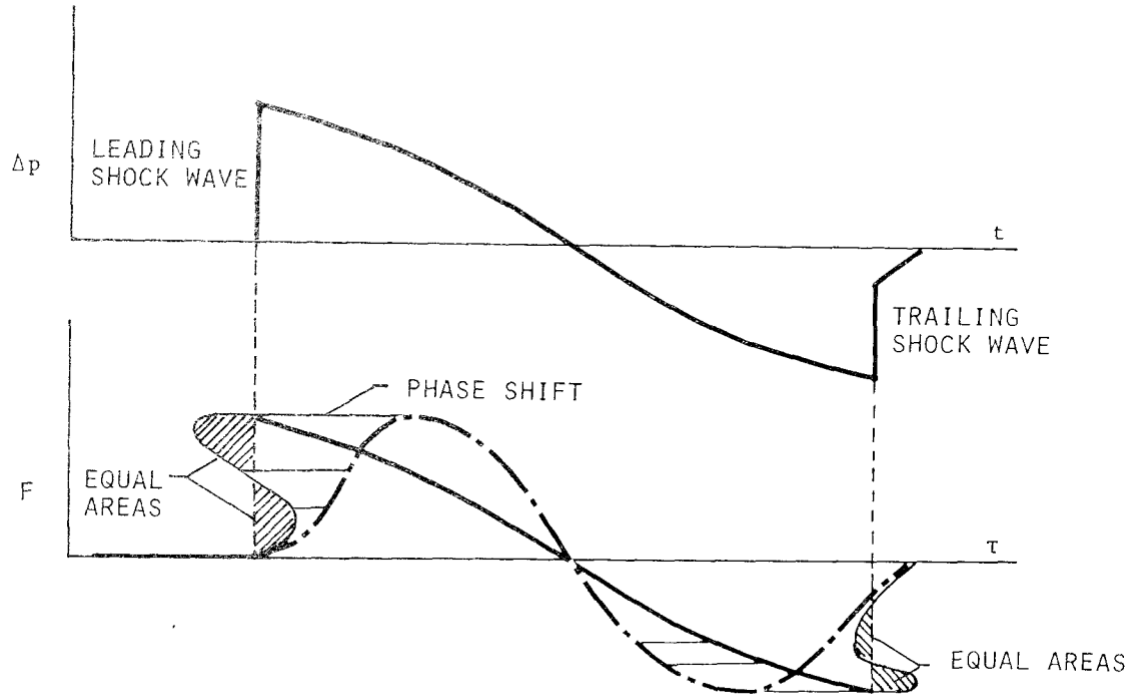



Figure 5
Signature Aging Process
Adapted from [Hayes et al:1969]

the ambient air density.

When a section of the waveform actually overtakes one ahead of it, the choice among the three or more possible values of overpressure is resolved by fitting a shock (pressure jump), thereby cutting off the lobes of the overtaking and overtaken portions. To conserve mass, the shocks are so placed as to balance the area within the cutoff lobes using the so-called "equal area rule" .

Both the APAP and TRAPS routines perform the computations of age and the slanting of wave forms and area terms. The difference between them is that the TPAPS program actually carries out the process of locating the position of the shocks, cutting off the lobes, and determining the overpressures on both sides of the jump, while the ARAP program was content to leave this process up to the user to perform manually.

## 9. **The Caustic Passage**

A review of Figure 4 will suggest that any ray tube which is reversed at an M or H layer (comprising all carpets but the G) must pass through a folding surface, and hence the individual rays must pass through a caustic surface. While this is not strictly true, in that certain maneuvers and atmospheres can combine to prevent this, it is true for the bulk of these rays. By contrast, the bulk of rays in the G carpet do not pass through caustics except during special maneuvers.

As noted above, ray theory does not apply in the immediate vicinity of a caustic, even though rays may be traced through the caustic and ray tube areas computed without difficulty. For this reason, the ARAP program discontinues calculation whenever a caustic surface is encountered.

In fields other than acoustics, such as water wave theory or optics to which ray theory applies, a more general theory known as Uniform Asymptotic theory may be applied [Ludwig:1956]. Indeed, this more general theory holds for linearized acoustics as well, and can be used to determine the shape of the wave departing the caustic, given the shape of the wave approaching the caustic. After passage through the caustic, ordinary ray theory holds once again and the program may resume, now propagating the new signature. It is this technique which the TRAPS program uses to continue the evolution of the sonic boom.

It is a nearly invariable conclusion of the Uniform Asymptotic theory, to whichever physical process it has been applied, that the Fourier components of the outgoing signal are the same as they would be expected to be from the naive ray theory, except that each one has been shifted forward one quarter wavelength. Since the shorter wavelength components advance less than the longer components, the shape of any complex waveform such as ours can change significantly.

This transformation is commonly known by the name of "90 degree phase shift" (since there are 360 degrees in a full wave cycle) and there is a temptation to perform it by actually taking a finite Fourier transform, changing the coefficients, and inverting. However, even with the Fast Fourier Transform, this is an extremely inefficient procedure.

The reason lies in the shape of our input signal, which by the time of caustic passage has usually aged into a nearly N-wave form. As shown in Figure 6, the transform of the N-wave has two very thin peaks (logarithmic discontinuities) located where the jumps were. To resolve these peaks requires a number of very closely spaced points in their immediate vicinity. Elsewhere, the waveforms are smooth and such close spacing is extremely wasteful of computer resources. In particular, a much wider spacing should be used far ahead of and far behind the original waveform. But finite Fourier transforms require a uniform spacing of points, forcing a choice

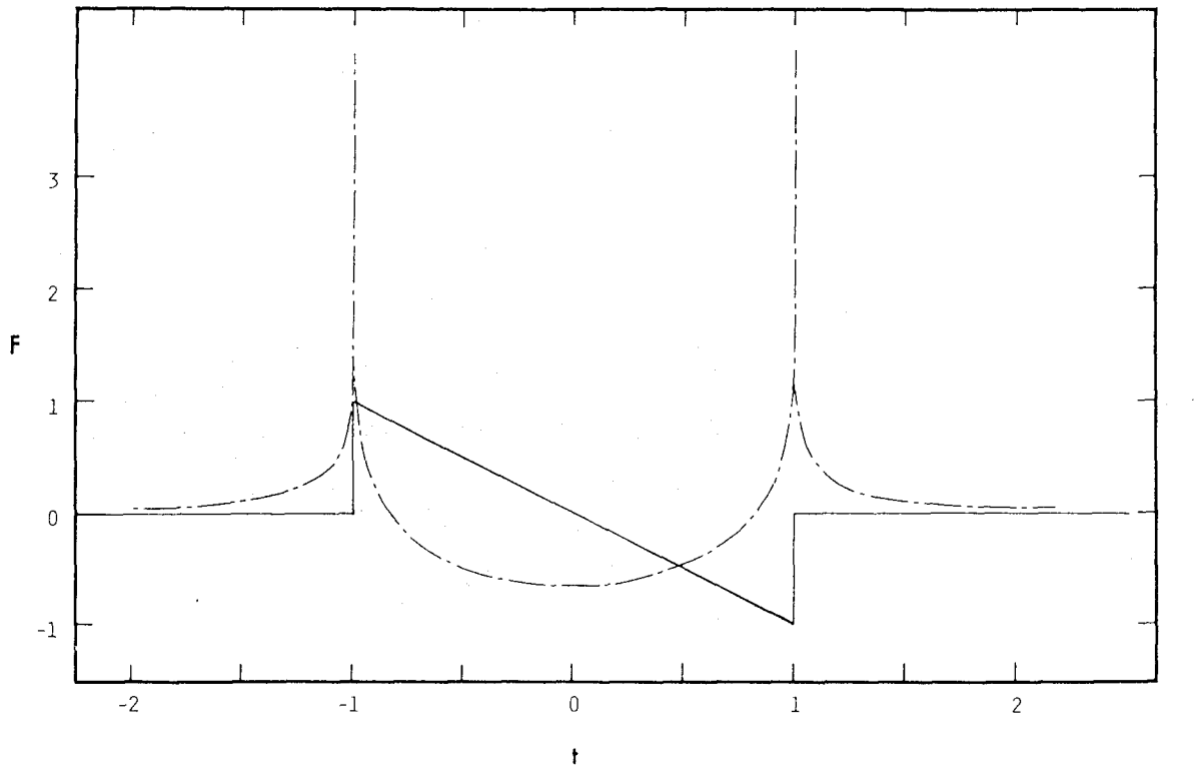



Figure 6
Caustic transformation of F-function

between inadequate resolution and waste of resources.

TRAPS uses an alternative to the above Fourier techniques, called the Hilbert Transform [Erdelyi et al:1954]. This is an integral transform with a singular kernel whose Fourier equivalent happens to be the 90 degree phase shift; it has the advantage that it may be evaluated at an arbitrary selection of points whose spacing may be chosen with the above principles in mind.

In the TRAPS program, the sonic boom signature is taken through the following evolutionary steps:

(i) Compute the age until the ground or a caustic is encountered;
(ii) Age the signature and fit shocks as appropriate;
(iii) If at a caustic, perform the Hilbert Transform and create a new signature;
(iv) Continue with step (i) until final ground contact.

Despite the aegis of the Uniform Asymptotic theory, there are potential doubts in our procedure, in that the shocks of the N-wave

indicate the operation of non-linear effects and the theory applies to linear systems. This doubt is reinforced by the appearance of infinities in the Hilbert transform of the N-wave. In reality, however, the N-wave with the shocks is an approximation to the actual signature. Since the sonic boom is weak, in the sense of supersonic flow theory, the shocks are not strong, well established features. Measurements often show [Herbert & Hass:1971] a "rise time" for the shocks of between 1/30 and 1/10 of the length of the N-wave, presumably due to some form of turbulent dispersion. With such a "thick shock", the infinities in the wave form all disappear, and the Uniform Asymptotic theory, if carried out, would lead to finite overpressures up to and past the caustic surface. This result places the validation of the Uniform Asymptotic theory on the same level as ray theory, as an approximation to the linear acoustic equations, and the validation of the linear theory as an approximation to the non-linear theory on the same level near the caustic as elsewhere.

## 10. **Summary**

The TRAPS program extends the capabilities of the earlier ARAP program to the handling of waves that have passed through a caustic surface, and to waves that have risen to high altitudes and returned to the ground. The theoretical background is on a level with the ARAP program, with the added feature of accomodating caustic passage, the validity of which is felt to be on a level with ray theory. The program is easier to use in that data entry requirements are closer to the available sources of data, and the post-processing burdens are also eased.

The user should be cautious when interpreting the results of either program. Amplitudes and waveforms sampled too near to a caustic or folding surface, or to a shadow zone, must be taken with a grain of salt. "Too near" is a concept which depends on the relation of the wave length to the relative curvature of the ray and the caustic. Neither the ARAP nor the TRAPS program can measure the curvature of the caustic surface, but in general a few wavelengths from that surface will suffice, and the size of a wavelength will be approximately that of the wave form the program supplies (on the order of one to hundreds of aircraft lengths, depending on aging and carpet type).

Away from these surfaces, the results are on a firm theoretical foundation and may be expected to be as good as the input data. When investigating the MG and GMG carpets, it is important to secure good atmospheric data up to around 55km; this calls for rocketsondes, which may not be available closer than hundreds of kilometers in distance and days in time. In view of the fact that the atmosphere can easily change over such an interval, one must allow for possible shifts in the location of the *MG* and *GMG* carpets and some changes in the amplitudes.

## 11. **References**


Ahlberg, J.H., E.N. Nilson & J.L. Walsh, "*The Theory of Splines and Their Applications*", Chapter II, Academic Press 1967

Balachandran, N.K., W.L. Donn and D.H. Rind, "*Concorde Sonic Booms as an Atmospheric Probe*" Science v 197 (July, 1977), pp 47-49

Blokhintzev, D.I., "*The Propagation of Sound in an Inhomogeneous and Moving Medium Part I*", J.Acoust. Soc. Am., vol 18, (1946) pp 322-334

Carlson, H.W.,"*Correlation of Sonic-Boom Theory with Wind-Tunnel and Flight Measurements*" NASA TR R-213, NTIS, Springfield, VA (1964)

COESA, "*The U.S. Standard Atmosphere, 1962*", U.S. Govt. Printing Office, Washington, D.C. (1962)

COESA, "*The U.S. Standard Atmosphere, 1976*", U.S. Govt. Printing Office, Washington, D.C. (1976)

DOT, "*Concorde Monitoring at John F. Kennedy International Airport, April 1978*" Dept. of Transportation unpublished report 1978

Erdelyi et al, "*Tables of Integral Transforms*", Vol II, Chapt. XV McGraw-Hill 1954 (Bateman Manuscript Project)

Gardner, J.H. & P.H. Rogers, "*Thermospheric Propagation of Sonic Booms from the Concorde Supersonic Transport*" J. Acoustic Soc. America v 67, (1980) pp 78-91

George, A.R. & Y.N. Kim, "*High-Altitude Long-Range Sonic Boom Propagation*" Journal of :Aircraft, v 16, n 9 (1979) pp637-639

Haglund, G.T. & E.J. Kane, "*Analysis of Sonic Boom Measurements Near Shock Wave Extremities for Flight Near Mach 1.0 and for Airplane Accelerations*" NASA CR-2417, NTIS, Springfield, VA (1974)

Hayes, W.D., R.C. Haefeli & H.E. Kulsrud, "*Sonic Boom Propagation in a Stratified Atmosphere, with Computer Program*" NASA CR-1299, NTIS, Springfield, VA (1969)

Herbert, G. & W. Hass, "*The Pendleton Project A Study of the Atmospheric Effect on Weak Shock Waves Traversing Long Ray Paths*" NOAA TR ERL 220-ARL 1, U.S. Govt. Printing Office, Washington, D.C. (1971)

Ludwig, D., "*Uniform Asymptotic Expansions at a Caustic*", Comm. Pure & Appl. Math. v 19, (1956) pp 215-250

Maglieri, D.J., V. Huckel, & H.R. Henderson, "*Sonic Boom Measurements for SR-71 Aircraft Operating at Mach Numbers to 3.0 and Altitudes to 24384 Meters*" NASA TN D-6823, NTIS, Springfield, VA (1972)

NRL, "NRL *Investigations of East Coast Acoustics Events 2 December 1977 - 15 February 1978*" Naval Research Laboratory unpublished report, March, 1978

Whitham, G.B., "*On the Propagation of Weak Shock Waves*", J. Fluid Mech. v 1, (1956) pp 290-318

# Appendix A - Structure of TRAPS

## Environment of the Program

The TRAPS program was written for use on the IBM 360/195 computer system at the NOAA computer site in Suitland, Maryland. It was written in the FORTRAN H+ EXT language, a superset of the ANSI (1966) FORTRAN language and a subset of the ANSI (1976) language. To enhance portability to other computers, care was taken to remain within the 1966 standard as far as practicable, but several machine and installation dependent features were either necessary or so convenient as to justify their inclusion. Installation on other machines requires consideration of these variations from standard, which all involve practices that have counterparts on most other large computers.

Specifically, we have used the following:

(1) TYPE REAL*8 and TYPE LOGICAL*l for the manipulation of character strings and individual characters. These may be replaced in any language which provides for explicit character string manipulation.

(2) Type REAL*8 for double-precision calculations. This was important for ray calculations very near a reversal layer.

(3) INTEGER*2 for half-word storage of integers. These may be replaced by ordinary INTEGER type statements with no effect on results.

(4) Special subroutines DREAD, FFA2I, and FFA2F called by the FREAD subroutine. These subroutines are available only at the NOAA site. Subroutines FFA2I and FFA2F convert character strings to binary integer or floating point numbers and may be replaced by similar calls to FFA2N, which is supplied on the fiche. Subroutine DREAD is part of a direct-access read-write package, and many other installations have similar software. If necessary, the entire FREAD subroutine may be replaced by a routine which backspaces or rewinds the file, searching for the correct card, but direct access is far preferable.

(5) A call to FFF2A in SUBROUTINE PTDHIN. FFF2A converts binary floating point numbers to character strings, and was used to facilitate output in which missing values would ideally be represented by blanks. It may be replaced by equivalent routines on other systems, or by allowing missing values to be represented by special numeric codes.

(6) A special subroutine DATIM2, called by SUBROUTINE SETUP. It supplies the calendar date and time in character string form. The date is merged with the

user-supplied title to assist in documentation of the computer output. Similar routines are available at many computer sites, or if not, the entire reference may be withdrawn.

Program Listings

A source listing of each of the subroutines in TRAPS is provided on the attached microfiche. A cross-reference listing of variables, SUBROUTINE calls, and COMMON blocks is provided for each subroutine, along with the source code itself.

The first item provided is actually a utility program for preparing the F-FUNCTION File (see Appendix B), and is not properly a part of TRAPS per se. When TRAPS is executed, this utility program should already be finished with its task.

The second item provided, listed in the fiche index as SONBOM, is the MAIN program for TRAPS and exercises control over all other subroutines. The remaining subroutines used in TRAPS are listed in alphabetic order and are all present, except for standard library routines and the exceptions cited in (4) through (6) above.

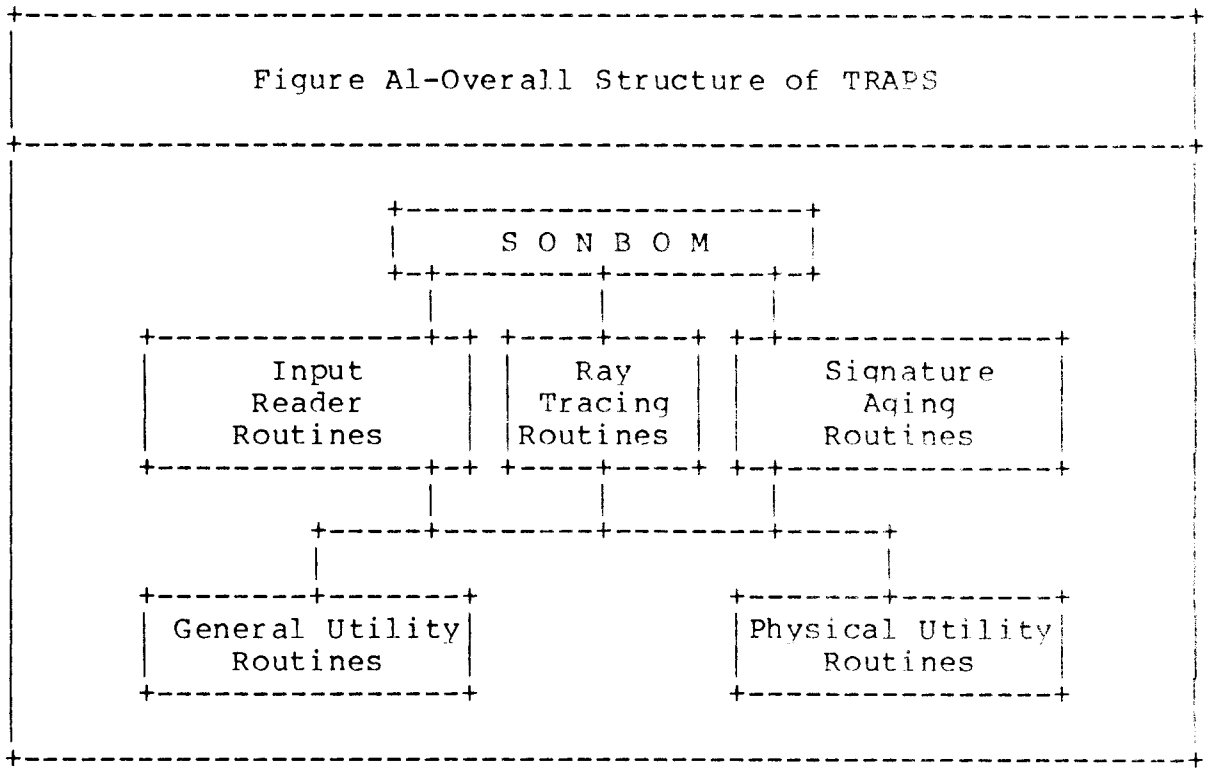
## Organization of the Program

Traps has been written in a modular form, in which each distinct function to be performed is assigned to a distinct subroutine. This method of programming simplifies future modifications to the program, e.g. something required by a future source of aircraft tracking data. Such changes need only be made in the one or two routines directly affected, and the rest of the program can remain untouched.

The logical interconnections among the various subroutines are diagrammed in Figures Al through A6. In these diagrams, a higher subroutine or group of subroutines may call a lower subroutine or group through any downward going connecting path, but control passes upward only as a return to a calling program.

The individual subroutines fall into classes as given in Figure Al. Under control of the MAIN program SONBOM, TRAPS first calls the Input Reader Routines to read the information from the physical data sets RAOB, WINDS, and TRACK and the first part of the CONTROL data set (see Appendix B). The Input Reader Routines are responsible for some checking of the data, converting the physical units into the units internal to the program (S.I. metric units), and performing other pre-processing tasks.

The Ray Tracing Routines are then called to emit acoustic rays from the aircraft track and to trace them, guided by the atmosphere data and the ray parameter cards from the CONTROL File (see Appendix B). On encountering the ground at the selected carpets, these routines write pertinent information (location, ray tube area, age(s), phase normal directions, etc.) on temporary files for further processing.



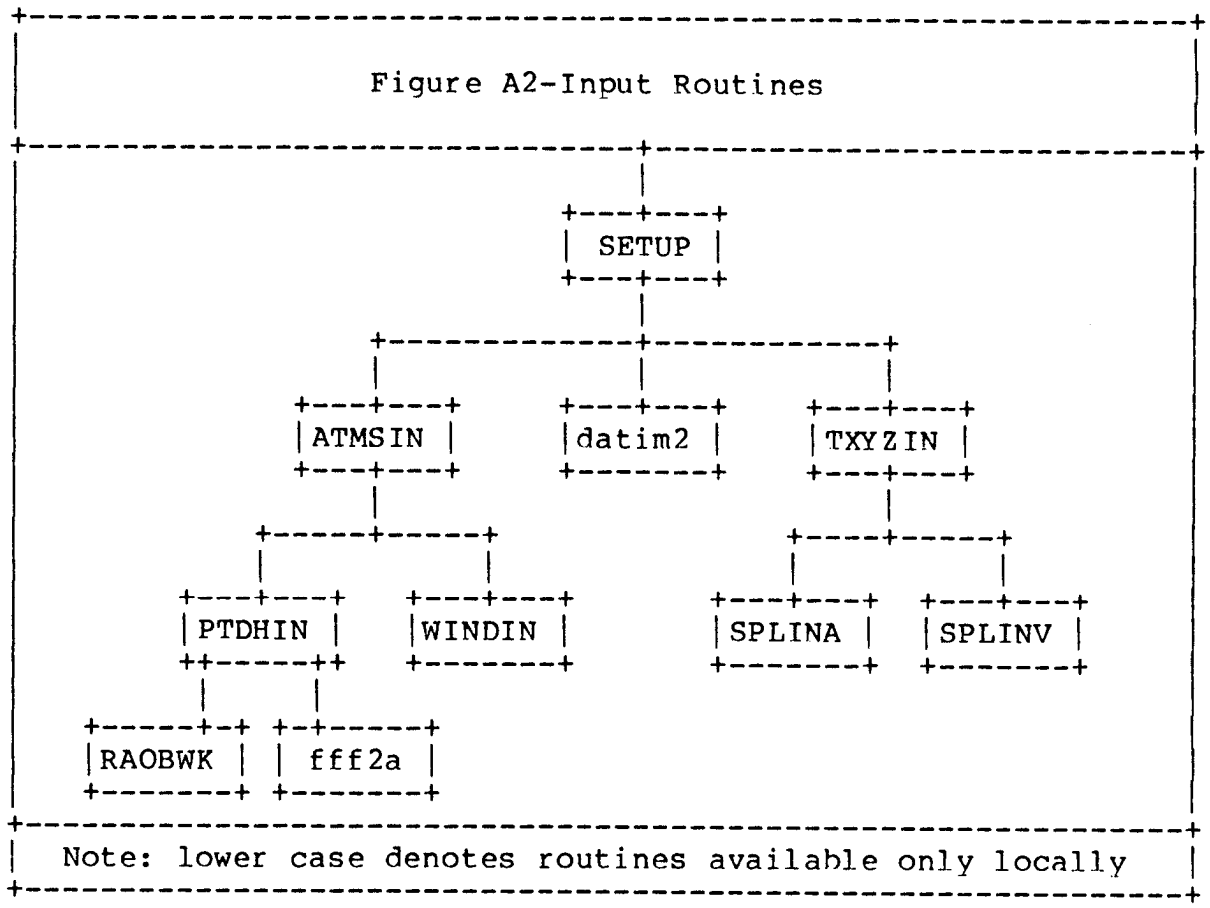

When all rays have been traced, the Signature Calculation Routines are called to analyze the information on the temporary files, together with the information on the F_FUNCTION File (see Appendix B) and compute and print the pressure "signature" (over pressure as a function of time or distance along the ground, as seen at the position or instant of ground contact). If desired only the location and over-pressures of any shocks, or the minimum and maximum overpressures will be printed.

The General Utility and Physical Utility Routines are subprograms which perform specific services which are used at several different parts of the program, so that it would be impractical to show explicitly the lines of control to them. Those that perform a service related to computer functions rather than to the physics of the problem are in the General Utility class; those which perform a service which is related to the physics fall in the Physical Utility class.

## Input Reader Routines

The Input Reader function is performed by the following routines:

SETUP -performs overall control of the input reading routines. Reads and interprets the aircraft identification and ray type requests from the CONTROL File.




ATMSIN-performs overall control of the routines written to input atmospheric data. Merges the results of PTDHIN and WINDIN and a pre-selected set of altitudes at which ray-trace output is wanted. Uses the above to create a single overall data table for use by subroutines AIR and RAYTRK.

PTDHIN-reads the RAOB File. Converts all data into S.I. units, interpolates dewpoint data as needed and calculates virtual or molecular scale temperatures* from the temperature and dewpoint data. Supplies hydrostatically valid height or pressure data, as appropriate, returns a table of virtual temperatures, pressures, and heights. Prints out all input data, together with the calculated pressure and height information in original units for comparison with other sources.

---

* The virtual or molecular-scale temperature is the temperature at which dry air of mean tropospheric chemical composition would have the same pressure-density relationship as the actual air. It is the appropriate temperature for calculating both thicknesses and sound speeds.

RAOBWK-called by PTDHIN to "work up a RAOB"; i.e., to calculate from the given temperatures and pressures the "thicknesses" (i.e. height of the column of air between each pair of pressure levels) and then, by keeping a running total of "thicknesses", calculate heights. Conversely, given thickness, calculate pressure drop.

WINDIN-called by ATMSIN to read the WINDS File and convert to SI units. Produces an internal table of wind speeds, directions, and "turning rates"; i.e. the rate of direction change with height between the levels in the WINDS File. The turning rate is provided to assist AIR in linear interpolation of wind direction; it has the sign and magnitude to cause the smallest rate of direction change meeting the given directions. Where the wind speed is zero on one side of a layer, the turning rate is taken to be the same as that of an adjacent layer. The routine also prints out the speed and direction data in the original units for documentation.

TXYZIN-called by SETUP to read the TRACK File, convert to internal units, and smooth. TXYZIN smoothes by a process of computing accelerations appropriate to a cubic spline fit through the initial data, applying a smoothing filter to the accelerations, and adjusting the original points to conform to a spline with those accelerations. The subroutine prints out the original coordinates and the altered coordinates, both in the original units, and the accelerations in units of g's.

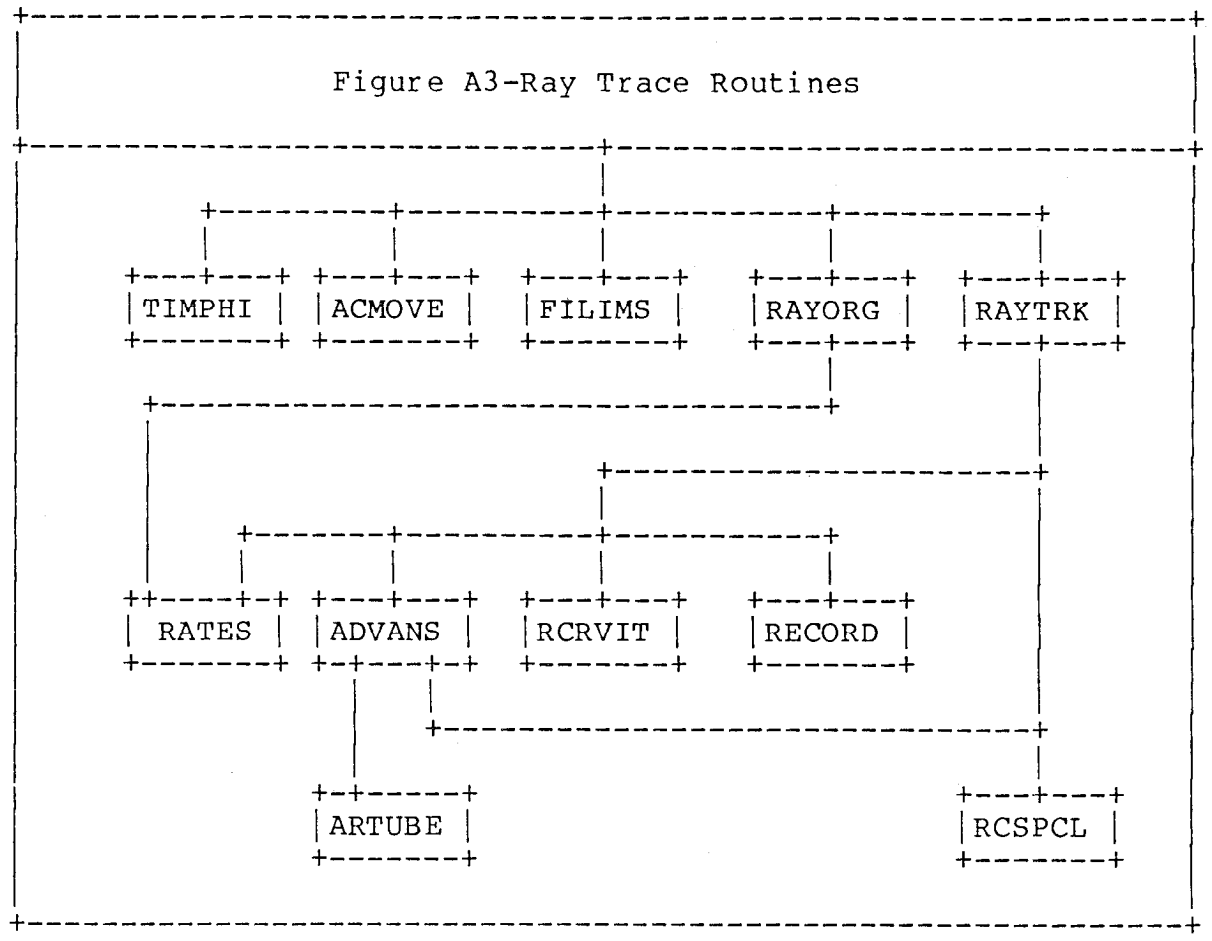
SPLINA-called by TXYZIN to compute accelerations, given coordinates, of a cubic spline. Follows procedure in [Ahlberg et al:1967], assuming accelerations at first and last points are zero.

SPLINV-called by TXYZIN to compute coordinates, given accelerations, of a cubic spline. Inverts procedure in [Ahlberg et al:1967], assuming coordinates of first and last points are as originally read in.

## Ray Tracing Routines

The function of emitting and tracking rays from aircraft to ground is performed by the Ray Tracing routines, under control of the MAIN program SONBOM. The operation of these routines is as follows:

TIMPHI-reads the next card from the CONTROL File and interprets it as a sequence of emission times and a sequence of $\phi$ angles. When CONTROL File is empty, makes non-standard return to SONBOM.




ACMOVE-interpolates aircraft track spline to current value of emission time. Computes and stores in COMMON block the position and velocity of the aircraft, the local sound speed and wind, the airspeed and its rate of change, the Mach number and its rate of change, the climb and bank angle and the wing loading, the direction cosines of a "ray cone coordinate system" and their rates of change. Prints out the information on the aircraft position and motion, both in an airborne reference frame and a ground reference frame.

FILIMS-given the information from the ACMOVE subroutine, and the wind velocity and sound speed at the ground, computes the limits of $\phi$ angle at the admittance ellipse (see text) for the ground level. Prints out the limiting $\phi$ angles for the arcs inside the admittance ellipse, if any.

RAYORG-for each emission time and for each value of $\phi$ lying within the admittance ellipse, computes the initial values of position, ray normals, "frequencies", and their rates of change. Sets current time equal to emission time. The rates of change are with respect to not only current time, but also the ray parameters of $\phi$ angle and of emission time. If ray trace printing is selected, prints out the initial ray trace values.

RAYTRK-from the initial values supplied from RAYORG, traces the ray to the ground level and reflects as many times as necessary. Controls the computation of the change in not only the position of the ray, but associated terms such as the ray normals, the ray tube area terms, and the age(s). If ray trace printing is selected, also prints a record of position, ray tube area, and time at selected altitudes.

RATES -computes the local rate of change of the ray position, the ray normals, and the associated derivatives with respect to the ray parameters $\phi$ and emission time.

ADVANS-utilizes information from RATES to compute advance in current time, and the change in ray position and associated variables corresponding to it.

RCRVIT-when a tentative advance brings ray beyond a reversal layer, will locate the exact position of the reversal layer.

RECORD-when the ray has been traced to ground in a selected carpet, will record the location and all the associated variables required to compute signatures on a temporary file (FORTRAN unit 9).

ARTUBE-computes the Jacobian, defining the ray tube area.

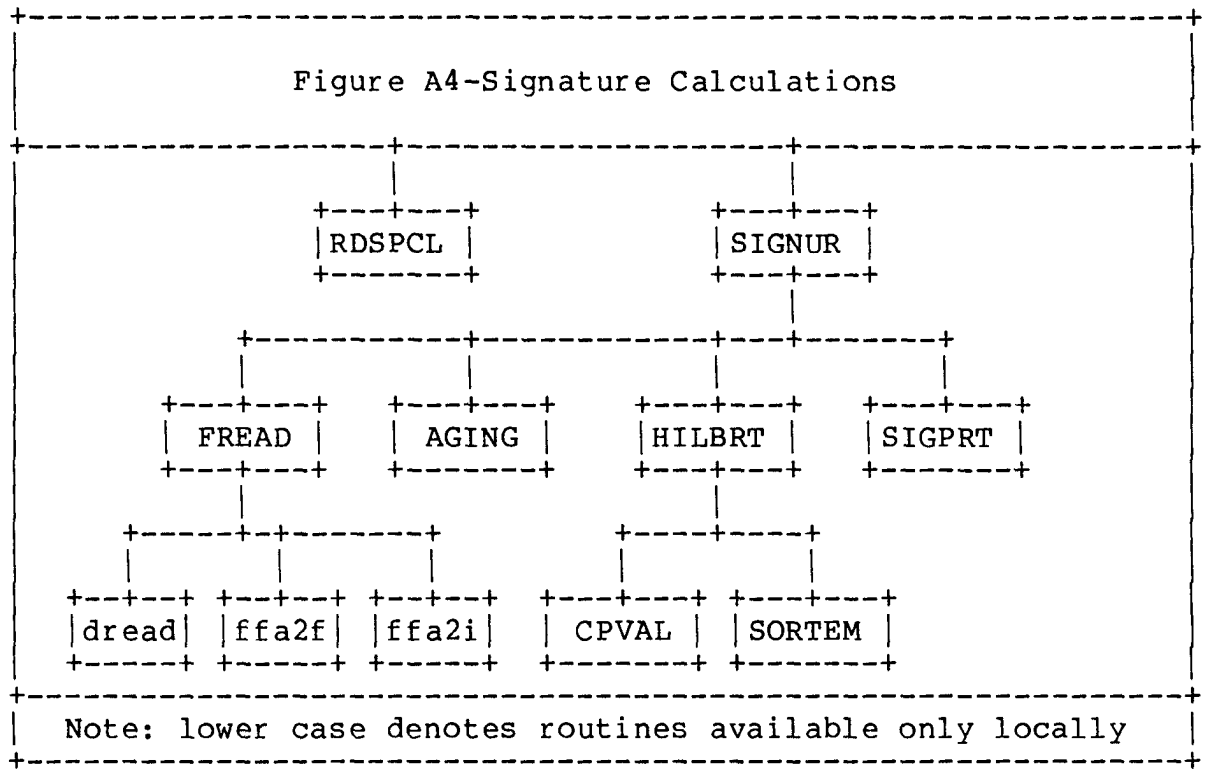
RCSPCL-records on a temporary file (FORTRAN unit 11) the positions and times for each "special point" in the ray's path. "Special points" include reversal layer encounters, ground encounters, and the encounters with the caustic surfaces.

## Signature Aging Routines

After all rays have been traced, it is the task of the Signature Aging Routines to perform the final calculations and determine the actual overpressures to be expected. They operate as follows:

RDSPCL-This routine is actually between the ray tracing routines and the signature calculations per se. It lists all the special points recorded by RCSPCL.

SIGNUR-has overall control of the aging and printout process. For each ray terminus recorded by RECORD, it reads, interprets and prints out the information on ray type, Mach number of aircraft, initiation time and $\phi$ angle, location, elevation and azimuth of the ray normals, and the conversion factors from F-function normalized coordinates to time (TFACT) and pressure (PFACT). It combines the F-functions according to this information and controls the evolution of the signature.




FREAD -determines whether the necessary F-function tables are in main memory, and if not, reads them into main memory.

AGING -shifts the abscissa values (phase) of the F-functions according to the age value, determines the total area of the resulting figure, and fits jump discontinuities as appropriate. Replaces the input F-function with the result.

HILBRT-has overall responsibility for calculating the Hilbert Transform. Replaces the input F-function, as modified by AGING and possibly containing shocks, by its Hilbert transform. Computes the transform at a selection of points determined by the overall structure of the function. This includes a set of points exponentially converging to each shock (terminating within a distance of the shock equal to $6*10^{-7}$ times the overall scale of the input F-function). it also includes a set of points which are centered on the mean abscissa value of the input F-function and which are spaced at increasing increments to cover an interval several times the abscissa scale of the input F-function.

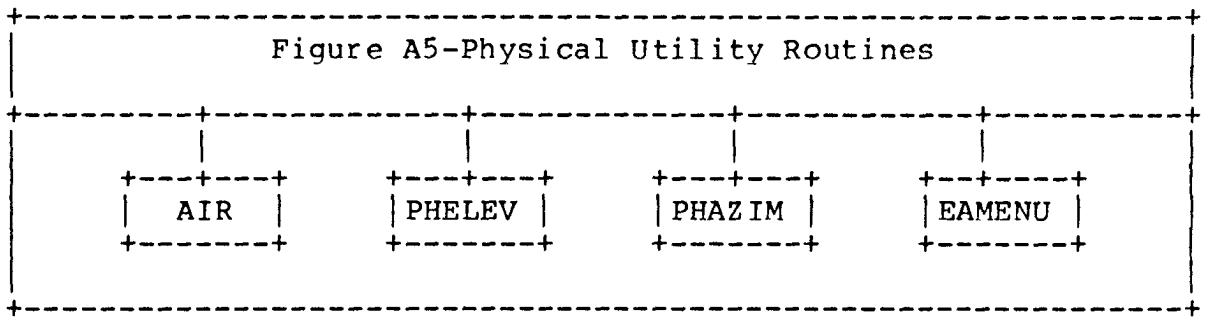
CPVAL -computes the value of the integral defining the Hilbert Transform, as a Cauchy Principal Value, at each point directed by HILBRT.

SORTEM-sorts the values calculated by HILBRT and CPVAL into ascending order of abscissa values, as required by AGING.

SIGPRT-prints out the final signature, as directed on the CONTROL File cards.

Physical Utility Routines

The Physical Utility Routines are called from many of the subroutines listed above to perform tasks




related to the physics of the problem. These routines are:

AIR -called to produce, at a specified altitude within a specified layer, the values of the sound speed and wind velocity, the first and second derivatives of those quantities with respect to height, and the density of the atmosphere. Uses linear interpolation of wind speed, wind direction, virtual temperature, and $\gamma$ with respect to geopotential height; the other quantities are derived from algebra and a hydrostatic assumption.

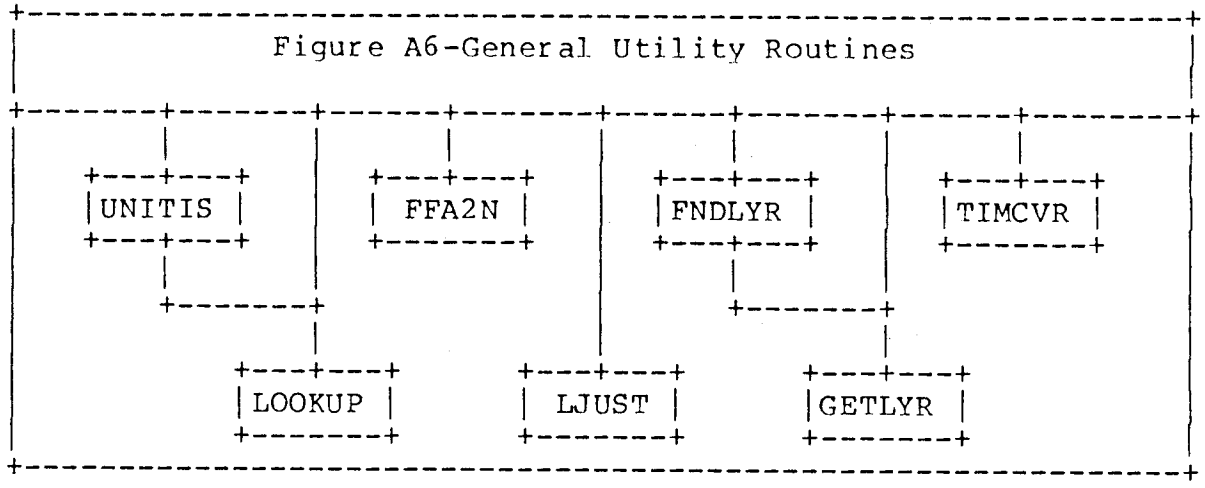
PHELEV-given the components of the wave-number vector, calculates the elevation angle of the normals to the phase surfaces of the wave.

PHAZIM-given the components of the wave-number vector, calculates the azimuth angle of the normals to the phase surfaces of the wave.

EAMENU-given the elevation angle, azimuth angle, and magnitude of a vector, calculates the east-, north, and upward components of that vector.

## General Utility Routines

The General Utility Routines are called from many of the subroutines listed above to perform tasks related to character string manipulation, list searches, and unit conversions. These routines are:




LJUST -given a character string(s) of specified length, eliminates all special characters other than digits or letters, left-justifies the result, and places it into new string(s).

FFA2N -given a character string(s) of specified length, determines if string is numeric characters, blank, or other. if numeric, converts character string to floating point binary number. If blank, selects default value from table. If other, returns error code.

LOOKUP-given a left-justified character string and a pre-sorted table of such strings, determines if character string matches a table entry or is a non-ambiguous abbreviation for such an entry. Returns index of table entry if a match exists, otherwise returns one code for ambiguous abbreviation and another for no match.

UNITIS-given a character string for unit type, a table of possible unit names, a default unit index, and a character string, determines the appropriate unit index or prints appropriate error message. Uses LOOKUP.

TIMCVR-if TRACK File chose HHMMSS units, converts hhmmss time units to seconds and vice-versa. Otherwise, leaves time units unchanged.

GETLYR-given a numeric value and a pre-sorted table of numeric values, performs a binary table search to determine between which two table entries the given value is located. If given numeric value not covered by the table, performs non-standard return.

FNDLYR-defines location of layer in atmosphere in which a given altitude is located. Sets numeric variables to top and bottom of layer. Called just prior to calling AIR by all routines except RAYTRK, which manages layer definition for itself. Uses GETLYR.

## Data Preparation for the TRAPS Program

The program for Tracing of Rays and Aging of Pressure Signatures (TRAPS) requires five input data sets. These are:

(1) COMMAND File - specifies which type of output is required and for which sets of rays.
(2) RAOB File - Pressures, Temperature, and Dewpoint as a function of Height
(3) WINDS File Wind profile as a function of Height
(4) TRACK File Aircraft position and height as a function of time
(5) F-FUNCTION File - normalized pressure signatures as experienced by an observer near the aircraft. They serve as initial conditions for the calculation of overpressures at ground level.

Since each of these types of data represents a distinct class and source of data, each is read on a separate FORTRAN 1/0 "unit".

The F-FUNCTION data set represents a special case. Since it depends principally on the geometry of the aircraft and only slightly, if at all on ambient atmospheric conditions or aircraft speed (within the speed range we are concerned with), it is the least likely to change from one run to another. As a result, it is given a distinct data format and is intended to be a permanent data set, residing as a direct access file on a disk pack. We will deal with it in greater detail later.

The remaining data sets are expected to be in the form of cards or card images of 80 character records, of which only the first 72 represent data. Columns 73 through 80 are free for other uses, such as comments or sequence numbers. In all cases, the first card or two is used as title information, and the remainder is expected to be in column format.

In column format, each card is divided into fields or columns of 8 characters width each. Data entered into these columns are either numeric values or keywords. Whether an item is a keyword or numeric, interspersed blanks are ignored as long as the entire datum lies within its 8-character field. A completely blank field will be interpreted as a missing datum, rather than a zero. Treatment of missing data will depend on the type of datum.

Numeric data will consist of an optional sign (+ or -) and one or more digits, with decimal point. If no decimal point is included, the value will be assumed to be an integer. "E" and "D" formats are not supported.

In keyword entries, all special characters, such as periods and hyphens are ignored; only letters and digits are considered.

Table B.1

| Physical Quantity | Physical Units Keyword | Physical Units Description |
|---|---|---|
| Pressure | PA,NSM | Pascals (Newtons Per Square Meter) |
| | KPA | KiloPascals |
| | MB | Millibars (100 Pascals) |
| | PSF | Pounds(force) per square foot |
| | PSI | Pounds(force) per square inch |
| Temperature | C | Degrees Celsius |
| | F | Degrees Fahrenheit |
| | K | Degrees Kelvin |
| | R | Degrees Rankine |
| Altitude | METERS | Meters |
| | GPM | Geopotential Meters (1) |
| | GMM | Geometric Meters |
| | FT | Feet |
| | GPFT | Geopotential Feet |
| | GMFT | Geometric Feet |
| Length (Horizontal) | METERS | Meters |
| | KM | Kilometers |
| | SMI,MILES | Statute miles |
| | NMI | Nautical Miles(2) |
| | FT | Feet |
| Speed | FPS,FTPS | Feet per second |
| | KNOTS,KT,NMPH | Nautical Miles(2) per hour |
| | MPS | Meters per second |
| | MPH,SMPH | Miles per hour |
| | KPH | Kilometers per hour |
| Time | HHMMSS | Hours,Minutes,Seconds |
| | SSSSSS | Seconds |
| Mass | GRAMS,GM | Grams |
| | KG | Kilograms |
| | POUNDS,LB | Pounds(mass) |

Notes:

(1) A geopotential meter is the height required to increase the gravitational potential of a unit mass by the same amount as a height of one meter at sea level. This is ordinarily larger than a geometric meter.

(2) The International Nautical Mile of 1852 Meters is assumed. This differs from the British Admiralty Nautical Mile of 6080 ft. and from the pre-1954 U.S. Nautical Mile of 6080.21ft.

The data sets for physical data (RAOB, WINDS, and TRACK) each have the following general format:

Card 1-- Title card
contains 72 characters of titling information.

Card 2 -- Units card
for each data field in the data set, provides the physical units in which that datum is expressed. Entry may be keyword or unique abbreviation thereof, appropriate to the physical quantity, as taken from Table B.1.

Card 3 - Card nn -- value cards
contains numeric values for each level or time and for each field.

cf. Figures B.2, B.4, and B.6.

RAOB File - Pressure,/Temperature./Heiqht Profile

At many points in the program, values are required for temperature, pressure and density at various heights, as well as their first and second derivatives with respect to height. This information is provided through a hydrostatically consistent form of interpolation within an internal table. The information in this table is supplied from the U. S. Standard Atmosphere Table, 1976 ([COESA:1976], referred to below as SAT76), or from user-supplied data, or both, and is controlled by the data set linked to FORTRAN unit 10. On IBM 360/370 computer systems using OS, this is the data set defined by DDNAME FT10F001.

If this data set is empty or if the first card contains the keyword "STANDARD" in columns 1-8, then the SAT76 table will be used. Otherwise, the user may provide data from rawinsonde observations or from rocketsondes using the format in Figure B.1:

Figure B.1
Data Format for RAOB File

| | | | | | |
|---|---|---|---|---|---|
| card 1 | up to 72 characters for title | | | | |
| card 2 | pressure unit | temperature unit | temperature unit | altitude unit | |
| card3-cardnn | <pressure> <values> | temperature values | <dewpoint> <values> | <height> <values> | |
| | <END> | | | | |

Note: the brackets <> denote fields which are optional (see below).

The keyword parameters for units in card 2 may be selected from Table B.1. If omitted (blank), default units will be assumed as follows:

Pressure --- MB

Height --- Geopotential Meters
(Heights assumed Geopotential unless specified Geometric)

Temperature/Dewpoint--- if a unit is provided for only one of these, the same unit will also be used for the other. If both are missing, Celsius will be assumed.

Numeric data from the rawinsonde or rocketsonde observations must be entered according to the following rules:

(1) Data must be entered in increasing order of altitude.

(2) At all levels, temperature must be supplied, and either pressure or height.

Figure B.2
Example of data for RAOB File

```
RAOB JFK-CHH-KWAL COMBINED 0000Z, FEB 11,1979
MB      C       C       GPM
1022.   -9.5    -20.5   8.
1000.   -11.1   -20.1
850.    -23.1   -24.6
718.    -31.5   -36.2
700.    -30.3   -36.3
540.    -27.7   -43.7
500.    -31.5   -45.5
433.    -39.3   -51.3
400.    -41.3
300.    -48.5
250.    -48.3
200.    -45.3
175.    -43.3
150.    -46.9
135.    -46.3
100.    -54.5
80.     -50.1
70.     -54.1           18250.
50.     -53.7           20410
30.     -52.3           23710.
20.     -50.3           26340.
15.     -46.3
5.      -42.7           35555.
2.      -30.6           41695.
1.      -20.2           47025.
.4      -18.3           53690.
        -14.            55511.
        -7.             57476.
        -18.            59439.
        -20.            61401.
        -22.            63362.
END
COMMENT -- SAMPLE OF RAOB DATA
```

Dewpoint is optional, but should be entered if available.

(3) For at least one level, both pressure and height must be supplied.

(4) The input reader will not read beyond the optional end statement; subsequent cards may be used for documentation if desired.

(5) There is an upper limit of 79 levels allowed for RAOB input.

(6) Dewpoint must never be greater than Temperature.

The above rules are designed to follow as closely as possible the nature of actual measurements made in practice. For radiosondes, these are pressure, temperature, and dewpoint; height is not measured but separately calculated. By contrast, rocketsondes normally report only height and temperature. One of each type of sounding may be combined to form a single data set, using pressures from one and heights from the other.

## Action taken by Input Reader Routine

The input reader routine handles the data in essentially* the same way that a RAOB is "worked up". First, a virtual temperature is calculated from the temperature and dewpoint information at each level. This is the temperature at which dry air would have the same pressure-density dependence as the actual air. Starting from the lowest level at which both temperature and pressure are provided, the routine proceeds to calculate heights of adjacent levels where pressures are known, or pressures of those levels where only heights were given. Except for the starting level, if both pressure and height are given for a level, the height is ignored but will be printed out with the calculated height for purposes of comparison. When differences of more than 100 meters are found, the input data should be rechecked.

If all dewpoints are missing at and above a certain level, then the air will be considered dry at and above that level. Otherwise, if there are dewpoints supplied above and below but not at a given level, dewpoint information will be interpolated. If there is dewpoint information above but not below or at a given level, the lowest available dewpoint will be used.

For altitudes above those for which the user has supplied data, heights and temperatures will be provided from SAT76. Pressures are then calculated from the nearest user-supplied datum level in the same manner as if these were user-supplied height/temperature data. The top limit to this interpolated data is 130,247 gpm. Instead of switching to a smooth profile above 86km as in SAT76, we continue to apply the conventions in force at lower altitudes; namely, linear temperature dependence on geopotential height within each layer. We have approximated the curved profile above 86km in SAT76 by a sequence of closely spaced height/temperature values.

Below the user supplied data, we have inserted an extra pressure-temperature point from SAT76, corresponding to the entry at 5000gpm below sea level. This will provide automatic interpolation below the user-supplied data if needed.

---

* There is a slight difference in that we use an interpolation algorithm for temperature which is linear with geopotential height rather than pressure. Although this agrees with SAT76, it is not general practice. Usually data points are chosen so closely that the difference is small.

WINDS File - Height/Direction/Speed Profile

Wind speeds and their derivatives with respect to height have a considerable effect on the ray trajectories and the resulting overpressures. Although winds are usually determined on the same balloon flights as the information given in the RAOB section, they are not generally reported at the same levels or in the same terms. Accordingly, the wind data, if any, will be supplied by the user on a separate data set linked to FORTRAN unit 15. On IBM 360/370 computer systems using OS, this is the data set defined by DDNAME FT15F001.

If this data set is empty or if the first card contains the keyword "NOWINDS" in columns 1-8, then the air will be assumed to be calm; a wind speed of zero will be used throughout. Otherwise, the user may provide data from rawinsonde observations or from rocketsondes using the format in Figure B.3:

Figure B.3
Data Format for WINDS File

| | | | | |
|---|---|---|---|---|
| card 1 | up to 72 characters for title | | | |
| card 2 | height unit | DEG | speed unit | |
| card3-cardnn | height values | direction values | speed values | |
| | <END> | | | |

Note: the brackets <> denote fields which are optional.

The keyword parameters for units in card 2 may be selected from Table B.1. If omitted (blank), default units will be assumed as follows:

Height -- Geopotential Meters
(Heights assumed Geopotential unless specified Geometric)

Direction -- Degrees (Note- units keyword will be ignored since the only unit allowed is degrees from which the wind is blowing).

Speed—Knots

The following rules hold for numeric data entry in this data set:

(1) Data must be entered in increasing order of altitude.

(2) Missing data will be assumed to be zero. Thus, Height data should never be missing.

Figure B.4
Example of data for WINDS File

```
WINDS JFK-CHH-KWAL COMBINED 0000Z, FEB 11, 1979
GMM     DEGREES  KNOTS
8.      310.     15.
119.    305.     22.
1219.   305.     30.
2439.   320.     43.
4881.   285.     94.
5796.   285.     106.
7018.   275.     79.
10380.  275.     73.
13277.  270.     56.
14970.  285.     37.
15981.  265.     37.
20410.  290.     8.
22000.  230.     2.
23710.  080.     13.
26340.  090.     24.
30000.  060.     14.
33000.  055.     17.
35555.  080.     49.
41000.  030.     16.
41695.  270.     17.
42000.  330.     16.
47025.  350.     17.
53690.  250.     43.
56000.  255.     103.
58000.  250.     140.
60000.  260.     146.
62000.  265.     144.
64000.  260.     148.
65000.  260.     144.
68000.  260.     81.
70000.  260.     81.
72000.  280.     79.
END
SAMPLE OF WINDS DATA
```

(3) The input reader will not read beyond the optional "END" statement; subsequent cards may be used for documentation if desired.

(4) There is an upper limit of 79 levels allowed for WINDS input.

(5) There is no need for the heights of levels in the WINDS File to be the same as in the RAOB File.

## Action taken by Input Reader Routine

The input reader routine reads the user-supplied data and converts them to internal units. In addition, it adds extra levels at geopotential altitudes of 5000 meters below sea level and 130,274 meters above, at which the winds are zero.

The program will interpolate the wind speeds and directions linearly with respect to geopotential height. This was considered more realistic than the alternative of interpolating the wind components, although the interpolation of wind directions poses special programming problems. In each layer between two levels for which the wind speed is non-zero, so there exists a definite wind direction at both ends, a rate of direction change with height is chosen for which the total direction change over the layer is no more than 180 degrees. If a layer is bounded by a level at which the speed is zero, so there is no definite direction at that end, no turn rate can be calculated. In such a case, there may be an adjacent layer for which a turn rate does exist; if so then that turn rate will be used. If not, then a turn rate of zero will be used.

After the input reader has processed the RAOB and WINDS files, it combines them into a single table, using all the levels from each file. This table is used to supply meteorological data to the main program on demand.

TRACK File - Aircraft Position/Height vs Time

The program requires data on the aircraft position and speed as a function of time for initial conditions for the ray tracing. In addition, information on accelerations is important for the computation of ray tube areas and hence overpressures. The program requires the user to supply horizontal coordinates x and y and altitude z, either from radar readouts or aircraft navigation system records, at closely spaced points in time. From this, and the meteorological data, it can calculate ground and air speeds, Mach numbers, azimuth and climb angles, and if perfectly coordinated turns are assumed, bank angles. Because accelerations are computed and used, it is important for the user to use data that are as accurate and precise as practicable. The TRACK File is linked to FORTRAN unit 20. On IBM 360/370 computer systems using OS, this is the data set defined by DDNAME FT20F001.

Figure B.5
Data Format for TRACK File

| | | | | | |
|---|---|---|---|---|---|
| card 1 | up to 72 characters for title | | | | |
| card 2 | time unit | x-coord unit | y-coord unit | height unit | |
| card3-cardnn | time values | x-coord values | y-coord values | height values | |
| | <END> | | | | |

The keyword parameters for units in card 2 may be selected from Table B.1. If omitted (blank), default units will be assumed as follows:

Time --- HHMMSS

Note: this unit allows data entry according to 24-hour clock time. Thus 32.7 seconds after 3:09PM is entered as 150932.7.

X,Y coords---If a unit is provided for only one of these, the same unit will also be used for the other. If units are not provided for either, NMI will be assumed.

Height--GMFT

Note: Geometric heights assumed unless user specifies otherwise.

Figure B.6
Example of data for TRACK File

```
CONCORDE FLIGHT FEB. 8, 1979
HHMMSS     N.MI.    N.MI.   GMFT
020409.    297.7    189.6   41800.
020415.    296.7    189.3   41800.
020421.    295.8    189.0   41200.
020427.    295.0    188.7   40500.
020433.    294.0    188.3   40500.
020439.    293.2    188.1   39900.
020445.    292.2    187.8   39900.
020451.    291.3    187.6   39500.
020457.    290.6    187.3   39200.
020503.    289.6    187.0   39200.
020509.    288.7    187.0   39167.
020515.    287.8    186.3   39133.
020521.    287.1    186.2   39100.
020527.    286.5    186.0   38900.
020533.    285.5    185.7   38900.
020539.    284.8    185.5   38900.
020545.    284.1    185.3   38900.
020551.    283.2    185.1   38900.
020557.    282.6    184.8   39000.
020603.    281.7    184.6   39000.
020609.    281.1    184.5   39000.
020615.    280.3    184.2   39000.
020621.    279.6    184.0   39000.
020627.    278.8    183.7   39000.
020633.    278.2    183.6   39000.
020639.    277.3    183.3   39000.
020645.    276.7    183.1   39000.
020651.    275.8    182.8   39000.
020657.    275.1    182.6   39000.
020703.    274.3    182.3   39000.
  END
SAMPLE TRACK DATA
```

The following rules hold for numeric data entry in this data set:

(1) Data must be entered in chronological order. For HHMMSS input, chronological order is forced by adding 24 hours; thus 000130. following 235845. is interpreted as 240130. i.e. 2 min, 45 sec later.

(2) Missing data will be assumed to be zero. Normally, data should never be missing.

(3) The input reader will not read beyond the optional "END" statement; subsequent cards may be used for documentation if desired.

(4) A minimum of 2 and a maximum of 100 position fixes are allowed for TRACK input.

Action taken by Input Reader Routine

The input reader routine reads the user-supplied data and converts them to internal units. In addition, it smoothes the data so that reasonable values of speed and acceleration can be provided to the main program.

The cubic spline algorithm which yields the interpolated positions, speeds and accelerations to the main program requires an internal table of accelerations especially calculated to fit all cubic polynomial segments smoothly together. However, the uncertainties in position (0.1 NMI in Figure B.6) at close time intervals (6 sec in Figure B.6) leads to acceleration uncertainties of as much as 1 g. To avoid these oscillations, a digital filter is applied to the accelerations from the cubic spline algorithm. Then, since the smoothed accelerations do not yield a spline fit through the original points, a new set of positions is calculated for which the smoothed accelerations do yield a spline fit. This is done by fixing the original end points and then reversing the usual cubic spline process.

This smoothing process minimizes the possibility of understating real accelerations as a digital filter applied to the original data might do, as well as ensuring consistent terms for the spline interpolations in the main program. The input reader prints out both the original and smoothed coordinates for comparison purposes. If the smoothed coordinates differ sharply (more than a few times the uncertainty in the original coordinates) from the input data, the original data should be carefully checked.

Control of the TRAPS program is provided through ray tracing requests included in the COMMAND data set. These requests are provided in the form of card images on the data set linked to FORTRAN unit 5. On IBM 360/370 computer systems under OS, this is the data set defined by DDNAME FT05F001.

The format for COMMAND data is given in Figure B.7:

Figure B.7
Data Format for COMMAND File

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| card1<br>card2 | up to 72 characters for title<br>up to 24 more characters | | | | | | |
| card3 | a/c ident | a/c weight | weight units | | | | |
| card4 | ground height | height units | reflect factor | | | | |
| card5.1<br>-<br>card5.n | ray class codes | and print codes | ... | ... | ... | ray class codes | and print codes |
| card6.1 | begin time | end time | time incr | begin phi | end phi | phi incr | trace code |
| . | | | -OR- | | | | |
| . | | | | | | | |
| . | | | | | | | |
| card6.n | begin time | end time | time incr | margin code(s) | total phi | phi incr | trace code |

The aircraft identification in card 3 should match the identification of one of the F-Functions in the F-Function data set. In comparing identifications, interspersed blanks and hyphens may be introduced for legibility; they will be ignored.

Weight and height units in cards 3 and 4 should be selected from Table B.1; if not supplied they will default to LB and GMM, respectively. Aircraft weight in card 3 must be supplied; ground height and reflection factor in card 4 will default to zero and one, respectively.

Cards 5.1 through 5.N specify to the program the types of ray that are to be tracked, and how many reflections to be followed (RAY CLASS), as well as the amount

and type of print-out. The codes allowed are specified in Table B.2.

Table B.2
Codes used in COMMAND File

RAY CLASS Codes:

| | | | |
|---|---|---|---|
| G | GMG | GMGMG | GMGMGMG |
| MG | MGMG | MGMGMG | |
| GHG | GHGHG | GHGHGHG | |
| HG | HGHG | HGHGHG | |

PRINT Codes:

| | |
|---|---|
| FULL | : Print all signature overpressures |
| SHOCKS | : Print shocks only |
| SUMMARY | : Print summary of maximum overpressures |
| NO | : Do not print signatures |
| END | : End of RAY CLASS and PRINT codes |

MARGIN Codes:

| | |
|---|---|
| UL | : trace rays from left of upper cone segment |
| UR | : trace rays from right of upper cone segment |
| LL | : trace rays from left of lower cone segment |
| LR | : trace rays from right of lower cone segment |
| AL | : trace rays from both sides of both segments |
| NO | : do not trace rays |

TRACE Codes:

| | |
|---|---|
| PRINT | :Print full tracing of rays |
| NOPRINT | :Do not print tracing of rays |

To interpret the ray class codes, note that in general the ray may reflect from the ground (G), or an upward moving ray may refract downwards after reaching a height near the stratopause (M - app. 50km) or the thermosphere (H - app. 100km). The G, M and H codes are concatenated to indicate the order in which the ray encounters each level. Thus, a GHG ray would leave the aircraft moving downward, reflect from the ground, refract from the thermosphere and impact the ground again, at which point the pressure signature and position are recorded. An MG ray would leave the aircraft in an upward direction, refract from the middle level and impact the ground, at which point its signature and position are recorded.

Note that the stratification assumptions ensure that if a ray refracts from the H layer, neither it nor its reflections can refract from the M laver, or vice versa, so that mixed rays of type GHGMG or HGMG can never occur and are not valid codes.

Print codes for cards 5.1 - 5.N refer to the amount of detail desired for the signature listings. FULL (the default) causes the entire signature (overpressure as a function of time) to be listed for the terminus of each ray being examined. SHOCKS causes only the overpressure before and after each shock, and the relative time of each shock, to be listed. SUMMARY causes only the maximum and minimum overpressure of the signature to be listed, in a summary listing of the positions of the termini. NO causes no printout of overpressure signature information.

As many ray class and print codes as are required may he supplied, at 9 fields to a card, until either a blank field or a field containing the code END is encountered, at which point processing of code requests will cease.

Cards 6.1 - 6.n specify the desired origins of the rays to he traced. There are two available formats, which may be intermingled. In each format, the program is requested to initiate rays at the aircraft at discrete times, starting at BEGIN TIME, and continuing at equal time increments given by TIME INCR until END TIME, is attained. BEGIN TIME and END TIME are to be in the same time units as specified for the TRACK File, whereas TIME INCR will be in seconds.

For each originate time, the program will examine the sound speed and wind speed at the aircraft and ground heights and determine the range of PHI angles for which Snell's law prevents a ray from reaching the ground. There will be, in general, two ranges of allowable PHI angles; one for upward moving rays and one for rays initially moving downward.

In the 6.1-6.N cards, the rays to be traced are specified using fields 4 through 6. Using the first format, the program will consider rays at PHI angle values beginning with BEGIN PHI and increment by the amount PHI INCR, until the value END PHI is attained. If the PHI angle falls in one of the admitted ranges, the ray will. be traced, but if not, the ray will be dropped in order to conserve computer time.

The margins of the admitted PHI angle regions, and hence of the sonic boom carpet, may be explored through the second format. By selecting any of the margin codes in Table B.2, the program will consider rays with PHI angles beginning at either the right or left limit of the upper or the lower allowed region, and increment by an amount PHI INCR toward the interior of the region until a range of TOTAL PHI is covered. By selecting more than one code, a combination of edges may be explored. If the code NO is selected, PHI INCR and TOTAL PHI will be ignored and no rays will be traced. In this case, only the limits on allowable PHI angles would be

output. Note that units on PHI angles will always he in degrees.

The trace code (field 7) specifies whether the program is to print out a full tracking of the position and age of the ray, and the area of the ray tube associated with it, on its progress through the atmosphere. If NOPRINT is selected, the full tracing will not be output, but only the values at the ground level. If PRINT is selected (or defaulted) a full tracing will be printed.

Figure B.8
Example of data for COMMAND File

```
      BA 171 CONCORDE FLIGHT JUNE 20,1979
JFK-BOUND FLIGHT
CONCORDE 108300. KG
     0.   METERS  1.
G       MG      GMG     SHOCKS
092230  092800  30      90      450     15      PRINT
092630  092700  10        LR    25      1       NOPRINT
092700  092700  0       -28.03  -26.03  .1      NOPRINT
```

Figure B.8 contains a sample of cards for the COMMAND File for one run of TRAPS. The first two cards are title cards; they identify the run as dealing with British Airways flight 171 on June 20, 1979 bound for Kennedy International Airport. The contents of these cards will be printed at the top of all main sections of printout, together with the date of the computer run.

The third card identifies the aircraft type for matching against the F-FUNCTION File; it also gives the weight of the aircraft at this point in its flight (considering the load and the amount of fuel burned) as 108,300 kg. The fourth card identifies the signatures as wanted at sea level. It also puts the reflection factor at one, signifying the amplitude of the reflected boom to be the same as that of the incident boom; this is appropriate for booms incident on water.

The fifth card instructs the program to cover the direct or primary (G) carpet, as well as both single-M secondary carpets (MG and GMG), but that double bounces (GMGMG, etc.) and high level or thermospheric rays (HG, etc.) are not needed. In addition, the output routines are instructed that the entire pressure signature will not be needed on this run; but the positions of each shock wave and the pressure before and after the shock wave will be wanted.

Cards 1 through 5 are read during the initial setup of the program. The remaining cards are read by the ray tracing routines one at a time and acted upon. In this case, it is assumed that a

TRACK data set has been provided giving times in a HHMMSS format. The first of the ray parameter cards tells the program that rays are wanted every 30 seconds, starting at 09:22:30 GMT until 09:28 GMT. The rays will be tested every 15 degrees in phi angle, starting with 90 degrees (9 o'clock, from the pilot's viewpoint) and continuing clockwise until 450 degrees (9 o'clock, again). If the ray lies inside the admittance ellipse for the ground, it will be traced. The complete course of each ray will be printed out, giving the time and position as it crosses each of a set of pre-selected altitudes, as well as the ground level. Furthermore, the values of ray tube area and age will be printed out.

The next card requests a closer look at a portion of the track, asking for rays to be traced every 10 seconds from 09:26:30 to 09:27. Further, it asks only for rays beginning at the lower right edge of the rays in the admittance ellipse, tracing every degree in phi angle up to 25 degrees. This time, the ray traces will not be printed out.

The final card asks for an even closer look. The zero for time separation indicates that rays are only wanted from the aircraft at one time, namely 09:27 GMT. Rays are to be traced with phi angles every tenth of a degree, from -28.03 to -26.03; i.e. around 5 o'clock from the pilot's viewpoint. Ray tracings will not be printed.

This mixture of resolutions is presented only for purposes of illustration. In fact, a more likely procedure in analyzing a flight would be to make one run with a coarse resolution (30 sec. time, 10 deg phi angle), determine from the output what portions require closer examination, and then make a second run with finer resolution in the region of interest.

## The F-FUNCTION File

The F-FUNCTION File provides the program with a means of specifying the pressure signatures of the sonic boom, as seen by an observer near the aircraft and at rest relative to the ambient air. This initial form is then aged by the program according to the history of the acoustic ray being traced.

The program constructs this initial signature as a linear combination of two components; one corresponding to the area distribution of the aircraft (A) and one due to the lift distribution (L). Both are recorded in the F-FUNCTION File as normalized functions of a normalized time variable; each must be subjected to rescaling by different scales depending on the Mach number, aircraft weight and maneuver, and the relative orientation of the ray to the wing. Only after scaling may they be superimposed.

F-functions for a particular aircraft may be developed theoretically (see the discussion in [Hayes et al:1969]) or by actual measurement in a wind tunnel (c.f. [Carlson: 1964] ) . In the latter case, separate sets of F-functions may be available at different Mach numbers for the same aircraft.

Operationally, it is desirable to have the F-functions for all the aircraft and all the Mach numbers with which we are concerned on a common data base. When the aircraft maneuvers cause it to move through various Mach number regimes, it is important to be able to change F-function tables quickly, and this is facilitated by using a direct access data set.

In its present version, the TRAPS program expects its F-function data on 80-character EBCDIC card images on a direct access file. The individual records are "addressed" by their position in the file, so that the program can read a specific numbered record without reading or backspacing over intervening records. This file is linked to the FORTRAN unit 90. On IBM 360/370 computer systems using OS, this is the data set defined by DDNAME FT90F001.

The F-FUNCTION File is subdivided into Aircraft Record Sets, one for each aircraft type under consideration. Each Aircraft Record Set is in turn subdivided into as many as 10 Mach Number Record Sets, one for each Mach number represented by an F-function table for that aircraft.

The first card in each Aircraft Record Set is the Header record. It contains

(a) the type of aircraft
(b) a relative address pointer to the next Aircraft Record Set
(c) a table of the Mach numbers represented in this Aircraft Record set, and the relative addresses of the Record Set for each.

The relative address refers to the difference in address between the Header record and the record being pointed to. To obtain the actual address, the relative address must be added to the address of the Header record. The use of this addressing technique allows the relocation of the entire Aircraft Record Set, without modification, in the event that at some future time a new Aircraft Record set is inserted in front of this one, or an existing one is modified by adding new Mach Number Record Sets.

Information in the Header record for an Aircraft Record Set is formatted as in Figure B.9:

Figure B.9 - Format for FFUNC Header Record

| Columns | FORTRAN FORMAT | Datum Description |
|---|---|---|
| 1-8 | A8 | Aircraft Identification |
| 18-22 | I5 | Relative address of next Aircraft Record Set |
| 26-27 | I2 | Number of Mach Number Record Sets in this Aircraft Record Set |
| 28-68 | F5.2,I5 | Mach Numbers and Relative addresses |

The Aircraft Identification must be from 1 through 8 characters, left justified with blank fill. No embedded blanks, periods, commas, hyphens, or special characters other than letters and digits are allowed. The TRAPS program will match this identifier against that read in from the CONTROL data set.

If the relative address is zero, then this is the last Aircraft Record Set in the F-FUNCTION File. Otherwise, this is the total number of cards in the Aircraft Record Set, and hence a pointer to the card immediately following.

Beginning at column 28, in a repeated (F5.2,I5) FORMAT, appear the Mach numbers of the individual Mach Number Record Sets, and the relative address of the first card of each set. Since columns 73-80 are reserved for possible sequence numbers or comments, only four Mach number pointers can appear on the Header card. Up to two secondary Header records can appear for continuing the Mach number table information; continuation begins at column 28.

The format of the individual data records is given in Figure B.10:

Figure B.10 - Format for FFUNC Data Record

| Columns | FORTRAN FORMAT | Datum Description |
|---|---|---|
| 2-9 | A8 | Aircraft Identification |
| 10-12 | F3.2 | Mach number |
| 13-15 | I3 | Total number of terms in this Mach Number Record Set |
| 16-20 | F6.2 | Aircraft length (meters) |
| 22-28 | F7.5 | increment in $\tau$ (see below) |
| 29-31 | I3 | item number within this Mach Number Record Set |
| 35-50 | E16.9 | Area component of F-function |
| 54-69 | E16.9 | Lift component of F-function |

Aircraft identification and Mach number should match that on the Header card, and can be used for a consistency check. The aircraft length participates in the normalization for the F-functions, both area and lift, although in different manners. The variable increment in the independent variable tau allows for F-functions to be of differing resolutions, depending on sources of data or complexity of aircraft structure.

The FREAD subroutine, on first being called, searches the Aircraft Header records to locate the Aircraft Record Set corresponding to the aircraft called for on the CONTROL File. If a match is found, FREAD then proceeds to construct an internal table of available Mach numbers and their absolute locations within the F-FUNCTION File. On subsequent calls, FREAD is notified of the Mach number of the aircraft at the time the ray was emitted, and constructs an internal table of $\tau$ values, area F-function components and lift F-function components. (This step is skipped if the table is already in memory from a previous call). Supplied with information on, among other items, the difference between the $\phi$ angle and the angle of bank, the density of the air and the weight of the aircraft at the time the ray was emitted, the TRAPS program can then combine the two terms to form a single F-function.

A deck of cards may be punched in this format for each aircraft type and for each Mach number, and these decks may be stacked together. The result is a sequential data set, since the computer must read all the cards until it reaches the desired one. To obtain a direct access data set for the TRAPS program to use, the card deck must be converted through the use of a utility routine which can read the cards and write them out using Direct-Access software. An example of such a utility routine is the first program listed on the enclosed

fiche of the TRAPS program, although it properly is not a part of that program. It should be regarded only as a prototype of such a utility, since it uses Direct-Access subroutines (DWRITE and DCLOSE) that are available only at the NOAA IBM 360/195 computer site. Comparable software is available at many other sites, however, and modifying the example to fit the local software is not difficult.

In Figure B.11 is a sample listing of an Aircraft Record Set in the F-FUNCTION File:

Figure B.11 Sample of FFUNC records

```
CONCORDE NEXT AC=   0 #= 5 1.25    2 1.50  102 2.00  202 2.50  302
                           3.00  402
 CONCORDE125100 9641 .01000   1 A= 0.00000000E+00 L= 0.
 CONCORDE125100 9641 .01000   2 A= 9.30030594E-03 L=  .22507907E+00
 CONCORDE125100 9641 .01000   3 A= 1.18642882E-02 L=  .27566444E+00
 CONCORDE125100 9641 .01000   4 A= 9.27360625E-03 L=  .63661977E+00
                     .            .            .
                     .            .            .
                     .            .            .
 CONCORDE125100 9641 .01000 98 A= 3.48695260E-04 L= -.21990643E+00
 CONCORDE125100 9641 .01000 99 A= 3.29247664E-04 L= -.21334520E+00
 CONCORDE125100 9641 .01000100 A= 3.11200026E-04 L= -.20711117E+00
 CONCORDE150100 9641 .01000   1 A= 0.00000000E+00 L= 0.
 CONCORDE150100 9641 .01000   2 A= 9.53856337E-03 L=  .15915494E+00
 CONCORDE150100 9641 .01000   3 A= 1.19034874E-02 L=  .38423402E+00
 CONCORDE150100 9641 .01000   4 A= 8.88716640E-03 L=  .50074352E+00
                     .            .            .
                     .            .            .
                     .            .            .
 CONCORDE150100 9641 .01000 98 A= 3.29658782E-04 L= -.22478510E+00
 CONCORDE150100 9641 .01000 99 A= 3.10687598E-04 L= -.21797945E+00
 CONCORDE150100 9641 .01000100 A= 2.93108227E-04 L= -.21151788E+00
 CONCORDE200100 9641 .01000   1 A= 0.00000000E+00 L= 0.
 CONCORDE200100 9641 .01000   2 A= 9.14146766E-03 L=  .22507907E+00
 CONCORDE200100 9641 .01000   3 A= 8.71988512E-03 L=  .11650950E+00
 CONCORDE200100 9641 .01000   4 A= 8.7819109 1E-03 L=  .25238575E+00
                     .            .            .
                     .            .            .
                     .            .            .
```

This sample of the F-FUNCTION File does not actually contain data for the Concorde aircraft, as it purports to. During the entire project, despite repeated requests, neither the F-function data nor the aircraft dimension data to compute them were made available. The F-functions which were used were actually from a different aircraft (SR-71), scaled to fit the length, width, and height of the Concorde.

| Summary of Data Sets | |
|---|---|
| FORTRAN unit | Purpose |
| 05 | COMMAND File |
| 10 | RAOB File |
| 15 | WINDS File |
| 20 | TRACK File |
| 90 | F-FUNCTION File |
| | |
| 06 | Print file |
| 09 | Temporary file for subroutines RECORD/SIGNUR |
| 11 | Temporary file for subroutines RCSPCL/RDSPCL |

```
C./       ADD   NAME=DREAD
      SUBROUTINE DREAD(KUNIT,KRECD,BUFFER,*)
      DIMENSION BUFFER(20)
      LOGICAL OPEN
      DATA OPEN/.FALSE./
      IF(OPEN) GO TO 5
      KRET=1
      GO TO 20
    5 READ(20'KRECD,ERR=7) BUFFER
      RETURN
    7 RETURN 1
      ENTRY DWRITE(KUNIT,KRECD,BUFFER)
      IF(OPEN) GO TO 10
      KRET=2
      GO TO 20
   10 WRITE(20'KRECD) BUFFER
      RETURN
      ENTRY DCLOSE
      RETURN
   20 CONTINUE
      DEFINE FILE 90(520,80,L,NEXT)
      OPEN=.TRUE.
      GO TO (5,10),KRET
      STOP
      END
C./       ADD   NAME=SEQ2DA
      DIMENSION BUFFER(20)
      K=0
    5 READ (10,20,END=100) BUFFER
   20 FORMAT(20A4)
      K=K+1
      CALL DWRITE(90,K,BUFFER)
      GO TO 5
  100 WRITE(6,110) K
  110 FORMAT(1X,I5,' RECORDS COPIED')
      CALL DCLOSE(90)
      STOP
      END
```

---

```
C./       ADD   NAME=SONICBOM
C  *****************************************************************
C  ****  T.R.A.P.S. - SONIC BOM MODELING PROGRAM                  ***
C  ***    T.RACING R.AYS AND A.GING P.RESSURE S.IGNATURES         ***
C  ***   (SEE NOAA TECHNICAL MEMORANDUM ERL ARL-87)               ***
C  *****************************************************************
C  ******************************************
C  ***  DR. ALBION D. TAYLOR,                 ***
C  ***  NOAA/AIR RESOURCES LABORATORIES R/E/AR ***
C  ***  RM. 921, GRAMAX BUILDING              ***
C  ***  8060 13TH STREET                      ***
C  ***  SILVER SPRING, MD 20910               ***
C  ******************************************
C  ***  JULY, 1980                            ***
C  ******************************************
      COMMON /PRINTS/ TITLE(30),KTPSIG,CVRTIM,TIMLBL
      REAL*8 TIMLBL
      LOGICAL CVRTIM
      COMMON /RAYLIM/ NLIMS,BEG(2),END(2)
      DIMENSION BEGEND(2,2)
      EQUIVALENCE (BEGEND(1),BEG(1))
      COMMON /ACIDNT/ IDENT,ACWT
      REAL*8 IDENT
      COMMON /RYCTRL/NORAYS,STND,UL,UR,LL,LR,PRTRAY,TIMBEG,DELTIM,NTIMS,
     APHIBEG,DELPHI,NPHIS
      LOGICAL*1 NORAYS,STND,UL,UR,LL,LR,PRTRAY,LOGIC(2,2)
      EQUIVALENCE (LOGIC(1),UL)
      REAL PHIB(8),DPHI(8),SGN(2)/1.,-1./
      INTEGER MDEX(2,2)/1,2,2,1/
      COMMON /RAYNIT/ KGMH,NDCRVS,NUCRVS,IUPDWN,T0,PHI0,X0,Y0,Z0,
     AP10,P20,P30,OMEGA,DELTA0,P1F0,P2F0,P3F0,OMEGAF,XT0,YT0,ZT0,
     BP1T0,P2T0,P3T0,OMEGAT,XS0,YS0,ZS0,P3S0,RHO0,PCONST,NAGES,AGES(20)
      INTEGER*2 KGMH,NDCRVS,NUCRVS,IUPDWN
      LOGICAL BETWEN
      BETWEN(A,B,C)=AMOD(AMOD(B-A,360.)+360.,360.) .LE.
     A              AMOD(AMOD(C-A,360.)+360.,360.)
      CALL SETUP
      REWIND 11
      REWIND 9
      WRITE(9) IDENT
    5 CONTINUE
      CALL TIMPHI(&900)
      DO 75 NTIM=1,NTIMS
      T0=TIMBEG+(NTIM-1)*DELTIM
      CALL ACMOVE(T0)
      CALL FILIMS
      IF(NORAYS.OR.(NLIMS.EQ.0))GO TO 75
      IF(.NOT.STND) GO TO 10
      NSETS=1
      PHIB(1)=PHIBEG
      DPHI(1)=DELPHI
      GO TO 40
   10 NSETS=0
      DO 35 N=1,NLIMS
      DO 30 M=1,2
      LUPDWN=1
      IF(BETWEN(-90.,BEGEND(N,M),90.)) LUPDWN=2
      LEFR=MDEX(LUPDWN,M)
      IF(.NOT.LOGIC(LEFR,LUPDWN)) GO TO 30
      NSETS=NSETS+1
      PHIB(NSETS)=BEGEND(N,M)
      DPHI(NSETS)=SIGN(DELPHI,SGN(M))
   30 CONTINUE
      DO 32 LEFR=1,2
      ANG=SIGN(90.,SGN(LEFR))
      IF(.NOT.BETWEN(BEG(N),ANG,END(N))) GO TO 32
      DO 31 LUPDWN=1,2
      IF(.NOT.LOGIC(LEFR,LUPDWN)) GO TO 31
      NSETS=NSETS+1
      PHIB(NSETS)=ANG
      DPHI(NSETS)=SIGN(DELPHI,SGN(MDEX(LUPDWN,LEFR)))
   31 CONTINUE
   32 CONTINUE
   35 CONTINUE
      IF(NSETS.EQ.0) GO TO 75
   40 DO 70 NSET=1,NSETS
      DO 65 NPHI=1,NPHIS
      PHI0=PHIB(NSET)+(NPHI-1)*DPHI(NSET)
      DO 45 L=1,NLIMS
      IF(BETWEN(BEG(L),PHI0,END(L))) GO TO 60
   45 CONTINUE
      GO TO 65
   60 CALL RAYORG(&65)
      CALL RAYTRK
   65 CONTINUE
   70 CONTINUE
   75 CONTINUE
      GO TO 5
  900 CALL RDSPCL
      CALL SIGNUR
      STOP
      END
C./       ADD   NAME=BLKDATA
      BLOCK DATA
      COMMON /PUNITS/ PTABL,CPTABL,TTABL,CTTABL,HTABL,CHTABL,
     1 ATMPOT,ACPOT,STABL,CSTABL,TIMTAB,LTABL,CLTABL,FTABL,CFTABL
      REAL*8 PTABL(6),TTABL(4),HTABL(6),STABL(9),TIMTAB(2),LTABL(6)
      REAL   CPTABL(6),CTTABL(2,4),CHTABL(6),CSTABL(9),CLTABL(6)
      REAL*8 FTABL(5)
      REAL   CFTABL(5)
      LOGICAL*1 ATMPOT(6),ACPOT(6)
C  DEFAULT ON HEIGHT = GEOPOTENTIAL(ATMOSPHERE), GEOMETRIC(AIRCRAFT)
C  INTERNAL PROGRAM UNITS HEIGHT GEOPOTENTIAL METERS(ATMOSPHERE),
C  GEOMETRIC METERS ALL OTHER HEIGHTS AND LENGTHS, TEMPERATURE
C   DEGREES KELVIN, PRESSURE KILOPASCALS (KPA), SPEED METERS PER SECOND.
      DATA PTABL/'KPA','MB','NSM','PA','PSF','PSI'/
      DATA CPTABL/1.,0.1,1.E3,1.E3,4.78803E-2,6.89476/
      DATA TTABL/'C','F','K','R'/
      DATA CTTABL/ 1.,273.150,  1.80,459.670,  1.,0.,  1.80,0./
      DATA HTABL/'FT','GMFT','GMM','GPFT','GPM','METERS'/
      DATA CHTABL/.3048,.3048,1.,.3048,1.,1./
      DATA ATMPOT/.TRUE.,2*.FALSE.,3*.TRUE./
      DATA ACPOT/3*.FALSE.,2*.TRUE.,.FALSE./
      DATA STABL/'FPS','FTPS','KNOTS','KT','KPH','MPH','MPS','NMPH',
     A 'SMPH'/
      DATA CSTABL/2*.3048,2*.5144444,.2777778,.4470400,1.,.5144444,
     A .4470400/
      DATA TIMTAB/'HHMMSS','SSSSSSSS'/
      DATA LTABL/'FT','KM','METERS','MILES','NMI','SMI'/
      DATA CLTABL/.3048,1E3,1.,1609.344,1852.,1609.344/
      DATA FTABL/'GM','GRAMS','KG','LB','POUNDS'/
      DATA CFTABL/2*1E-3,1.,2*.45359237/
C   KNOTS AND NAUTICAL MILE CONVERSIONS BASED ON THE INTERNATIONAL
C   NAUTICAL MILE OF 1852. METERS EXACTLY (6076.1155FT), AS ADOPTED
C   BY THE U.S. IN 1954, RATHER THAN THE BRITISH ADMIRALTY NAUTICAL
C   MILE OF 6080 FT. (1853.184METERS) OR THE U.S. NAUTICAL MILE PRIOR
C   TO 1954 OF 6080.21 FT. (1853.250METERS)
C ***********************************************
      COMMON /ATMCON/ REARTH,G0,RSTAR,R0M0,R0G0M0
C  REARTH=RADIUS OF EARTH FOR CONVERSION GEOMETRIC TO GEOPOTENTIAL
C  METERS.  R0M0=RSTAR/M0 AND R0G0M0=RSTAR/(G0*M0)
C   WHERE RSTAR=UNIVERSAL GAS CONSTANT=8.31432E3 JOULES / (KMOL-DEGK),
C    G0=9.80665M/SEC**2, AND M0=MEAN MOLECULAR WEIGHT OF STANDARD DRY
C   AIR (28.9644 KG/KMOL)  (SEE U.S. STANDARD ATMOSPHERE 1976)
      DATA REARTH/6.35677E6/,R0G0M0/29.27127/,RSTAR/8.31432E3/
      DATA G0/9.80665/,R0M0/287.0531/
C ***********************************************
      COMMON /CLASES/CNAMES(30),NRCURV(2,2),TYPRAY(3,2,2),DIRECT,LOFT,
     A UP,DOWN
      REAL*8 CNAMES
      LOGICAL *1 TYPRAY,DIRECT,LOFT,UP,DOWN
C   RAY CLASSES. G=GROUND, M=MID HEIGHT (ABOUT 50KM) H=EXTREME
C   HEIGHT (100KM OR MORE). RAY CLASSES DEFINED IN THE ORDER IN WHICH
C   A RAY TOUCHES AND RETURNS FROM ANY OF THESE LAYERS. THUS, A GMG
C   RAY HAS REFLECTED FROM THE GROUND, RECURVED FROM THE MID LEVEL, AND
C   TOUCHED THE GROUND AGAIN. A MG (OR M) RAY ROSE DIRECTLY FROM
C   AIRCRAFT TO MID LAYER AND RECURVED TO TOUCH GROUND.
      DATA CNAMES/'ENDCLASS','FULL','G','GH','GHG','GHGH','GHGHG',
     A'GHGHGH','GHGHGHG','GM','GMG','GMGM','GMGMG','GMGMGM','GMGMGMG',
     B'H','HG','HGH','HGHG','HGHGH','HGHGHG','M','MG','MGM','MGMG',
     C'MGMGM','MGMGMG','NOPRINT','SHOCKS','SUMMARY'/
      END
C./       ADD   NAME=SETUP
C  *****************************************************************
C  *** INPUT ROUTINES - SETUP,ATMSIN,PTDHIN,RAOBWK,WINDIN,TXYZIN     ***
C  ***                  SPLINA,SPLINV (DATIM2,FFF2A)                ***
C  *****************************************************************
      SUBROUTINE SETUP
      COMMON /ACIDNT/ IDENT,ACWT
      REAL*8 IDENT
      COMMON /PRINTS/ TITLE(30),KTPSIG,CVRTIM,TIMLBL
      REAL*8 TIMLBL
      LOGICAL CVRTIM
      COMMON /GROUND/ ZGRND,CGRND,UGRND,VGRND,REFLFC
      COMMON /ATMSPH/ GAM,C,U,V,DCDZ,DUDZ,DVDZ,D2CDZ2,D2UDZ2,D2VDZ2,RHO
      REAL*8 GAM,C,U,V
      COMMON /PUNITS/ PTABL,CPTABL,TTABL,CTTABL,HTABL,CHTABL,
```

```
     1 ATMPOT,ACPOT,STABL,CSTABL,TIMTAB,LTABL,CLTABL,FTABL,CFTABL
      REAL*8 PTABL(6),TTABL(4),HTABL(6),STABL(9),TIMTAB(2),LTABL(6)
      REAL   CPTABL(6),CTTABL(2,4),CHTABL(6),CSTABL(9),CLTABL(6)
      REAL*8 FTABL(5)
      REAL   CFTABL(5)
      LOGICAL*1 ATMPOT(6),ACPOT(6)
C  DEFAULT ON HEIGHT = GEOPOTENTIAL(ATMOSPHERE), GEOMETRIC(AIRCRAFT)
C  INTERNAL PROGRAM UNITS HEIGHT GEOPOTENTIAL METERS(ATMOSPHERE),
C  GEOMETRIC METERS ALL OTHER HEIGHTS AND LENGTHS, TEMPERATURE
C   DEGREES KELVIN, PRESSURE KILOPASCALS (KPA), SPEED METERS PER SECOND.
      COMMON /ATMCON/ REARTH,G0,RSTAR,R0M0,R0G0M0
      COMMON /CLASES/CNAMES(30),NRCURV(2,2),TYPRAY(3,2,2),DIRECT,LOFT,
     A UP,DOWN
      REAL*8 CNAMES
      LOGICAL *1 TYPRAY,DIRECT,LOFT,UP,DOWN
C   RAY CLASSES. G=GROUND, M=MID HEIGHT (ABOUT 50KM) H=EXTREME
C   HEIGHT (100KM OR MORE). RAY CLASSES DEFINED IN THE ORDER IN WHICH
C   A RAY TOUCHES AND RETURNS FROM ANY OF THESE LAYERS. THUS, A GMG
C   RAY HAS REFLECTED FROM THE GROUND, RECURVED FROM THE MID LEVEL, AND
C   TOUCHED THE GROUND AGAIN. A MG (OR M) RAY ROSE DIRECTLY FROM
C   AIRCRAFT TO MID LAYER AND RECURVED TO TOUCH GROUND.
      INTEGER KTRNS(29)/4,0,24*-1,1,3,2/
      REAL*8 PRTYP(4)/'  NO','SUMMARY',' SHOCKS','  FULL'/
      REAL*8 KARD(9),BUF(9),TPUNIT(3)/'WEIGHT','HEIGHT','RAYCLASS'/
      REAL SBUF(18)
      EQUIVALENCE(BUF(1),SBUF(1))
C   READ 2 TITLE CARDS (1-72 ON FIRST CARD, 1-24 ON 2ND)
      READ(5,10) TITLE
   10 FORMAT(18A4)
      CALL DATIM2(SBUF)
      DO 7 K=1,6
      TITLE(K+24)=SBUF(K)
    7 CONTINUE
      WRITE(6,15) TITLE
   15 FORMAT('1',30A4)
C
C   READ AIRCRAFT I/D, AIRCRAFT WEIGHT & WEIGHT UNITS.
C   1-8 ACID  9-16 AC WEIGHT 17-24 WEIGHT UNITS
      READ(5,20) KARD
   20 FORMAT(9A8)
      CALL LJUST(8,3,KARD,BUF)
      CALL UNITIS(BUF(3),FTABL,5,IMUNIT,TPUNIT(1),4)
      IDENT=BUF(1)
      CALL FFA2N(KARD,9,8,1,ACWT,0.,KERR)
      ACWT=ACWT*CFTABL(IMUNIT)
      BUF(9)=ACWT/CFTABL(IMUNIT)
      WRITE(6,35) IDENT,BUF(9),FTABL(IMUNIT)
   35 FORMAT(' AIRCRAFT TYPE= ',A8,', WEIGHT=',F8.0,1X,A8)
C
C READ GROUND HEIGHT AND UNITS. READ REFLECTION FACTOR.
C   1-8 HEIGHT  9-16 HEIGHT UNITS  17-24 REFLECTION FACTOR.
      READ(5,20) KARD
      CALL LJUST(8,2,KARD,BUF)
      CALL UNITIS(BUF(2),HTABL,6,IGUNIT,TPUNIT(2),6)
      CALL FFA2N(BUF,1,8,1,HEIGHT,0.,KERR)
      CALL FFA2N(BUF,17,8,1,REFLFC,1.,KERR)
      HEIGHT=HEIGHT/CHTABL(IGUNIT)
      IF(ACPOT(IGUNIT)) HEIGHT=HEIGHT/(1.-HEIGHT/REARTH)
      ZGRND=HEIGHT
      IF(ACPOT(IGUNIT)) HEIGHT=HEIGHT/(1.+HEIGHT/REARTH)
      HEIGHT=HEIGHT*CHTABL(IGUNIT)
      WRITE(6,40) HEIGHT,HTABL(IGUNIT),REFLFC
   40 FORMAT(' HEIGHT OF GROUND=',F5.0,1X,A8,'   REFLECTION FACTOR=',
     AF5.3)
C
C   READ RAY TYPES TO BE RECORDED.
      DO 42 K=1,2
      DO 42 L=1,2
      DO 41 M=1,3
      TYPRAY(M,K,L)=.FALSE.
   41 CONTINUE
      NRCURV(K,L)=-1
   42 CONTINUE
      KTPSIG=-1
      UP=.FALSE.
      DOWN=.FALSE.
      DIRECT=.FALSE.
      LOFT=.FALSE.
   43 READ(5,20) KARD
      CALL LJUST(8,9,KARD,BUF)
      DO 55 K=1,9
      CALL UNITIS(BUF(K),CNAMES,30,LCUNIT,TPUNIT(3),1)
      IF(LCUNIT.LE.1) GO TO 60
      IF(KTRNS(LCUNIT-1)) 47,45,50
   45 DIRECT=.TRUE.
      DOWN=.TRUE.
      GO TO 55
   47 IF(LCUNIT.GE.28) GO TO 50
      LOFT=LOFT.OR.(LCUNIT.LE.14)
      LCUNIT=(LCUNIT-4)/2
      KDWNUP=LCUNIT/6
      KHM=LCUNIT/3-KDWNUP*2
      KMH=2-KHM
      KRCRV=LCUNIT-3*KHM-6*KDWNUP
      TYPRAY(KRCRV+1,KDWNUP+1,KMH)=.TRUE.
      NRCURV(KDWNUP+1,KMH)=MAX0(NRCURV(KDWNUP+1,KMH),KRCRV+1)
      UP=UP.OR.(KDWNUP.NE.0)
      DOWN=DOWN.OR.(KDWNUP.EQ.0)
      GO TO 55
   50 KTPSIG=MAX0(KTPSIG,KTRNS(LCUNIT-1)-1)
   55 CONTINUE
      GO TO 43
   60 WRITE(6,65)
   65 FORMAT('0',T10,'PROCESS FOLLOWING CLASSES OF RAYS:')
      IF(DIRECT) WRITE(6,70) CNAMES(3)
   70 FORMAT(T20,A8)
      DO 75 K=1,2
      DO 75 L=1,2
      DO 75 M=1,3
      IF(.NOT.TYPRAY(M,K,L)) GO TO 75
      KTABL=12*(K-1)+6*(2-L)+2*M+3
      WRITE(6,70)CNAMES(KTABL)
   75 CONTINUE
      IF(KTPSIG.EQ.-1) KTPSIG=3
      WRITE(6,78) PRTYP(KTPSIG+1)
   78 FORMAT('0',T10,'PERFORM ',A7,' LISTING FROM SIGNATURES.')
      CALL ATMSIN
      CALL TXYZIN
      CALL FNDLYR(ZGRND,&80)
   80 CALL AIR(DBLE(ZGRND))
      CGRND=C
      UGRND=U
      VGRND=V
      RETURN
      END
C./       ADD   NAME=ATMSIN
      SUBROUTINE ATMSIN
      COMMON /PTH/ NPTH,PRESS(97),TMPMOL(97),GPHC(97),GAMMA(97)
      COMMON /WINDS/ NWINDS,GPHW(80),DIR(80),TURN(79),SPEED(80)
      COMMON /ATMCON/ REARTH,G0,RSTAR,R0M0,R0G0M0
      COMMON /GROUND/ ZGRND,CGRND,UGRND,VGRND,REFLFC
      COMMON /LYRDEF/NLAYER,GMZA(200),INDPTH(200),INDWND(200),
     ALYRPRT(200),KLAYER,ZTOP,ZBOT
      INTEGER*2 INDPTH,INDWND
      LOGICAL*1 LYRPRT
      REAL PRINTL(24)/-5.E3,0.,2.E3,4.E3,6.E3,8.E3,10.E3,15.E3,20.E3,
     A25.E3,30.E3,35.E3,40.E3,45.E3,5.E4,6.E4,7.E4,8.E4,9.E4,10.E4,11.E4
     B,12.E4,13.E4,1.E7/
      ZFROMH(H)=H/(1.-H/REARTH)
      CALL PTDHIN
      CALL WINDIN
      ZPTH=ZFROMH(GPHC(1))
      ZWIND=ZFROMH(GPHW(1))
      NLAYER=1
      GMZA(NLAYER)=AMAX1(ZGRND,ZPTH,ZWIND)
      DO 2 KPRT=1,24
      IF(PRINTL(KPRT).GT.GMZA(1)) GO TO 3
      LPRT=MIN0(KPRT,23)
    2 CONTINUE
    3 DO 4 KPTH=1,NPTH
      ZPTH=ZFROMH(GPHC(KPTH))
      IF(ZPTH.GT.GMZA(1)) GO TO 5
      LPTH=MIN0(KPTH,NPTH-1)
    4 CONTINUE
    5 INDPTH(NLAYER)=LPTH
      DO 6 KWIND=1,NWINDS
      ZWIND=ZFROMH(GPHW(KWIND))
      IF(ZWIND.GT.GMZA(1)) GO TO 7
      LWIND=MIN0(KWIND,NWINDS-1)
    6 CONTINUE
    7 INDWND(NLAYER)=LWIND
   10 KPRT=MIN0(LPRT+1,24)
      KPTH=MIN0(LPTH+1,NPTH)
      KWIND=MIN0(LWIND+1,NWINDS)
      ZPRT=PRINTL(KPRT)
      ZPTH=ZFROMH(GPHC(KPTH))
      ZWIND=ZFROMH(GPHW(KWIND))
      ZLEVEL=AMIN1(ZPRT,ZWIND,ZPTH)
      IF(ZLEVEL.LE.GMZA(NLAYER).OR.(NLAYER.GE.200)) GO TO 200
      NLAYER=NLAYER+1
      GMZA(NLAYER)=ZLEVEL
      LYRPRT(NLAYER)=.FALSE.
      IF(GMZA(NLAYER).LT.ZPRT) GO TO 30
      LYRPRT(NLAYER)=.TRUE.
      LPRT=MIN0(KPRT,23)
   30 IF(GMZA(NLAYER).EQ.ZPTH) LPTH=MIN0(KPTH,NPTH-1)
      INDPTH(NLAYER)=LPTH
      IF(GMZA(NLAYER).EQ.ZWIND) LWIND=MIN0(KWIND,NWINDS-1)
      INDWND(NLAYER)=LWIND
      GO TO 10
  200 LYRPRT(1)=.TRUE.
      LYRPRT(NLAYER)=.TRUE.
      RETURN
      END
C./       ADD   NAME=PTDHIN
      SUBROUTINE PTDHIN
      COMMON /PRINTS/ TITLE(30),KTPSIG,CVRTIM,TIMLBL
      REAL*8 TIMLBL
      LOGICAL CVRTIM
      COMMON /PTH/ NPTH,PRESS(97),TMPMOL(97),GPHC(97),GAMMA(97)
C  --  TMPMOL= 'MOLECULAR SCALE TEMPERATURE' = VIRTUAL TEMPERATURE
C  --  GPH = GEOPOTENTIAL HEIGHT
      REAL GPH(80),TEMPK(80),DEWPNT(80)
      REAL STANHT(21)/-5E3,11E3,20E3,32E3,47E3,51E3,71E3,84852.,89716.,
     A 94572.,97482.,99420.,102326.,104261.,106196.,107162.,108129.,
     B 117777.,121627.,125473.,130274./
      REAL STANTP(21)/320.65,216.65,216.65,228.65,270.65,270.65,214.65,
     A 186.95,187.16,189.35,194.28,204.63,213.22,221.65,234.19,242.86,
     B 254.27,397.09,453.89,508.05,571.42/
      REAL STANGM(21)/7*1.401,2*1.402,1.404,1.406,1.408,1.411,1.413,
     A 1.416,1.417,1.419,1.432,1.436,1.441,1.446/
      COMMON /PUNITS/ PTABL,CPTABL,TTABL,CTTABL,HTABL,CHTABL,
     1 ATMPOT,ACPOT,STABL,CSTABL,TIMTAB,LTABL,CLTABL,FTABL,CFTABL
      REAL*8 PTABL(6),TTABL(4),HTABL(6),STABL(9),TIMTAB(2),LTABL(6)
      REAL   CPTABL(6),CTTABL(2,4),CHTABL(6),CSTABL(9),CLTABL(6)
      REAL*8 FTABL(5)
      REAL   CFTABL(5)
      LOGICAL*1 ATMPOT(6),ACPOT(6)
C  DEFAULT ON HEIGHT = GEOPOTENTIAL(ATMOSPHERE), GEOMETRIC(AIRCRAFT)
C  INTERNAL PROGRAM UNITS HEIGHT GEOPOTENTIAL METERS(ATMOSPHERE),
```

```
C  GEOMETRIC METERS ALL OTHER HEIGHTS AND LENGTHS, TEMPERATURE
C   DEGREES KELVIN, PRESSURE KILOPASCALS (KPA), SPEED METERS PER SECOND.
      REAL*8 STANRD/'STANDARD'/,FINISH/'END'/,BLANK/' '/
      REAL*8 TPUNIT(4)/'PRESSURE','  TEMP',' DEW PT.',' HEIGHT'/
      REAL*8 KARD(9),BUF(4),KARD7(7)
      REAL DUMMY(7),DEFARY(4)/4*-1.E6/
      COMMON /WKRAOB/ HMISS(97),PMISS(97)
      LOGICAL*1 HMISS,PMISS
      LOGICAL*1 DMISS(80),GEOMET,GEOPOT,SOMEHT,TRUE/.TRUE./
      LOGICAL*1 FALSE/.FALSE./
      COMMON /ATMCON/ REARTH,G0,RSTAR,R0M0,R0G0M0
C  REARTH=RADIUS OF EARTH FOR CONVERSION GEOMETRIC TO GEOPOTENTIAL
C  METERS.
      VAPRS(DWPT)=.6105*EXP(25.22*(1.-273./DWPT)-5.31*ALOG(DWPT/273.))
      RATMIX(PRS,DWPT)=0.622/((PRS/VAPRS(DWPT))-1.)
      VIRTMP(TMP,RTMIX)=TMP*(1.+0.61653*RTMIX)
      GMW(RTMIX)=1.401*(1.+1.899*RTMIX)/(1.+2.016*RTMIX)
      PRESS(1)=177.68
      PMISS(1)=FALSE
      TMPMOL(1)=STANTP(1)
C
C   READ TITLE/STANDARD CARD AND INTERPRET
C
      READ(10,5,END=200) KARD
    5 FORMAT(9A8)
      CALL LJUST(8,4,KARD,BUF)
      CALL LOOKUP(8,1,STANRD,BUF(1),ISTND,&6,&7)
    6 IF(BUF(1).EQ.BLANK) GO TO 7
      GO TO 200
    7 WRITE(6,8) TITLE
      WRITE(6,9) KARD
    8 FORMAT('1',30A4)
    9 FORMAT('0',9A8)
C
C   READ UNITS CARD AND INTERPRET
C
      READ(10,5,END=200) KARD
      CALL LJUST(8,4,KARD,BUF)
      CALL UNITIS(BUF(1),PTABL,6,IPUNIT,TPUNIT(1),2)
      CALL UNITIS(BUF(2),TTABL,4,ITUNIT,TPUNIT(2),0)
      CALL UNITIS(BUF(3),TTABL,4,IDUNIT,TPUNIT(3),ITUNIT)
      IF(ITUNIT.EQ.0) ITUNIT=IDUNIT
      IF(ITUNIT.NE.0) GO TO 30
      ITUNIT=1
      IDUNIT=1
   30 CALL UNITIS(BUF(4),HTABL,6,IHUNIT,TPUNIT(4),5)
      GEOMET=.NOT.ATMPOT(IHUNIT)
C
C  READ IN DATA VALUES P-T-D-H. CONVERT TO INTERNAL UNITS.
C  CHECK FOR MISSING VALUES OF DEWPOINT AND HEIGHT.
C
      TEMPK(1)=STANTP(1)
      HMISS(1)=TRUE
      DMISS(1)=TRUE
      SOMEHT=FALSE
      DO 50 N=2,80
      READ(10,5,END=55) KARD
      CALL LJUST(8,4,KARD,BUF)
      IF(BUF(1).EQ.FINISH) GO TO 55
      CALL FFA2N(KARD,1,8,4,DUMMY,DEFARY,KERR)
      PRESS(N)=DUMMY(1)*CPTABL(IPUNIT)
      TEMPK(N)=(DUMMY(2)+CTTABL(2,ITUNIT))/CTTABL(1,ITUNIT)
      DEWPNT(N)=(DUMMY(3)+CTTABL(2,IDUNIT))/CTTABL(1,IDUNIT)
      GPH(N)=DUMMY(4)*CHTABL(IHUNIT)
      IF(GEOMET) GPH(N)=GPH(N)/(1.+GPH(N)/REARTH)
      TMPMOL(N)=TEMPK(N)
      GPHC(N)=GPH(N)
      DMISS(N)=DEWPNT(N).LT.0.
      PMISS(N)=PRESS(N).LE.0.
      HMISS(N)=GPH(N).LT.-1.E4
      IF(TEMPK(N).LT.0..OR.(PMISS(N).AND.HMISS(N))) GO TO 65
      IF(HMISS(N).OR.PMISS(N)) GO TO 50
      IF(SOMEHT) GO TO 50
      SOMEHT=TRUE
      IPTHT=N
   50 CONTINUE
      N=81
      WRITE(6,51)
   51 FORMAT(' P-T-D-H READING TERMINATED AFTER 79 ITEMS.')
   55 NPTH=N-1
      IF(SOMEHT) GO TO 70
      WRITE(6,60)
   60 FORMAT(' AT NO LEVEL IS BOTH HEIGHT AND PRESSURE GIVEN. CANNOT EVA
     ALUATE ATMOSPHERIC PROFILE. RUN ABORTED.')
      STOP 650
   65 WRITE(6,67) KARD
   67 FORMAT(' INSUFFICIENT DATA ON CARD:''',9A8,''''/' RUN ABORTED.')
      STOP 650
   70 CALL RAOBWK(1,IPTHT,-1)
      CALL RAOBWK(IPTHT,NPTH,1)
C
C   WORK DOWN TO OBTAIN VIRTUAL TEMPERATURES.  BEFORE TOPMOST DEW POINT,
C   MIXING RATIO IS ZERO.  DEW POINT INTERPOLATED LINEARLY ACROSS GAPS
C   W.R.T. DRY GPH, CONSTANT BELOW LOWEST INPUT DEW POINT.
C
      DO 71 NN=1,NPTH
      N=NPTH-NN+1
      IF (.NOT.DMISS(N)) GO TO 72
      GAMMA(N)=1.401
   71 CONTINUE
      GO TO 80
   72 N2=N
      DOLD=DEWPNT(N2)
      HOLD=GPHC(N2)
      DO 77 NN=2,N2
      N=N2-NN+2
      DO 73 N3=2,N
      N4=N-N3+1
      IF (.NOT.DMISS(N4)) GO TO 74
   73 CONTINUE
      DNEW=DOLD
      GO TO 75
   74 DNEW=DEWPNT(N4)
   75 HNEW=GPHC(N4)
      N4=N4+1
      DO 76 N5=N4,N
      D=((GPHC(N5)-HOLD)*DNEW+(HNEW-GPHC(N5))*DOLD)/(HNEW-HOLD)
      RTMIX=RATMIX(PRESS(N5),D)
      GAMMA(N5)=GMW(RTMIX)
      TMPMOL(N5)=VIRTMP(TEMPK(N5),RTMIX)
   76 CONTINUE
      DOLD=DNEW
      HOLD=HNEW
   77 CONTINUE
      RTMIX=RATMIX(PRESS(1),DOLD)
      GAMMA(1)=GMW(RTMIX)
      TMPMOL(1)=VIRTMP(TEMPK(1),RTMIX)
   80 CALL RAOBWK(1,IPTHT,-1)
      CALL RAOBWK(IPTHT,NPTH,1)
C
C PRINT OUT WORKED UP VALUES IN ORIGINAL UNITS
C
      WRITE(6,100) PTABL(IPUNIT),TTABL(ITUNIT),TTABL(IDUNIT),
     AHTABL(IHUNIT),HTABL(IHUNIT),TTABL(ITUNIT)
  100 FORMAT('0',T17,'TEMPERATURE',T35,'HEIGHT',T49,'VIRTUAL',T60,
     A 'SOUND'/2X,'PRESSURE',T13,'KINETIC',T21,'DEW POINT',T32,'INPUT',
     B T39,'COMPUTED',T50,'TEMP.',T60,'SPEED'/5X,6A9,T62,'MPS')
      GEOPOT=.NOT.GEOMET
      DO 110 N=2,NPTH
      DUMMY(1)=PRESS(N)/CPTABL(IPUNIT)
      DUMMY(2)=TEMPK(N)*CTTABL(1,ITUNIT)-CTTABL(2,ITUNIT)
      DUMMY(3)=DEWPNT(N)*CTTABL(1,IDUNIT)-CTTABL(2,IDUNIT)
      H1PRNT=GPH(N)
      H2PRNT=GPHC(N)
      IF(GEOPOT) GO TO 105
      H1PRNT=H1PRNT/(1.-H1PRNT/REARTH)
      H2PRNT=H2PRNT/(1.-H2PRNT/REARTH)
  105 DUMMY(4)=H1PRNT/CHTABL(IHUNIT)
      DUMMY(5)=H2PRNT/CHTABL(IHUNIT)
      DUMMY(6)=TMPMOL(N)*CTTABL(1,ITUNIT)-CTTABL(2,ITUNIT)
      DUMMY(7)=SQRT(R0M0*GAMMA(N)*TMPMOL(N))
      CALL FFN2A(KARD7,1,-8,4,7,DUMMY)
      IF(HMISS(N)) KARD7(4)=BLANK
      IF (DMISS(N)) KARD7(3)=BLANK
      WRITE(6,107) KARD7
  107 FORMAT(1X,7A9)
  110 CONTINUE
      GO TO 300
C
C   STANDARD ATMOSPHERE BASIS PREPARATION
C
  200 GPHC(1)=STANHT(1)
      HMISS(1)=FALSE
      GAMMA(1)=STANGM(1)
      NPTH=1
      WRITE(6,210)
  210 FORMAT('0STANDARD ATMOSPHERE TABLE SELECTED.')
C
C   MERGE IN STANDARD ATMOSPHERE (1976)
C
  300 STANLO=GPHC(NPTH)+1000.
      IF(GPHC(NPTH).GT.STANHT(21)) RETURN
      DO 310 K=2,21
      IF(STANLO.LE.STANHT(K)) GO TO 320
  310 CONTINUE
      K=21
  320 L2=MIN0(22-K,97-NPTH)
      IF(L2.LT.1) RETURN
      DO 350 L=1,L2
      NL=NPTH+L
      LK=L+K-1
      GPHC(NL)=STANHT(LK)
      TMPMOL(NL)=STANTP(LK)
      GAMMA(NL)=STANGM(LK)
      PMISS(NL)=TRUE
      HMISS(NL)=FALSE
  350 CONTINUE
      CALL RAOBWK(NPTH,NPTH+L2,1)
      NPTH=NPTH+L2
      RETURN
      END
C./       ADD   NAME=RAOBWK
      SUBROUTINE RAOBWK(ILOW,IHIGH,IDIR)
      COMMON /PTH/ NPTH,PRESS(97),TMPMOL(97),GPHC(97),GAMMA(97)
C  --  TMPMOL= 'MOLECULAR SCALE TEMPERATURE' = VIRTUAL TEMPERATURE
C  --  GPH = GEOPOTENTIAL HEIGHT
      COMMON /WKRAOB/ HMISS(97),PMISS(97)
      LOGICAL*1 HMISS,PMISS
      COMMON /ATMCON/ REARTH,G0,RSTAR,R0M0,R0G0M0
C   R0M0=RSTAR/M0 AND R0G0M0=RSTAR/(G0*M0)
C   WHERE RSTAR=UNIVERSAL GAS CONSTANT=8.31432E3 JOULES / (KMOL-DEGK),
C    G0=9.80665M/SEC**2, AND M0=MEAN MOLECULAR WEIGHT OF STANDARD DRY
C   AIR (28.9644 KG/KMOL)  (SEE U.S. STANDARD ATMOSPHERE 1976)
      F1S(TAU)=(((TAU/5.+1.)*TAU/4.+1.)*TAU/3.+1.)*TAU/2.+1.
      F1A(TAU)=(EXP(TAU)-1.)/TAU
      IF(ILOW.GE.IHIGH) RETURN
      IDIR=ISIGN(1,IDIR)
      IF (IDIR.EQ.0) RETURN
      KOFSET=0
      IF(IDIR.LT.0) KOFSET=ILOW+IHIGH
```

```
      ISTOP=IHIGH-1
      DO 9 NN=ILOW,ISTOP
      N=KOFSET+ISIGN(NN,IDIR)
      TAU=ALOG(TMPMOL(N)/TMPMOL(N+IDIR))
      IF(ABS(TAU).GT..1) GO TO 2
      FACTOR=TMPMOL(N)*F1S(TAU)
      GO TO 3
    2 FACTOR=TMPMOL(N)*F1A(TAU)
    3 IF(PMISS(N+IDIR)) GO TO 5
      THICK=ALOG(PRESS(N)/PRESS(N+IDIR))*FACTOR*R0G0M0
      GPHC(N+IDIR)=GPHC(N)+THICK
      GO TO 9
    5 PRESS(N+IDIR)=PRESS(N)*EXP((GPHC(N)-GPHC(N+IDIR))/(FACTOR*R0G0M0))
    9 CONTINUE
      RETURN
      END
C./       ADD   NAME=WINDIN
      SUBROUTINE WINDIN
      COMMON /PRINTS/ TITLE(30),KTPSIG,CVRTIM,TIMLBL
      REAL*8 TIMLBL
      LOGICAL CVRTIM
      COMMON /WINDS/ NWINDS,GPHW(80),DIR(80),TURN(79),SPEED(80)
      COMMON /PUNITS/ PTABL,CPTABL,TTABL,CTTABL,HTABL,CHTABL,
     1 ATMPOT,ACPOT,STABL,CSTABL,TIMTAB,LTABL,CLTABL,FTABL,CFTABL
      REAL*8 PTABL(6),TTABL(4),HTABL(6),STABL(9),TIMTAB(2),LTABL(6)
      REAL   CPTABL(6),CTTABL(2,4),CHTABL(6),CSTABL(9),CLTABL(6)
      REAL*8 FTABL(5)
      REAL   CFTABL(5)
      LOGICAL*1 ATMPOT(6),ACPOT(6)
C  DEFAULT ON HEIGHT = GEOPOTENTIAL(ATMOSPHERE), GEOMETRIC(AIRCRAFT)
C  INTERNAL PROGRAM UNITS HEIGHT GEOPOTENTIAL METERS(ATMOSPHERE),
C  GEOMETRIC METERS ALL OTHER HEIGHTS AND LENGTHS, TEMPERATURE
C   DEGREES KELVIN, PRESSURE KILOPASCALS (KPA), SPEED METERS PER SECOND.
      COMMON /ATMCON/ REARTH,G0,RSTAR,R0M0,R0G0M0
C  REARTH=RADIUS OF EARTH FOR CONVERSION GEOMETRIC TO GEOPOTENTIAL
C  METERS.  R0M0=RSTAR/M0 AND R0G0M0=RSTAR/(G0*M0)
C   WHERE RSTAR=UNIVERSAL GAS CONSTANT=8.31432E3 JOULES / (KMOL-DEGK),
C    G0=9.80665M/SEC**2, AND M0=MEAN MOLECULAR WEIGHT OF STANDARD DRY
C   AIR (28.9644 KG/KMOL)  (SEE U.S. STANDARD ATMOSPHERE 1976)
      REAL*8 NOWIND/'NOWINDS'/,FINISH/'END'/
      REAL*8 TPUNIT(3)/'HEIGHT','DIRECT','SPEED'/
      REAL*8 KARD(9),BUF(4),BLANK/' '/
      REAL DUMMY(3),DEFARY(3)/3*0./
      LOGICAL*1 GEOMET,NOCAP,TRUE/.TRUE./,FALSE/.FALSE./
      GPHW(1)=-5E3
      DIR(1)=0.
      SPEED(1)=0.
C
C   READ TITLE/NOWINDS CARD AND INTERPRET
C
      READ(15,5,END=200) KARD
    5 FORMAT(9A8)
      CALL LJUST(8,3,KARD,BUF)
      CALL LOOKUP(8,1,NOWIND,BUF(1),ISTND,&6,&7)
    6 IF(BUF(1).EQ.BLANK) GO TO 7
      GO TO 200
    7 WRITE(6,8) TITLE
      WRITE(6,9) KARD
    8 FORMAT('1',30A4)
    9 FORMAT('0',5X,9A8)
C
C   READ UNITS CARD AND INTERPRET
C
      READ(15,5,END=200) KARD
      CALL LJUST(8,3,KARD,BUF)
      CALL UNITIS(BUF(1),HTABL,6,IHUNIT,TPUNIT(1),5)
      CALL UNITIS(BUF(3),STABL,9,ISUNIT,TPUNIT(3),3)
      GEOMET=.NOT.ATMPOT(IHUNIT)
C
C  READ IN DATA VALUES H-DIR-SPD. CONVERT TO INTERNAL UNITS.
C COMPUTE TURN (RATE OF DIRECTION CHANGE PER METER)
C
      OLDTRN=0.
      NOCAP=FALSE
      DO 40 N=2,80
      READ(15,5,END=45) KARD
      CALL LJUST(8,3,KARD,BUF)
      IF(BUF(1).EQ.FINISH) GO TO 45
      CALL FFA2N(KARD,1,8,3,DUMMY,DEFARY,KERR)
      GPHW(N)=DUMMY(1)*CHTABL(IHUNIT)
      DIR(N)=DUMMY(2)
      SPEED(N)=DUMMY(3)*CSTABL(ISUNIT)
      IF(GEOMET) GPHW(N)=GPHW(N)/(1.+GPHW(N)/REARTH)
      IF(SPEED(N).EQ.0.) GO TO 35
      OLDTRN=(AMOD(AMOD(DIR(N)-DIR(N-1),360.)+540.,360.)-180.)/
     1 (GPHW(N)-GPHW(N-1))
      TURN(N-1)=OLDTRN
      GO TO 40
   35 TURN(N-1)=OLDTRN
      OLDTRN=0.
   40 CONTINUE
      N=81
      NOCAP=TRUE
      WRITE(6,41)
   41 FORMAT(' H-DIR-SPD READING TERMINATED AFTER 79 ITEMS.')
   45 NWINDS=N-1
      IF(NWINDS.EQ.80) GO TO 50
      NWINDS=NWINDS+1
      GPHW(NWINDS)=130274.
      SPEED(NWINDS)=0.
      DIR(NWINDS)=0.
      TURN(NWINDS-1)=OLDTRN
C
C   WORK DOWN TURN AND DIR FOR THE CASE SPEED=0.
C
   50 DO 60 NN=2,NWINDS
      N=NWINDS-NN+2
      IF(SPEED(N).EQ.0.) GO TO 60
      IF(SPEED(N-1).EQ.0.) GO TO 55
      OLDTRN=TURN(N-1)
      GO TO 60
   55 TURN(N-1)=OLDTRN
      DIR(N-1)=DIR(N)-OLDTRN*(GPHW(N)-GPHW(N-1))
      OLDTRN=0.
   60 CONTINUE
C
C PRINT OUT IN ORIGINAL UNITS
C
      WRITE(6,90) HTABL(IHUNIT),STABL(ISUNIT)
   90 FORMAT('0',T4,'HEIGHT',T16,'DIR',T23,'SPEED'/5X,A9,T16,'DEG',T25,
     A A9)
      NPRNT=NWINDS-1
      IF(NOCAP) NPRNT=NPRNT+1
      DO 100 N=2,NPRNT
      HPRNT=GPHW(N)
      IF(GEOMET) HPRNT=HPRNT/(1.-HPRNT/REARTH)
      HPRNT=HPRNT/CHTABL(IHUNIT)
      SPRNT=SPEED(N)/CSTABL(ISUNIT)
      WRITE (6,95) HPRNT,DIR(N),SPRNT
   95 FORMAT(1X,3F9.0)
  100 CONTINUE
      RETURN
C
C   NOWINDS SELECTED
C
  200 NWINDS=2
      GPHW(2)=130274.
      SPEED(2)=0.
      DIR(2)=0.
      TURN(1)=0.
      WRITE (6,210)
  210 FORMAT('0NOWINDS SELECTED.')
      RETURN
      END
C./       ADD   NAME=TXYZIN
      SUBROUTINE TXYZIN
      COMMON /PRINTS/ TITLE(30),KTPSIG,CVRTIM,TIMLBL
      REAL*8 TIMLBL,TIMCVR
      LOGICAL CVRTIM
      COMMON /FLIGHT/NFIXES,TIMEAC(100),XAC(100),YAC(100),ZAC(100),
     A XPPAC(100),YPPAC(100),ZPPAC(100)
      DIMENSION TMARK(100),X0AC(100),Y0AC(100),Z0AC(100),DUMMY(10)
      REAL DEFARY(4)/4*0./
      COMMON /PUNITS/ PTABL,CPTABL,TTABL,CTTABL,HTABL,CHTABL,
     1 ATMPOT,ACPOT,STABL,CSTABL,TIMTAB,LTABL,CLTABL
      REAL*8 PTABL(6),TTABL(4),HTABL(6),STABL(9),TIMTAB(2),LTABL(6)
      REAL   CPTABL(6),CTTABL(2,4),CHTABL(6),CSTABL(9),CLTABL(6)
      LOGICAL*1 ATMPOT(6),ACPOT(6)
C  DEFAULT ON HEIGHT = GEOPOTENTIAL(ATMOSPHERE), GEOMETRIC(AIRCRAFT)
C  INTERNAL PROGRAM UNITS HEIGHT GEOPOTENTIAL METERS(ATMOSPHERE),
C  GEOMETRIC METERS ALL OTHER HEIGHTS AND LENGTHS, TEMPERATURE
C   DEGREES KELVIN, PRESSURE KILOPASCALS (KPA), SPEED METERS PER SECOND.
      COMMON /ATMCON/ REARTH,G0,RSTAR,R0M0,R0G0M0
C  REARTH=RADIUS OF EARTH FOR CONVERSION GEOMETRIC TO GEOPOTENTIAL
C  METERS.  R0M0=RSTAR/M0 AND R0G0M0=RSTAR/(G0*M0)
C   WHERE RSTAR=UNIVERSAL GAS CONSTANT=8.31432E3 JOULES / (KMOL-DEGK),
C    G0=9.80665M/SEC**2, AND M0=MEAN MOLECULAR WEIGHT OF STANDARD DRY
C   AIR (28.9644 KG/KMOL)  (SEE U.S. STANDARD ATMOSPHERE 1976)
      REAL*8 KARD(9),BUF(4),FINISH/'END'/
      REAL*8 TPUNIT(4)/'TIME','XPOINT','YPOINT','HEIGHT'/
      LOGICAL*1 GEOPOT,TRUE/.TRUE./,FALSE/.FALSE./
      DIMENSION R0AC(100,3),RPPAC(100,3),RAC(100,3)
      DIMENSION RPPO(3),RPPN(3)
      EQUIVALENCE (R0AC(1,1),X0AC(1)),(R0AC(1,2),Y0AC(1)),
     A (R0AC(1,3),Z0AC(1)),(RAC(1,1),XAC(1)),(RPPAC(1,1),XPPAC(1))
C
C   READ SUBTITLE CARD
C
      READ(20,5,END=200) KARD
      WRITE (6,3) TITLE
    3 FORMAT('1',30A4)
      WRITE(6,4) KARD
    4 FORMAT('0',5X,9A8)
C
C   READ UNITS CARD AND INTERPRET
C
      READ(20,5,END=200) KARD
    5 FORMAT(9A8)
      CALL LJUST(8,4,KARD,BUF)
      CALL UNITIS(BUF(1),TIMTAB,2,ITIMZ,TPUNIT(1),1)
      TIMLBL=TIMTAB(ITIMZ)
      CVRTIM=ITIMZ.EQ.1
      CALL UNITIS(BUF(2),LTABL,6,IXUNIT,TPUNIT(2),0)
      CALL UNITIS(BUF(3),LTABL,6,IYUNIT,TPUNIT(3),IXUNIT)
      IF(IXUNIT.EQ.0) IXUNIT=IYUNIT
      IF(IXUNIT.NE.0) GO TO 30
      IXUNIT=5
      IYUNIT=5
   30 CALL UNITIS(BUF(4),HTABL,6,IZUNIT,TPUNIT(4),2)
   40 GEOPOT=ACPOT(IZUNIT)
C
C   READ IN DATA VALUES T-X-Y-Z. CONVERT TO INTERNAL UNITS.
C
      DO 50 N=1,100
      READ(20,5,END=55) KARD
      CALL LJUST(8,4,KARD,BUF)
      IF(BUF(1).EQ.FINISH) GO TO 55
      CALL FFA2N(KARD,1,8,4,DUMMY,DEFARY,KERR)
      TIMEAC(N)=TIMCVR(DBLE(DUMMY(1)),1)
      X0AC(N)=DUMMY(2)
```

```
      Y0AC(N)=DUMMY(3)
      Z0AC(N)=DUMMY(4)
      IF(N.GT.1) TIMEAC(N)=AMOD(AMOD(TIMEAC(N)-TIMEAC(N-1),86400.)+
     A 86400.,86400.)+TIMEAC(N-1)
      X0AC(N)=X0AC(N)*CLTABL(IXUNIT)
      Y0AC(N)=Y0AC(N)*CLTABL(IYUNIT)
      Z0AC(N)=Z0AC(N)*CHTABL(IZUNIT)
      IF(GEOPOT) Z0AC(N)=Z0AC(N)/(1.-Z0AC(N)/REARTH)
   50 CONTINUE
      N=101
      WRITE (6,51)
   51 FORMAT(' T-X-Y-Z READING TERMINATED AFTER 100 ITEMS.')
   55 NFIXES=N-1
      IF(NFIXES.GT.1) GO TO 57
      WRITE (6,56) NFIXES
   56 FORMAT(1X,I2,' FIXES OF AIRCRAFT POSITION SUPPLIED. MUST HAVE AT L
     AEAST TWO TO RUN.  RUN ABORTED.')
      STOP 650
C
C   COMPUTE SECOND DERIVATIVES FOR SPLINE FIT
C
   57 CALL SPLINA(TIMEAC,R0AC,RPPAC,100,3,NFIXES,TMARK)
C
C   SMOOTH SECOND DERIVATIVES
C
   73 DO 78 MM=1,2
      TN=(TIMEAC(1)+TIMEAC(2))*.5
      DO 74 I=1,3
      RPPN(I)=(RPPAC(1,I)+RPPAC(2,I))*.5
   74 CONTINUE
      DO 78 K=3,NFIXES
      TO=TN
      TN=(TIMEAC(K-1)+TIMEAC(K))*.5
      DTK=TN-TIMEAC(K-1)
      DTKM=TIMEAC(K-1)-TO
      DTT=DTK+DTKM
      DO 78 I=1,3
      RPPO(I)=RPPN(I)
      RPPN(I)=(RPPAC(K-1,I)+RPPAC(K,I))*.5
      RPPAC(K-1,I)=(RPPO(I)*DTK+RPPN(I)*DTKM)/DTT
   78 CONTINUE
C
C   ADJUST RAC VALUES FOR SMOOTHED SECOND DERIVATIVES
C
      CALL SPLINV(TIMEAC,R0AC,RPPAC,100,3,NFIXES,RAC,TMARK)
C
C   PRINT OUT VALUES IN ORIGINAL UNITS
C
  150 WRITE (6,155) TIMTAB(ITIMZ),LTABL(IXUNIT),LTABL(IXUNIT),
     ALTABL(IYUNIT),LTABL(IYUNIT),HTABL(IZUNIT),HTABL(IZUNIT)
  155 FORMAT('0AIRCRAFT TRAJECTORY'/T6,'TIME',T16,'XIN',T23,'XSMOOTH',
     AT36,'YIN',T43,'YSMOOTH',T56,'ZIN',T63,'ZSMOOTH',T85,'XACC',T95,
     B'YACC',T105,'ZACC'/1X,A10,1X,6A10,T86,'G''S',T96,'G''S',T106,
     C 'G''S')
      DO 170 N=1,NFIXES
      DUMMY(1)=TIMCVR(DMOD(DBLE(TIMEAC(N)),86400.D0),2)
      DUMMY(2)=X0AC(N)/CLTABL(IXUNIT)
      DUMMY(3)=XAC(N)/CLTABL(IXUNIT)
      DUMMY(4)=Y0AC(N)/CLTABL(IYUNIT)
      DUMMY(5)=YAC(N)/CLTABL(IYUNIT)
      DUMMY(6)=Z0AC(N)
      IF(GEOPOT) DUMMY(6)=DUMMY(6)/(1.+DUMMY(6)/REARTH)
      DUMMY(6)=DUMMY(6)/CHTABL(IZUNIT)
      DUMMY(7)=ZAC(N)
      IF(GEOPOT) DUMMY(7)=DUMMY(7)/(1.+DUMMY(7)/REARTH)
      DUMMY(7)=DUMMY(7)/CHTABL(IZUNIT)
      DUMMY(8)=XPPAC(N)/G0
      DUMMY(9)=YPPAC(N)/G0
      DUMMY(10)=ZPPAC(N)/G0
      WRITE (6,160) DUMMY
  160 FORMAT(1X,F8.1,3(F10.1,F10.3),10X,3F10.5)
  170 CONTINUE
      RETURN
  20)0 WRITE (6,210
  210 FORMAT(' FILE 20, GIVING AIRCRAFT POSITION DATA, IS EMPTY. CANNOT
     APROCEED.')
      STOP 650
      END
C./       ADD   NAME=SPLINA
      SUBROUTINE SPLINA(ABSC,ORD,ORD2D,NABS,NORDS,NTERMS,WORK)
      DIMENSION ABSC(NABS),ORD(NABS,NORDS),ORD2D(NABS,NORDS),WORK(NABS)
      DO 10 NORD=1,NORDS
      ORD2D(1,NORD)=0.
      ORD2D(NTERMS,NORD)=0.
   10 CONTINUE
      WORK(1)=1.
      WORK(NTERMS)=1.
      IF(NTERMS.LE.2) RETURN
      NTRMN1=NTERMS-1
      DO 20 K=2,NTRMN1
      DTP1=ABSC(K+1)-ABSC(K)
      DTM1=ABSC(K)-ABSC(K-1)
      WORK(K)=2.*(DTP1+DTM1)
      DO 20 NORD=1,NORDS
      ORD2D(K,NORD)=6.*((ORD(K+1,NORD)-ORD(K,NORD))/DTP1
     A                 -(ORD(K,NORD)-ORD(K-1,NORD))/DTM1)
   20 CONTINUE
      IF(NTERMS.EQ.3) GO TO 70
      DO 60 K=3,NTRMN1
      DTM1=ABSC(K)-ABSC(K-1)
      FACT=DTM1/WORK(K-1)
      WORK(K)=WORK(K)-DTM1*FACT
      DO 60 NORD=1,NORDS
      ORD2D(K,NORD)=ORD2D(K,NORD)-FACT*ORD2D(K-1,NORD)
   60 CONTINUE
   70 DO 80 KK=2,NTRMN1
      K=NTERMS-KK+1
      DTP1=ABSC(K+1)-ABSC(K)
      FACT=1./WORK(K)
      DO 80 NORD=1,NORDS
      ORD2D(K,NORD)=(ORD2D(K,NORD)-DTP1*ORD2D(K+1,NORD))*FACT
   80 CONTINUE
      RETURN
      END
C./       ADD   NAME=SPLINV
      SUBROUTINE SPLINV(ABSC,ORDIN,ORD2D,NABS,NORDS,NTERMS,ORDOUT,WORK)
      DIMENSION ABSC(NABS),ORDIN(NABS,NORDS),ORD2D(NABS,NORDS)
      DIMENSION WORK(NABS),ORDOUT(NABS,NORDS)
      DO 10 NORD=1,NORDS
      ORDOUT(1,NORD)=ORDIN(1,NORD)
      ORDOUT(NTERMS,NORD)=ORDIN(NTERMS,NORD)
   10 CONTINUE
      WORK(1)=1.
      WORK(NTERMS)=1.
      IF(NTERMS.LE.2) RETURN
      NTRMN1=NTERMS-1
      DO 20 K=2,NTRMN1
      DTP1=ABSC(K+1)-ABSC(K)
      DTM1=ABSC(K)-ABSC(K-1)
      WORK(K)=-(1./DTP1+1./DTM1)
      CENT=2.*(DTP1+DTM1)
      DO 20 NORD=1,NORDS
      ORDOUT(K,NORD)=(DTM1*ORD2D(K-1,NORD)+CENT*ORD2D(K,NORD)
     A       +DTP1*ORD2D(K+1,NORD))/6.
   20 CONTINUE
      DTM1=ABSC(2)-ABSC(1)
      DO 30 NORD=1,NORDS
      ORDOUT(2,NORD)=ORDOUT(2,NORD)-ORDOUT(1,NORD)/DTM1
   30 CONTINUE
      IF(NTERMS.EQ.3) GO TO 70
      DO 60 K=3,NTRMN1
      DTM1=ABSC(K)-ABSC(K-1)
      FACT=1./(DTM1*WORK(K-1))
      WORK(K)=WORK(K)-FACT/DTM1
      DO 60 NORD=1,NORDS
      ORDOUT(K,NORD)=ORDOUT(K,NORD)-FACT*ORDOUT(K-1,NORD)
   60 CONTINUE
   70 DO 80 KK=2,NTRMN1
      K=NTERMS-KK+1
      DTP1=ABSC(K+1)-ABSC(K)
      FACT=1./WORK(K)
      DO 80 NORD=1,NORDS
      ORDOUT(K,NORD)=(ORDOUT(K,NORD)-ORDOUT(K+1,NORD)/DTP1)*FACT
   80 CONTINUE
      RETURN
      END
C./       ADD   NAME=TIMPHI
C  ****************************************************************
C  *** RAY TRACE ROUTINES - TIMPHI,ACMOVE,FILIMS,RAYORG,RAYTRK       ***
C  ***                     RATES,ADVANS,ARTUBE,RCRVIT,RECORD,RCSPCL ***
C  ****************************************************************
      SUBROUTINE TIMPHI(*)
      COMMON /PRINTS/ TITLE(30),KTPSIG,CVRTIM,TIMLBL
      REAL*8 TIMLBL,TIMCVR
      LOGICAL CVRTIM
      COMMON /RYCTRL/NORAYS,STND,UL,UR,LL,LR,PRTRAY,TIMBEG,DELTIM,NTIMS,
     APHIBEG,DELPHI,NPHIS
      LOGICAL*1 NORAYS,STND,UL,UR,LL,LR,PRTRAY,LOGIC(6)
      EQUIVALENCE (LOGIC(1),NORAYS)
      COMMON /FLIGHT/NFIXES,TIMEAC(100),XAC(100),YAC(100),ZAC(100),
     A XPPAC(100),YPPAC(100),ZPPAC(100)
      REAL DUMMY(6),DEFARY(6)/2*0.,10.,-90.,270.,10./
      REAL*8 KARD(9)
      INTEGER*2 MARGIN(6)/'AL','LL','LR','NO','UL','UR'/,KDUM(4)
      INTEGER MDEX(6)/6,5,6,1,3,4/
      REAL*8 PRTEST,DOPRNT(2)/'NOPRINT','PRINT'/
      READ(5,10,END=900) KARD
   10 FORMAT(9A8)
      WRITE(6,20) TITLE
   20 FORMAT('1',30A4)
      DEFARY(1)=TIMCVR(DBLE(TIMEAC(1)),2)
      DEFARY(2)=TIMCVR(DBLE(TIMEAC(NFIXES)),2)
      CALL FFA2N(KARD,1,8,3,DUMMY,DEFARY,KERR)
      DUMMY(1)=TIMCVR(DBLE(DUMMY(1)),1)
      DUMMY(2)=TIMCVR(DBLE(DUMMY(2)),1)
      TIMBEG=AMAX1(TIMEAC(1),AMIN1(DUMMY(1),DUMMY(2),TIMEAC(NFIXES)))
      TIMEND=AMIN1(TIMEAC(NFIXES),AMAX1(DUMMY(1),DUMMY(2),TIMEAC(1)))
      DELTIM=ABS(DUMMY(3))
      IF(DELTIM.LT.1E-7) GO TO 25
      NTIMS=(TIMEND-TIMBEG)/DELTIM +1.
      GO TO 30
   25 NTIMS=1
      DELTIM=0.
   30 CALL LJUST(8,1,KARD(7),PRTEST)
      PRTRAY=.TRUE.
      CALL LOOKUP(8,2,DOPRNT,PRTEST,KTEST,&33,&33)
      PRTRAY=KTEST.NE.1
   33 DUMMY(1)=TIMCVR(DBLE(TIMBEG),2)
      DUMMY(2)=TIMCVR(DBLE(TIMBEG+(NTIMS-1)*DELTIM),2)
      DUMMY(3)=DELTIM
      CALL FFA2N(KARD,33,8,2,DUMMY(5),DEFARY(5),KERR)
      DO 34 M=1,6
      LOGIC(M)=.FALSE.
   34 CONTINUE
      CALL LJUST(8,1,KARD(4),KDUM)
      STND=.TRUE.
      DO 37 L=1,4
      CALL LOOKUP(2,6,MARGIN,KDUM(L),KTEST,&37,&37)
      STND=.FALSE.
      IF(KTEST.NE.1) GO TO 36
```

```
      DO 35 M=3,6
      LOGIC(M)=.TRUE.
   35 CONTINUE
   36 LOGIC(MDEX(KTEST))=.TRUE.
   37 CONTINUE
      IF(NORAYS) GO TO 100
      IF(STND) GO TO 150
      DELPHI=ABS(DUMMY(6))
      NPHIS=1
      IF(DELPHI.EQ.0.) GO TO 40
      NPHIS=1.+ABS(DUMMY(5)/DELPHI)
   40 WRITE(6,45) UL,UR,LL,LR
   45 FORMAT('0INSTRUCTIONS -- PROCESS FOLLOWING MARGINS:'/'0',T30,'UL='
     A,L1,' UR=',L1,' LL=',L1,' LR=',L1)
      WRITE(6,55) DUMMY(1),DUMMY(2),TIMLBL,DUMMY(3)
   55 FORMAT('0ORIGIN TIMES FROM',F12.0,' TO',F12.0,' (',A6,') IN INCREM
     AENTS OF',F5.0,' SECONDS.')
      DUMMY(5)=ABS(DUMMY(5))
      WRITE(6,60) DUMMY(5),DELPHI
   60 FORMAT('0COVER',F7.2,' DEGREES IN INCREMENTS OF',F7.2,' DEGREES.')
      GO TO 200
  100 WRITE(6,105)
  105 FORMAT('0INSTRUCTIONS -- AIRCRAFT MANUVERS ONLY. PROCESS NO RAYS.'
     A)
      WRITE(6,55) DUMMY(1),DUMMY(2),TIMLBL,DUMMY(3)
      GO TO 200
  150 CALL FFA2N(KARD,25,8,1,DUMMY(4),DEFARY(4),KERR)
      PHIBEG=DUMMY(4)
      DELPHI=DUMMY(6)
      IF(ABS(DELPHI).LT.1.E-7) GO TO 160
      S360=SIGN(360.,DELPHI)
      NPHIS=(AMOD(AMOD(DUMMY(5)-DUMMY(4),S360)-S360,S360)+S360)/DELPHI
     A +1.
      GO TO 170
  160 NPHIS=1
      DELPHI=0.
  170 DUMMY(4)=PHIBEG
      DUMMY(5)=PHIBEG+(NPHIS-1)*DELPHI
      DUMMY(6)=DELPHI
      WRITE(6,180)
  180 FORMAT('0INSTRUCTIONS -- PROCESS PENCIL OF RAYS')
      WRITE(6,55)DUMMY(1),DUMMY(2),TIMLBL,DUMMY(3)
      WRITE(6,190)(DUMMY(I),I=4,6)
  190 FORMAT('0ORIGIN PHI-ANGLES FROM',F8.2,' TO',F8.2,' DEGREES IN INCR
     CS OF',F7.2,' DEGREES.')
  200 IF(PRTRAY) WRITE(6,210)
      IF(.NOT.PRTRAY) WRITE(6,220)
      RETURN
  210 FORMAT('0PRINT RAY TRACINGS')
  220 FORMAT('0DO NOT PRINT RAY TRACINGS')
  900 RETURN 1
      END
C./       ADD   NAME=ACMOVE
      SUBROUTINE ACMOVE(T)
      COMMON /PRINTS/ TITLE(30),KTPSIG,CVRTIM,TIMLBL
      REAL*8 TIMLBL,TIMCVR
      LOGICAL CVRTIM
      COMMON /ACSPOT/TIME,XR0,YR0,ZR0,XDOT,YDOT,ZDOT,AIRSPD,ASPDOT,
     AC0,U0,V0,CDOT,XMACH,XMADOT,XMU,XMUDOT,COSMU,SINMU,EK(3,3),
     B EKDOT(3,3),GLOAD,HEADIN,CLIMB,BANK
      DIMENSION R0(3),RDOT(3),RDDOT(3),RLWDOT(3),OMEGA(3)
      EQUIVALENCE(R0(1),XR0),(XDOT,RDOT(1))
      COMMON /ATMCON/ REARTH,G0,RSTAR,R0M0,R0G0M0
      COMMON /FLIGHT/NFIXES,TIMEAC(100),XAC(100),YAC(100),ZAC(100),
     A XPPAC(100),YPPAC(100),ZPPAC(100)
      DIMENSION RAC(100,3),RPPAC(100,3)
      EQUIVALENCE (RAC(1,1),XAC(1)),(RPPAC(1,1),XPPAC(1))
      COMMON /ATMSPH/ GAM,C,U,V,DCDZ,DUDZ,DVDZ,D2CDZ2,D2UDZ2,D2VDZ2,RHO
      REAL*8 GAM,C,U,V
      INTEGER SKEW(4)/2,3,1,2/
      DATA DGPRAD/57.295780/
      TIME=T
      CALL GETLYR(T,TIMEAC,NFIXES,NODE,&200)
      TAU=T-TIMEAC(NODE)
      CTAU=TIMEAC(NODE+1)-T
      DELTAT=TAU+CTAU
      DT2=DELTAT**2/6.
      TAU2=TAU*TAU/2.
      TAU3=TAU2*TAU/3.
      CTAU2=CTAU*CTAU/2.
      CTAU3=CTAU2*CTAU/3.
      DO 10 I=1,3
      GN=RAC(NODE,I)-RPPAC(NODE,I)*DT2
      GNC=RAC(NODE+1,I)-RPPAC(NODE+1,I)*DT2
      R0(I)=(GN*CTAU+GNC*TAU+RPPAC(NODE+1,I)*TAU3+RPPAC(NODE,I)*CTAU3)
     A                                        /DELTAT
      RDOT(I)=(GNC-GN+RPPAC(NODE+1,I)*TAU2-RPPAC(NODE,I)*CTAU2)/DELTAT
      RDDOT(I)=(RPPAC(NODE+1,I)*TAU+RPPAC(NODE,I)*CTAU)/DELTAT
   10 CONTINUE
      CALL FNDLYR(ZR0,&250)
      CALL AIR(DBLE(ZR0))
      C0=C
      U0=U
      V0=V
      CDOT=DCDZ*ZDOT
      UAS=XDOT-U
      VAS=YDOT-V
      ASPH=UAS**2+VAS**2
      AIRSPD=SQRT(ASPH+ZDOT**2)
      ASPH=SQRT(ASPH)
      RLWDOT(1)=RDDOT(1)-DUDZ*ZDOT
      RLWDOT(2)=RDDOT(2)-DVDZ*ZDOT
      RLWDOT(3)=RDDOT(3)
      ASPDOT=(RLWDOT(1)*UAS+RLWDOT(2)*VAS+ZDOT*RLWDOT(3))/AIRSPD
      XMACH=AIRSPD/C0
      XMADOT=(ASPDOT*C0-AIRSPD*CDOT)/C0**2
      IF(XMACH.GT.1.) GO TO 15
      XMU=90.
      XMUDOT=0.
      SINMU=1.
      COSMU=0.
      GO TO 20
   15 SINMU=1./XMACH
      COSMU=SQRT(1.-SINMU**2)
      XMU=DGPRAD*ASIN(SINMU)
      XMUDOT=-DGPRAD*XMADOT*SINMU**2/COSMU
   20 EK(1,1)=UAS/AIRSPD
      EK(2,1)=VAS/AIRSPD
      EK(3,1)=ZDOT/AIRSPD
      EK(1,2)=VAS/ASPH
      EK(2,2)=-UAS/ASPH
      EK(3,2)=0.
      DO 30 K=1,3
      K1=SKEW(K)
      K2=SKEW(K+1)
      EK(K,3)=EK(K1,2)*EK(K2,1)-EK(K2,2)*EK(K1,1)
      OMEGA(K)=(RLWDOT(K1)*EK(K2,1)-RLWDOT(K2)*EK(K1,1))/AIRSPD
   30 CONTINUE
      FACT=(OMEGA(1)*EK(1,1)+OMEGA(2)*EK(2,1))/(EK(1,1)**2+EK(2,1)**2)
      HLOAD=0.
      VLOAD=G0*EK(3,3)/(1.+ZR0/REARTH)**2
      DO 40 K=1,3
      OMEGA(K)=OMEGA(K)-FACT*EK(K,1)
      HLOAD=HLOAD+RDDOT(K)*EK(K,2)
      VLOAD=VLOAD+RDDOT(K)*EK(K,3)
   40 CONTINUE
      GLOAD=SQRT(HLOAD**2+VLOAD**2)/G0
      BANK=DGPRAD*ATAN2(HLOAD,VLOAD)
      HEADIN=DGPRAD*ATAN2(-EK(1,1),-EK(2,1))+180.
      CLIMB=DGPRAD*ASIN(EK(3,1))
      DO 50 K=1,3
      K1=SKEW(K)
      K2=SKEW(K+1)
      OM1=OMEGA(K1)
      OM2=OMEGA(K2)
      DO 50 L=1,3
      EKDOT(K,L)=EK(K1,L)*OM2-EK(K2,L)*OM1
   50 CONTINUE
      WRITE(6,60) TITLE
   60 FORMAT('1',30A4)
      TPRINT=TIMCVR(DBLE(TIME),2)
      WRITE(6,65)TIMLBL,TPRINT,XR0,YR0,ZR0,XMACH,GLOAD,BANK
   65 FORMAT('0AIRCRAFT MANEUVER DATA'/'0',T4,'TIME',T16,'X',T26,'Y',T36
     A,'Z',T45,'MACH',T54,'LOAD',T64,'BANK'/T4,A8,T15,'MET',T25,'MET'
     B,T35,'MET',T46,'NO.',T55,'G''S',T64,'DEGS.'/1X,4F10.0,2F10.5,F10.1
     C )
      HEADG=DGPRAD*ATAN2(-XDOT,-YDOT)+180.
      GNTSPD=SQRT(XDOT**2+YDOT**2+ZDOT**2)
      GCLMB=DGPRAD*ASIN(ZDOT/GNTSPD)
      WRITE(6,70)AIRSPD,UAS,VAS,ZDOT,CLIMB,HEADIN,GNTSPD,XDOT,YDOT,ZDOT,
     AGCLMB,HEADG
   70 FORMAT('0',T11,10('-'),T25,'SPEED MPS',T41,10('-'),1X,5('-'),2X,'A
     ANGLE',3X,5('-')/T14,'TOTAL',T23,'X-COMP',T33,'Y-COMP',T43,'Z-COMP'
     B,T54,'CLIMB',T64,'HEADING'/' AIR',T11,4F10.0,F10.2,F10.1/' GROUND'
     C,T11,4F10.0,F10.2,F10.1)
      RETURN
  200 WRITE(6,210) T,TIMEAC(1),TIMEAC(NFIXES)
  210 FORMAT(' IN CALL TO ACMOVE, TIME''',F10.1,''' IS OUTSIDE RANGE''',
     AF10.1,''' TO''',F10.1)
      RETURN
  250 WRITE(6,260) T,ZR0
  260 FORMAT(' IN CALL TO ACMOVE AT TIME''',F10.1,''' AIRCRAFT IS AT ALT
     AITUDE Z='',F10.2,''' METERS AND OUTSIDE ATMOSPHERE TABLE.')
      STOP 600
      END
C./       ADD   NAME=FILIMS
      SUBROUTINE FILIMS
      COMMON /ACSPOT/TIME,XR0,YR0,ZR0,XDOT,YDOT,ZDOT,AIRSPD,ASPDOT,
     AC0,U0,V0,CDOT,XMACH,XMADOT,XMU,XMUDOT,COSMU,SINMU,EK(3,3),
     B EKDOT(3,3),GLOAD,HEADIN,CLIMB,BANK
      DATA DGPRAD/57.295780/,TWOPI/6.28318531/
      COMMON /GROUND/ ZGRND,CGRND,UGRND,VGRND,REFLFC
      COMMON /RAYLIM/ NLIMS,BEG(2),END(2)
      DIMENSION ZRO(5),TRND(5)
      EQUIVALENCE (SINGAM,EK(3,1)),(COSGAM,EK(3,3))
      IF(XMACH.LT.1.) GO TO 102
      U0MG=U0-UGRND
      V0MG=V0-VGRND
      ALPHA1=1.+SINMU*(U0MG*EK(1,1)+V0MG*EK(2,1))/C0
      ALPHA2=COSMU*(U0MG*EK(1,2)+V0MG*EK(2,2))/C0
      ALPHA3=COSMU*(U0MG*EK(1,3)+V0MG*EK(2,3))/C0
      A0=ALPHA1**2+.5*(ALPHA2**2+ALPHA3**2)
      A1=-2.*ALPHA1*ALPHA3
      A2=-2.*ALPHA1*ALPHA2
      A3=.5*(ALPHA3**2-ALPHA2**2)
      A4=ALPHA2*ALPHA3
      SINMU2=SINMU**2
      COSMU2=COSMU**2
      CGFACT=(CGRND/C0)**2
      CSGM2=COSGAM**2
      A0=A0-CGFACT*(SINMU2*CSGM2+(1.-.5*CSGM2)*COSMU2)
      A1=A1-CGFACT*2.*SINGAM*COSGAM*SINMU*COSMU
      A3=A3+CGFACT*.5*(CSGM2)*COSMU2
      A34=SQRT(A3**2+A4**2)
      DFULIM=SQRT(A1**2+A2**2)+A34
      CPPMX=DFULIM+3.*A34
      EPS=5E-6*DFULIM
      PHI=-90./DGPRAD
      FO=A0-A2-A3
      PHIO=PHI
```

```
      PHIBEG=PHI
      KZRO=1
      IF(DFULIM.LT.ABS(A0).OR.DFULIM.EQ.0.) GO TO 100
    5 IF(PHIO.GT.PHIBEG+TWOPI) GO TO 100
      SINPHI=SIN(PHI)
      COSPHI=COS(PHI)
      TWOPHI=PHI+PHI
      COS2FI=COS(TWOPHI)
      SIN2FI=SIN(TWOPHI)
      F=A0+A1*COSPHI+A2*SINPHI+A3*COS2FI+A4*SIN2FI
      IF(ABS(F).LT.EPS) GO TO 25
      IF(F*FO.LE.0.) GO TO 10
C  CASE NO ZERO CROSSING.  ADVANCE PHI
      FPR=-A1*SINPHI+A2*COSPHI-2.*(A3*SIN2FI-A4*COS2FI)
      FPPMX=SIGN(CPPMX,F)
      DPHI1=FPR/FPPMX
      DPHI2=SQRT(FPR**2+2.*F*FPPMX)/CPPMX
      DPHI=AMAX1(ABS(DPHI1),DPHI2+DPHI1,1.E-5)
      PHIO=PHI
      FO=F
      PHI=PHI+DPHI
     @Ga TO 5
C   CASE ZERO IS CROSSED.  LOCATE ZERO BY HALVES.
   10 PHIHI=PHI
      FHI=F
   15 PHI=.5*(PHIHI+PHIO)
      SINPHI=SIN(PHI)
      COSPHI=COS(PHI)
      TWOPHI=PHI+PHI
      COS2FI=COS(TWOPHI)
      SIN2FI=SIN(TWOPHI)
      F=A0+A1*COSPHI+A2*SINPHI+A3*COS2FI+A4*SIN2FI
      IF(ABS(F).LT.EPS) GO TO 25
      IF(F*FHI.GT.0.) GO TO 20
      PHIO=PHI
      FO=F
      GO TO 15
   20 PHIHI=PHI
      FHI=F
      GO TO 15
   25 FPR=-A1*SINPHI+A2*COSPHI-2.*(A3*SIN2FI-A4*COS2FI)
      DPHI=ABS(FPR/CPPMX)
      DF=.5*ABS(FPR)*DPHI
      IF(DF.LT.EPS) GO TO 30
      MULT=1
      SGN=SIGN(1.,FPR)
      GO TO 50
   30 FPPR=-A1*COSPHI-A2*SINPHI-4.*(A3*COS2FI+A4*SIN2FI)
      DPHI=ABS(2.*FPPR)/(CPPMX+4.*A34)
      DF=ABS(FPPR)*DPHI*DPHI/6.
      IF(DF.LT.EPS) GO TO 35
      MULT=2
      SGN=-SIGN(1.,FPPR)
      GO TO 50
   35 FP3R=A1*SINPHI-A2*COSPHI+8.*(A3*SIN2FI-A4*COS2FI)
      DPHI=ABS(3.*FP3R)/(CPPMX+12.*A34)
      DF=ABS(FP3R)*DPHI*DPHI*DPHI/24.
      IF(DF.LT.EPS) GO TO 40
      MULT=3
      SGN=SIGN(1.,FP3R)
      GO TO 50
   40 FP4R=A1*COSPHI+A2*SINPHI+16.*(A3*COS2FI+A4*SIN2FI)
      MULT=4
      SGN=-SIGN(1.,FP4R)
      DPHI=TWOPI
   50 DO 55 K=1,MULT
      ZRO(KZRO)=PHI
      TRND(KZRO)=SGN
      SGN=-SGN
      KZRO=KZRO+1
      IF(KZRO.GT.5) GO TO 100
   55 CONTINUE
      PHIO=PHI+DPHI
      PHI=PHIO
      TWOPHI=PHI+PHI
      FO=A0+A1*COS(PHI)+A2*SIN(PHI)+A3*COS(TWOPHI)+A4*SIN(TWOPHI)
      GO TO 5
  100 IF(KZRO.GT.1) GO TO 110
      IF(FO.GE.0) GO TO 105
  102 NLIMS=0
      GO TO 130
  105 NLIMS=1
      BEG(1)=-90.
      END(1)=270.
      GO TO 130
  110 KZRO=KZRO-1
      IF(MOD(KZRO,2).EQ.1) GO TO 115
      KZRO=KZRO+1
      ZRO(KZRO)=ZRO(1)+TWOPI
      TRND(KZRO)=TRND(1)
  115 NLIMS=(KZRO-1)/2
      L=1
      IF(TRND(1).LT.0.) L=2
      DO 120 N=1,NLIMS
      BEG(N)=ZRO(N*2+L-2)*DGPRAD
      END(N)=ZRO(N*2+L-1)*DGPRAD
  120 CONTINUE
  130 IF(NLIMS.GT.0) GO TO 150
      WRITE(6,145)
  145 FORMAT('0 RAYS WILL NOT TOUCH GROUND OR AIRCRAFT IS SUBSONIC.')
      RETURN
  150 WRITE(6,155) NLIMS
  155 FORMAT('0',T10,I2,' PHI-ANGLE INTERVALS:')
      DO 165 N=1,NLIMS
      BEG1=AMOD(AMOD(BEG(N),360.)+450.,360.)-90.
      END1=AMOD(AMOD(END(N),360.)+450.,360.)-90.
      WRITE(6,160) N,BEG1,END1
  160 FORMAT('0INTERVAL',I2,'   FROM',F7.2,' DEGREES TO',F7.2,' DEGREES.
     A')
  165 CONTINUE
      RETURN
      END
C./       ADD   NAME=RAYORG
      SUBROUTINE RAYORG(*)
      COMMON /RYCTRL/NORAYS,STND,UL,UR,LL,LR,PRTRAY,TIMBEG,DELTIM,NTIMS,
     APHIBEG,DELPHI,NPHIS
      LOGICAL*1 NORAYS,STND,UL,UR,LL,LR,PRTRAY
      COMMON /CLASES/CNAMES(30),NRCURV(2,2),TYPRAY(3,2,2),DIRECT,LOFT,
     A UP,DOWN
      REAL*8 CNAMES
      LOGICAL *1 TYPRAY,DIRECT,LOFT,UP,DOWN
      COMMON /PRINTS/ TITLE(30),KTPSIG,CVRTIM,TIMLBL
      REAL*8 TIMLBL
      LOGICAL CVRTIM
      COMMON /ATMSPH/ GAM,C,U,V,DCDZ,DUDZ,DVDZ,D2CDZ2,D2UDZ2,D2VDZ2,RHO
      REAL*8 GAM,C,U,V
      COMMON /ACSPOT/TIME,XR0,YR0,ZR0,XDOT,YDOT,ZDOT,AIRSPD,ASPDOT,
     AC0,U0,V0,CDOT,XMACH,XMADOT,XMU,XMUDOT,COSMU,SINMU,EK(3,3),
     B EKDOT(3,3),GLOAD,HEADIN,CLIMB,BANK
      REAL XKR0(3),XKDOT(3)
      EQUIVALENCE(XKR0(1),XR0),(XKDOT(1),XDOT)
      COMMON /RAYNIT/ KGMH,NDCRVS,NUCRVS,IUPDWN,T0,PHI0,X0,Y0,Z0,
     AP10,P20,P30,OMEGA,DELTA0,P1F0,P2F0,P3F0,OMEGAF,XT0,YT0,ZT0,
     BP1T0,P2T0,P3T0,OMEGAT,XS0,YS0,ZS0,P3S0,RHO0,PCONST,NAGES,AGES(20)
      INTEGER*2 KGMH,NDCRVS,NUCRVS,IUPDWN
      REAL XK0(3),PK0(3),PKF0(3),XKT0(3),PKT0(3),XKS0(3)
      EQUIVALENCE(XK0(1),X0),(PK0(1),P10),(PKF0(1),P1F0),(XKT0(1),XT0)
      EQUIVALENCE(PKT0(1),P1T0),(XKS0(1),XS0)
      COMMON /RAYVAR/ZDIR,PKK,RTPAA0,ATTEN,SIGMA,X,Y,Z,DAGE,XF,YF,ZF,
     AXT,YT,ZT,XS,YS,ZS,XSS,YSS,ZSS,XSSS,YSSS,ZSSS,P3,P3F,P3T,P3S,
     BXFS,YFS,XTS,YTS,ZFTP3,XFTZ,YFTZ,ZFTZ,ZFA,ZTA,P3FTZ,P3FA,P3TA,AREA,
     CDAGDS
      REAL*8 SIGMA,X,Y,Z,DAGE,XF,YF,ZF,XT,YT,ZT
      REAL XKS(3),XKFS(2),XKTS(2),XKFTZ(3),XKSS(3),XKSSS(3)
      REAL*8 XK(3),XKF(3),XKT(3)
      EQUIVALENCE (XK(1),X),(XKF(1),XF),(XKT(1),XT),(XKS(1),XS)
      EQUIVALENCE (XKFS(1),XFS),(XKTS(1),XTS),(XKFTZ(1),XFTZ)
      EQUIVALENCE (XKSS(1),XSS),(XKSSS(1),XSSS)
      REAL*8 RENORM
      DATA DGPRAD/57.295780/
      NAGES=1
      DAGE=0.D0
      MEDHI=1
      COSPHI=COS(PHI0/DGPRAD)
      SINPHI=SIN(PHI0/DGPRAD)
      RENORM=0.D0
      DO 10 K=1,3
      XK0(K)=XKR0(K)
      EH=-SINPHI*EK(K,2)-COSPHI*EK(K,3)
      EHDOT=-SINPHI*EKDOT(K,2)-COSPHI*EKDOT(K,3)
      PK0(K)=EK(K,1)+COSMU*EH/SINMU
      PKF0(K)=(COSMU*(-COSPHI*EK(K,2)+SINPHI*EK(K,3)))/SINMU
      PKT0(K)=EKDOT(K,1)+(EHDOT*COSMU-EH*XMUDOT/(DGPRAD*SINMU))/SINMU
      RENORM=RENORM+DBLE(PK0(K))**2
   10 CONTINUE
      RENORM=SINMU*DSQRT(RENORM)
      CALL FNDLYR(Z0,&20)
   20 CALL AIR(DBLE(Z0))
      RHO0=RHO
      PCONST=AIRSPD**2*SQRT(.5*RHO0)
      DELTA0=AIRSPD
      CSQD=C0*SINMU
      DO 25 K=1,3
      PK0(K)=PK0(K)/RENORM
      XKS0(K)=CSQD*PK0(K)
      XKT0(K)=XKDOT(K)-XKS0(K)
   25 CONTINUE
      XS0=XS0+U0
      YS0=YS0+V0
      XT0=XT0-U0
      YT0=YT0-V0
      RTPAA0=SQRT(P10**2+P20**2)
      DELTA0=C*SQRT(P30**2+RTPAA0**2)
      OMEGA=DELTA0+U*P10+V*P20
      OMEGAT=ASPDOT+(DUDZ*P10+DVDZ*P20)*ZDOT+U*P1T0+V*P2T0
      OMEGAF=U*P1F0+V*P2F0
      P3S0=-DCDZ*DELTA0/C0-P10*DUDZ-P20*DVDZ
      P3T0=P3T0-P3S0
      IUPDWN=1
      IF(P30.LT.0.) IUPDWN=2
      IF(IUPDWN.EQ.1.AND..NOT.UP)RETURN 1
      IF(IUPDWN.EQ.2.AND..NOT.DOWN) RETURN 1
      SIGMA=T0
      ZDIR=SIGN(1.,P30)
      DO 50 K=1,3
      XK(K)=XK0(K)
      XKF(K)=0.D0
      XKT(K)=XKT0(K)
   50 CONTINUE
      P3F=P3F0
      P3T=P3T0
      CALL RATES(&100,&100)
      AREA=0.
      ATTEN=1.
      IF(.NOT.PRTRAY) RETURN
      WRITE(6,60) TITLE
   60 FORMAT('1',30A4)
      TPRINT=TIMCVR(DBLE(T0),2)
      FIPRNT=AMOD(AMOD(PHI0,360.)+450.,360.)-90.
      AZIM=PHAZIM(0.)
```

```
      WRITE(6,65) TPRINT,TIMLBL,FIPRNT,P10,P20,AZIM
   65 FORMAT('0',T20,'DATA FOR RAY DEPARTING AIRCRAFT TIME=',F10.0,1X,A8
     A,'PHI ANGLE=',F7.2,' DEGREES.'/T15,'P1=',G14.5,',P2=',G14.5,'PHASE
     B NORMAL AZIMUTH=',F6.0,' DEGREES.')
      TPRINT=TIMCVR(SIGMA,2)
      ELEV=PHELEV(0.)
      WRITE(6,70) TIMLBL,TPRINT,X,Y,Z,P3,ELEV,C,ZS,AREA,DAGE
   70 FORMAT('0',T5,'SIGMA',T18,'X',T28,'Y',T39,'Z',T45,'P3',T53,'PHASE'
     A,T65,'C',T73,'DZ/DS',T81,'AREA',T91,'AGE'/T5,A8,T17,'MET',T27,
     B'MET',T38,'MET',T54,'ELEV',T63,'M/SEC',T73,'M/SEC',T79,'M**2/SEC',
     CT89,'MET**.5'/
     D1X,F10.1,3F10.0,G10.3,F6.1,2F10.1,2G10.4)
      RETURN
  100 WRITE(6,101)
  101 FORMAT(' IMPROPER RETURN FROM RATES IN RAYORG')
      RETURN
      END
C./       ADD   NAME=RAYTRK
      SUBROUTINE RAYTRK
      COMMON /CLASES/CNAMES(30),NRCURV(2,2),TYPRAY(3,2,2),DIRECT,LOFT,
     A UP,DOWN
      REAL*8 CNAMES
      LOGICAL *1 TYPRAY,DIRECT,LOFT,UP,DOWN
      COMMON /RYCTRL/NORAYS,STND,UL,UR,LL,LR,PRTRAY,TIMBEG,DELTIM,NTIMS,
     APHIBEG,DELPHI,NPHIS
      LOGICAL*1 NORAYS,STND,UL,UR,LL,LR,PRTRAY
      COMMON /RAYNIT/ KGMH,NDCRVS,NUCRVS,IUPDWN,T0,PHI0,X0,Y0,Z0,
     AP10,P20,P30,OMEGA,DELTA0,P1F0,P2F0,P3F0,OMEGAF,XT0,YT0,ZT0,
     BP1T0,P2T0,P3T0,OMEGAT,XS0,YS0,ZS0,P3S0,RHO0,PCONST,NAGES,AGES(20)
      INTEGER*2 KGMH,NDCRVS,NUCRVS,IUPDWN
      REAL PK(2),PKF(2),PKT(2)
      EQUIVALENCE (PK(1),P10),(PKF(1),P1F0),(PKT(1),P1T0)
      COMMON /RAYVAR/ZDIR,PKK,RTPAA0,ATTEN,SIGMA,X,Y,Z,DAGE,XF,YF,ZF,
     AXT,YT,ZT,XS,YS,ZS,XSS,YSS,ZSS,XSSS,YSSS,ZSSS,P3,P3F,P3T,P3S,
     BXFS,YFS,XTS,YTS,ZFTP3,XFTZ,YFTZ,ZFTZ,ZFA,ZTA,P3FTZ,P3FA,P3TA,AREA,
     CDAGDS
      REAL*8 SIGMA,X,Y,Z,DAGE,XF,YF,ZF,XT,YT,ZT
      REAL XKS(3),XKFS(2),XKTS(2),XKFTZ(3),XKSS(3),XKSSS(3)
      REAL*8 XK(3),XKF(3),XKT(3)
      EQUIVALENCE (XK(1),X),(XKF(1),XF),(XKT(1),XT),(XKS(1),XS)
      EQUIVALENCE (XKFS(1),XFS),(XKTS(1),XTS),(XKFTZ(1),XFTZ)
      EQUIVALENCE (XKSS(1),XSS),(XKSSS(1),XSSS)
      COMMON /ATMSPH/ GAM,C,U,V,DCDZ,DUDZ,DVDZ,D2CDZ2,D2UDZ2,D2VDZ2,RHO
      REAL*8 GAM,C,U,V
      COMMON /GROUND/ ZGRND,CGRND,UGRND,VGRND,REFLFC
      COMMON /LYRDEF/NLAYER,GMZA(200),INDPTH(200),INDWND(200),
     ALYRPRT(200),KLAYER,ZTOP,ZBOT
      INTEGER*2 INDPTH,INDWND
      LOGICAL*1 LYRPRT
      REAL*8 TIMCVR,TPRINT
      COMMON /RAYHLD/HSIGMA,HX,HY,HZ,HDAGE,HXF,HYF,HZF,HXT,HYT,HZT,
     AHXS,HYS,HZS,HXSS,HYSS,HZSS,HXSSS,HYSSS,HZSSS,HP3,HP3F,HP3T,HP3S,
     BHXFS,HYFS,HXTS,HYTS,HZFTP3,HXFTZ,HYFTZ,HZFTZ,HZFA,HZTA,HP3FTZ,
     CHP3FA,HP3TA,HAREA,HDAGDS
      REAL*8 HSIGMA,HX,HY,HZ,HDAGE,HXF,HYF,HZF,HXT,HYT,HZT
      REAL HXKS(3),HXKFS(2),HXKTS(2),HXKFTZ(3),HXKSS(3),HXKSSS(3)
      REAL*8 HXK(3),HXKF(3),HXKT(3)
      EQUIVALENCE (HXK(1),HX),(HXKF(1),HXF),(HXKT(1),HXT),(HXKS(1),HXS)
      EQUIVALENCE (HXKFS(1),HXFS),(HXKTS(1),HXTS),(HXKFTZ(1),HXFTZ)
      EQUIVALENCE (HXKSS(1),HXSS),(HXKSSS(1),HXSSS)
      REAL HOLDVR(28),HOLDHD(28)
      REAL*8 H8VR(11),H8HD(11),DELZ
      EQUIVALENCE (HOLDVR(1),XS),(HOLDHD(1),HXS)
      EQUIVALENCE (H8VR(1),SIGMA),(H8HD(1),HSIGMA)
      NDCRVS=0
      NUCRVS=0
      KGMH=1
    1 CONTINUE
      DO 2 L=1,28
      HOLDHD(L)=HOLDVR(L)
    2 CONTINUE
      DO 3 L=1,11
      H8HD(L)=H8VR(L)
    3 CONTINUE
      TDLSIG=.30
      IF(ZDIR.GT.0.) GO TO 10
      IF(ZS+ZSS*TDLSIG.GT.0.) TDLSIG=AMAX1(0.,-ZS/ZSS)
      DELZ=DMAX1(-50.D0,ZBOT-Z,DMIN1(-1.D0,(ZS+.5D0*ZSS*TDLSIG)*TDLSIG))
      IF(DELZ.LT.0.D0) GO TO 15
      LPRNT=KLAYER
      KLAYER=KLAYER-1
      IF(KLAYER.LE.0) GO TO 450
      GO TO 400
   10 IF(ZS+ZSS*TDLSIG.LT.0.) TDLSIG=AMAX1(0.,-ZS/ZSS)
      DELZ=DMIN1(50.D0,ZTOP-Z,DMAX1(1.D0,(ZS+.5D0*ZSS*TDLSIG)*TDLSIG))
      IF(DELZ.GT.0.D0) GO TO 15
      KLAYER=KLAYER+1
      LPRNT=KLAYER
      IF(KLAYER.GE.NLAYER) GO TO 500
      GO TO 400
   15 Z=HZ+DELZ
      CALL RATES(&320,&300)
      CALL ADVANS
      GO TO 1
  300 CALL RCRVIT
  305 CALL RATES(&320,&420)
  320 CALL ADVANS
      IF(.NOT.PRTRAY) GO TO 315
      WRITE(6,310)
  310 FORMAT(' RECURVATURE POINT ATTAINED.')
      TPRINT=TIMCVR(SIGMA,2)
      ELEV=PHELEV(0.)
      WRITE(6,60) TPRINT,X,Y,Z,P3,ELEV,C,ZS,AREA,DAGE
  315 IF(ZDIR.LT.0.) GO TO 350
      ZDIR=-1.
      NDCRVS=NDCRVS+1
      KGMH=2
      IF(Z.GT.70E3) KGMH=3
      CALL RCSPCL('RAY HIGH',SIGMA,XK,P3,XKF,XKT,XKS,AREA)
      IF(NDCRVS.GT.NRCURV(3-IUPDWN,KGMH-1)) RETURN
      GO TO 1
  350 IF(Z.LE.ZGRND) GO TO 450
      NUCRVS=NUCRVS+1
      CALL RCSPCL('RAY LOW ',SIGMA,XK,P3,XKF,XKT,XKS,AREA)
      IF(PRTRAY)WRITE(6,355)
  355 FORMAT(' RAY RECURVING UPWARD; WILL NEVER TOUCH GROUND.')
      IF(Z-ZGRND.GE.1.) RETURN
      GO TO (370,380,380),KGMH
  370 IF(LOFT) GO TO 480
      RETURN
  380 IF(NDCRVS.GE.NRCURV(3-IUPDWN,KGMH-1)) RETURN
      GO TO 480
  400 ZBOT=GMZA(KLAYER)
      ZTOP=GMZA(KLAYER+1)
      CALL RATES(&410,&420)
      FCTJMP=(P3S-HP3S)/ZS
      P3F=HP3F+ZF*FCTJMP
      P3T=HP3T+ZT*FCTJMP
      IF(.NOT.(LYRPRT(LPRNT).AND.PRTRAY)) GO TO 1
      TPRINT=TIMCVR(SIGMA,2)
      ELEV=PHELEV(0.)
      WRITE(6,60) TPRINT,X,Y,Z,P3,ELEV,C,ZS,AREA,DAGE
   60 FORMAT(1X,F10.1,3F10.0,G10.3,F6.1,2F10.1,2G10.4)
      GO TO 1
  410 IF(ZDIR.LT.0.) GO TO 350
      KLAYER=KLAYER-1
      ZBOT=GMZA(KLAYER)
      ZTOP=GMZA(KLAYER+1)
      GO TO 305
  420 WRITE(6,421)
  421 FORMAT(' IMPROPER RETURN FROM RATES IN RAYTRK')
      RETURN
  450 TPRINT=TIMCVR(SIGMA,2)
      ELEV=PHELEV(0.)
      IF(PRTRAY)WRITE(6,60) TPRINT,X,Y,Z,P3,ELEV,C,ZS,AREA,DAGE
      NUCRVS=NUCRVS+1
      CALL RCSPCL(' GROUND ',SIGMA,XK,P3,XKF,XKT,XKS,AREA)
      CALL RECORD(&480)
      RETURN
  480 ZDIR=1.
      KLAYER=1
      IF(ZS.EQ.0.) GO TO 495
      FCTJMP=2.*HP3S/ZS
      P3F=-HP3F+FCTJMP*HZF
      P3T=-HP3T+FCTJMP*HZT
      ZF=-HZF
      ZT=-HZT
      AREA=-HAREA
      ATTEN=ATTEN*REFLFC
      CALL RATES(&495,&420)
      IF(PRTRAY)WRITE(6,490)
  490 FORMAT(' **************    REFLECTING FROM GROUND    ***********')
      GO TO 1
  495 IF(PRTRAY)WRITE (6,496)
  496 FORMAT(' ********RAY TANGENT AT GROUND LEVEL*********')
      GO TO 1
  500 IF(PRTRAY)WRITE(6,505)
  505 FORMAT(' STOPPING AT TOP OF ATMOSPHERE.')
      RETURN
      END
C./       ADD   NAME=RATES
      SUBROUTINE RATES(*,*)
      COMMON /RAYNIT/ KGMH,NDCRVS,NUCRVS,IUPDWN,T0,PHI0,X0,Y0,Z0,
     AP10,P20,P30,OMEGA,DELTA0,P1F0,P2F0,P3F0,OMEGAF,XT0,YT0,ZT0,
     BP1T0,P2T0,P3T0,OMEGAT,XS0,YS0,ZS0,P3S0,RHO0,PCONST,NAGES,AGES(20)
      INTEGER*2 KGMH,NDCRVS,NUCRVS,IUPDWN
      REAL PK(2),PKF(2),PKT(2)
      EQUIVALENCE (PK(1),P10),(PKF(1),P1F0),(PKT(1),P1T0)
      COMMON /RAYVAR/ZDIR,PKK,RTPAA0,ATTEN,SIGMA,X,Y,Z,DAGE,XF,YF,ZF,
     AXT,YT,ZT,XS,YS,ZS,XSS,YSS,ZSS,XSSS,YSSS,ZSSS,P3,P3F,P3T,P3S,
     BXFS,YFS,XTS,YTS,ZFTP3,XFTZ,YFTZ,ZFTZ,ZFA,ZTA,P3FTZ,P3FA,P3TA,AREA,
     CDAGDS
      REAL*8 SIGMA,X,Y,Z,DAGE,XF,YF,ZF,XT,YT,ZT
      REAL XKS(3),XKFS(2),XKTS(2),XKFTZ(3),XKSS(3),XKSSS(3)
      REAL*8 XK(3),XKF(3),XKT(3)
      EQUIVALENCE (XK(1),X),(XKF(1),XF),(XKT(1),XT),(XKS(1),XS)
      EQUIVALENCE (XKFS(1),XFS),(XKTS(1),XTS),(XKFTZ(1),XFTZ)
      EQUIVALENCE (XKSS(1),XSS),(XKSSS(1),XSSS)
      COMMON /ATMSPH/ GAM,C,U,V,DCDZ,DUDZ,DVDZ,D2CDZ2,D2UDZ2,D2VDZ2,RHO
      REAL*8 GAM,C,U,V
      REAL UK*8(2),DUKDZ(2),D2UKDZ(2)
      REAL*8 DDELTA,RTPKK
      EQUIVALENCE (UK(1),U),(DUKDZ(1),DUDZ),(D2UKDZ(1),D2UDZ2)
      CALL AIR(Z)
      DDELTA=DBLE(OMEGA)-U*DBLE(P10)-V*DBLE(P20)
      DELTA=DDELTA
      RTPKK=DDELTA/C
      PKK=RTPKK**2
      IF(RTPKK.LT.RTPAA0) RETURN 2
      CSQOD=C*C/DDELTA
      P3=DSQRT((RTPKK-RTPAA0)*(RTPKK+RTPAA0))
      P3=SIGN(P3,ZDIR)
      ZS=P3*CSQOD
      DO 20 K=1,2
      XKS(K)=CSQOD*PK(K)+UK(K)
   20 CONTINUE
      DELTAF=OMEGAF-U*P1F0-V*P2F0
      DELTAT=OMEGAT-U*P1T0-V*P2T0
      DELTAZ=-(P10*DUDZ+P20*DVDZ)
      DLTAZZ=-(P10*D2UDZ2+P20*D2VDZ2)
```

```
      DLNDLZ=DELTAZ/DELTA
      D2LNDL=DLTAZZ/DELTA-DLNDLZ**2
      DLNCDZ=DCDZ/C
      D2LCDZ=D2CDZ2/C-DLNCDZ**2
      CSQODZ=CSQOD*(2.*DLNCDZ-DLNDLZ)
      CSQDZZ=CSQODZ*(2.*DLNCDZ-DLNDLZ)+CSQOD*(2.*D2LCDZ-D2LNDL)
      P3S=DELTA*(DLNDLZ-DLNCDZ)
      P3SZ=-DELTA*D2LCDZ-DELTAZ*DLNCDZ+DLTAZZ
      ZSS=CSQOD*P3S+CSQODZ*P3*ZS
      P3SS=P3SZ*ZS
      ZSSS=CSQOD*P3SS+2.*CSQODZ*P3S*ZS+
     A P3*(CSQDZZ*ZS*ZS+CSQODZ*ZSS)
      P3TA=-(P1T0*DUDZ+P2T0*DVDZ)-DELTAT*DLNCDZ
      P3FA=-(P1F0*DUDZ+P2F0*DVDZ)-DELTAF*DLNCDZ
      P3FTZ=P3SZ
      ZFTP3=CSQOD
      ZFTZ=P3*CSQODZ
      ZFA=-DELTAF*P3/PKK
      ZTA=-DELTAT*P3/PKK
      DO 40 K=1,2
      XKFTZ(K)=DUKDZ(K)+CSQODZ*PK(K)
      XKFS(K)=ZFTP3*PKF(K)-PK(K)*DELTAF/PKK
      XKTS(K)=ZFTP3*PKT(K)-PK(K)*DELTAT/PKK
      XKSS(K)=ZS*(CSQODZ*PK(K)+DUKDZ(K))
      XKSSS(K)=ZSS*(CSQODZ*PK(K)+DUKDZ(K))+
     A ZS*ZS*(CSQDZZ*PK(K)+D2UKDZ(K))
   40 CONTINUE
      DAGDS=PCONST*.5*(1.+GAM)*((SNGL(RTPKK)/DELTA0)**1.5)/SQRT(RHO)
      IF(ZS.EQ.0.) RETURN 1
      RETURN
      END
C./       ADD   NAME=ADVANS
      SUBROUTINE ADVANS
      COMMON /RYCTRL/NORAYS,STND,UL,UR,LL,LR,PRTRAY,TIMBEG,DELTIM,NTIMS,
     APHIBEG,DELPHI,NPHIS
      LOGICAL*1 NORAYS,STND,UL,UR,LL,LR,PRTRAY
      COMMON /RAYNIT/ KGMH,NDCRVS,NUCRVS,IUPDWN,T0,PHI0,X0,Y0,Z0,
     AP10,P20,P30,OMEGA,DELTA0,P1F0,P2F0,P3F0,OMEGAF,XT0,YT0,ZT0,
     BP1T0,P2T0,P3T0,OMEGAT,XS0,YS0,ZS0,P3S0,RHO0,PCONST,NAGES,AGES(20)
      INTEGER*2 KGMH,NDCRVS,NUCRVS,IUPDWN
      REAL PK0(2),PKF0(2),PKT0(2)
      EQUIVALENCE (PK0(1),P10),(PKF0(1),P1F0),(PKT0(1),P1T0)
      COMMON /ACSPOT/TIME,XR0,YR0,ZR0,XDOT,YDOT,ZDOT,AIRSPD,ASPDOT,
     AC0,U0,V0,CDOT,XMACH,XMADOT,XMU,XMUDOT,COSMU,SINMU,EK(3,3),
     B EKDOT(3,3),GLOAD,HEADIN,CLIMB,BANK
      REAL XKR0(3),XKDOT(3)
      EQUIVALENCE(XKR0(1),XR0),(XKDOT(1),XDOT)
      COMMON /RAYVAR/ZDIR,PKK,RTPAA0,ATTEN,SIGMA,X,Y,Z,DAGE,XF,YF,ZF,
     AXT,YT,ZT,XS,YS,ZS,XSS,YSS,ZSS,XSSS,YSSS,ZSSS,P3,P3F,P3T,P3S,
     BXFS,YFS,XTS,YTS,ZFTP3,XFTZ,YFTZ,ZFTZ,ZFA,ZTA,P3FTZ,P3FA,P3TA,AREA,
     CDAGDS
      REAL*8 SIGMA,X,Y,Z,DAGE,XF,YF,ZF,XT,YT,ZT
      REAL XKS(3),XKFS(2),XKTS(2),XKFTZ(3),XKSS(3),XKSSS(3)
      REAL*8 XK(3),XKF(3),XKT(3)
      EQUIVALENCE (XK(1),X),(XKF(1),XF),(XKT(1),XT),(XKS(1),XS)
      EQUIVALENCE (XKFS(1),XFS),(XKTS(1),XTS),(XKFTZ(1),XFTZ)
      EQUIVALENCE (XKSS(1),XSS),(XKSSS(1),XSSS)
      COMMON /RAYHLD/HSIGMA,HX,HY,HZ,HDAGE,HXF,HYF,HZF,HXT,HYT,HZT,
     AHXS,HYS,HZS,HXSS,HYSS,HZSS,HXSSS,HYSSS,HZSSS,HP3,HP3F,HP3T,HP3S,
     BHXFS,HYFS,HXTS,HYTS,HZFTP3,HXFTZ,HYFTZ,HZFTZ,HZFA,HZTA,HP3FTZ,
     CHP3FA,HP3TA,HAREA,HDAGDS
      REAL*8 HSIGMA,HX,HY,HZ,HDAGE,HXF,HYF,HZF,HXT,HYT,HZT
      REAL HXKS(3),HXKFS(2),HXKTS(2),HXKFTZ(3),HXKSS(3),HXKSSS(3)
      REAL*8 HXK(3),HXKF(3),HXKT(3)
      EQUIVALENCE (HXK(1),HX),(HXKF(1),HXF),(HXKT(1),HXT),(HXKS(1),HXS)
      EQUIVALENCE (HXKFS(1),HXFS),(HXKTS(1),HXTS),(HXKFTZ(1),HXFTZ)
      EQUIVALENCE (HXKSS(1),HXSS),(HXKSSS(1),HXSSS)
      REAL*8 TPRINT,TIMCVR
      REAL*8 RF(3),RT(3),RK(3),SIG
      LOGICAL TONE
      REAL RS(3)
      LOGICAL OPSIGN
      OPSIGN(A,B)=((A.LT.0.).AND.(B.GE.0.)).OR.((A.GT.0.).AND.(B.LT.0.))
      AA=Z-HZ
      BB=.5*(ZS+HZS)
      CC=(ZSS-HZSS)/10.
      DD=(ZSSS+HZSSS)/120.
      IF(AA.EQ.0.) RETURN
      DELSIG=AA/BB
      DO 10 K=1,5
      ENUM=((-DD*DELSIG+CC)*DELSIG-BB)*DELSIG+AA
      DEN=(-3.*DD*DELSIG+2.*CC)*DELSIG-BB
      IF(DEN*AA.GE.0.) GO TO 12
      DELSIG=DELSIG-ENUM/DEN
   10 CONTINUE
      GO TO 15
   12 WRITE(6,14)
   14 FORMAT(' TDLSIG TOO LARGE.')
   15 SIGMA=HSIGMA+DELSIG
      HDLSIG=.5*DELSIG
      DLSIG6=DELSIG/6.
      DO 20 K=1,2
      XK(K)=HXK(K)+(((XKSSS(K)+HXKSSS(K))*DELSIG/12.-(XKSS(K)-HXKSS(K)))
     A*DELSIG*.2+(XKS(K)+HXKS(K)))*HDLSIG
   20 CONTINUE
      EM11=1.-DLSIG6*(2.*ZFTZ+HZFTZ)
      EM12=-DLSIG6*(2.*ZFTP3+HZFTP3)
      EM21=-DLSIG6*(2.*P3FTZ+HP3FTZ)
      EM22=1.
      DET=EM11*EM22-EM12*EM21
      HEM11=1.+DLSIG6*(ZFTZ+2.*HZFTZ)
      HEM12=DLSIG6*(ZFTP3+2.*HZFTP3)
      HEM21=DLSIG6*(P3FTZ+2.*HP3FTZ)
      HEM22=1.
      AZ=HEM11*HZF+HEM12*HP3F+HDLSIG*(ZFA+HZFA)
      BZ=HEM21*HZF+HEM22*HP3F+HDLSIG*(P3FA+HP3FA)
      ZF=(EM22*AZ-EM12*BZ)/DET
      P3F=(-EM21*AZ+EM11*BZ)/DET
      AZ=HEM11*HZT+HEM12*HP3T+HDLSIG*(ZTA+HZTA)
      BZ=HEM21*HZT+HEM22*HP3T+HDLSIG*(P3TA+HP3TA)
      ZT=(EM22*AZ-EM12*BZ)/DET
      P3T=(-EM21*AZ+EM11*BZ)/DET
      DO 40 K=1,2
      XKF(K)=HXKF(K)+HDLSIG*(XKFS(K)+HXKFS(K))+DLSIG6*(ZF*(2.*XKFTZ(K)+
     A HXKFTZ(K))+HZF*(XKFTZ(K)+2.*HXKFTZ(K)))
      XKT(K)=HXKT(K)+HDLSIG*(XKTS(K)+HXKTS(K))+DLSIG6*(ZT*(2.*XKFTZ(K)+
     A HXKFTZ(K))+HZT*(XKFTZ(K)+2.*HXKFTZ(K)))
   40 CONTINUE
      AREA=ARTUBE(P3,XKF,XKT)
C     PFACT=PCONST*C*SQRT(RHO*RTPKK/(DELTA0*(ABS(AREA)+1.E-12)))
      ARFCT=SQRT(ABS(AREA)+1.E-12)
      HARFCT=SQRT(ABS(HAREA)+1.E-12)
      IF(OPSIGN(HAREA,AREA)) GO TO 70
      DAGE=HDAGE+ATTEN*DELSIG*(DAGDS*(2.*HARFCT+ARFCT)+HDAGDS*
     A (HARFCT+2.*ARFCT))/(1.5*(ARFCT+HARFCT)**2)
      RETURN
   70 AR1=HAREA
      AR2=AREA
      TONE=AR2.GT.AR1
      TAU1=0.
      TAU2=1.
  100 TAU=.5*(TAU1+TAU2)
      TAUPR=1.-TAU
      DO 110 K=1,3
      RF(K)=HXKF(K)*TAUPR+XKF(K)*TAU
      RT(K)=HXKT(K)*TAUPR+XKT(K)*TAU
  110 CONTINUE
      PZ=TAUPR*HP3+TAU*P3
      IF(TAU2-TAU1.LT.1.E-6) GO TO 160
      ARM=ARTUBE(PZ,RF,RT)
      IF(ARM) 120,160,140
  120 IF(TONE) GO TO 150
  130 TAU2=TAU
      GO TO 100
  140 IF(TONE) GO TO 130
  150 TAU1=TAU
      GO TO 100
  160 SIG=TAUPR*HSIGMA+TAU*SIGMA
      DAGE=HDAGE
      IF(HAREA.NE.0.) DAGE=HDAGE+ATTEN*DLSIG6*8.*(HDAGDS*(1.5-TAU)+
     A DAGDS*TAU)*TAU/HARFCT
      DO 170 K=1,3
      RK(K)=TAU*XK(K)+TAUPR*HXK(K)
      RS(K)=TAU*XKS(K)+TAUPR*HXKS(K)
  170 CONTINUE
      IF(.NOT.PRTRAY) GO TO 200
      WRITE(6,180)
  180 FORMAT(' CAUSTIC POINT CROSSED.')
      TPRINT=TIMCVR(SIGMA,2)
      ELEV=PHELEV(0.)
      WRITE(6,190) TPRINT,X,Y,Z,P3,ELEV,ZS,DAGE
  190 FORMAT(1X,F10.1,3F10.0,G10.3,F6.1,10X,F10.1,2X,'0.',6X,G10.4)
  200 CALL RCSPCL('CAUSTIC ',SIG,RK,PZ,RF,RT,RS,0.)
      AGES(NAGES)=DAGE
      NAGES=NAGES+1
      DAGE=0.D0
      IF(AREA.NE.0.) DAGE=ATTEN*DLSIG6*8.*(DAGDS*(1.5-TAUPR)+
     A HDAGDS*TAUPR)*TAUPR/ARFCT
      RETURN
      END
C./       ADD   NAME=RCRVIT
      SUBROUTINE RCRVIT
      COMMON /RAYNIT/ KGMH,NDCRVS,NUCRVS,IUPDWN,T0,PHI0,X0,Y0,Z0,
     AP10,P20,P30,OMEGA,DELTA0,P1F0,P2F0,P3F0,OMEGAF,XT0,YT0,ZT0,
     BP1T0,P2T0,P3T0,OMEGAT,XS0,YS0,ZS0,P3S0,RHO0,PCONST,NAGES,AGES(20)
      INTEGER*2 KGMH,NDCRVS,NUCRVS,IUPDWN
      COMMON /RAYVAR/ZDIR,PKK,RTPAA0,ATTEN,SIGMA,X,Y,Z,DAGE,XF,YF,ZF,
     AXT,YT,ZT,XS,YS,ZS,XSS,YSS,ZSS,XSSS,YSSS,ZSSS,P3,P3F,P3T,P3S,
     BXFS,YFS,XTS,YTS,ZFTP3,XFTZ,YFTZ,ZFTZ,ZFA,ZTA,P3FTZ,P3FA,P3TA,AREA,
     CDAGDS
      REAL*8 SIGMA,X,Y,Z,DAGE,XF,YF,ZF,XT,YT,ZT
      COMMON /ATMSPH/ GAM,C,U,V,DCDZ,DUDZ,DVDZ,D2CDZ2,D2UDZ2,D2VDZ2,RHO
      REAL*8 GAM,C,U,V
      COMMON /RAYHLD/HSIGMA,HX,HY,HZ,HDAGE,HXF,HYF,HZF,HXT,HYT,HZT,
     AHXS,HYS,HZS,HXSS,HYSS,HZSS,HXSSS,HYSSS,HZSSS,HP3,HP3F,HP3T,HP3S,
     BHXFS,HYFS,HXTS,HYTS,HZFTP3,HXFTZ,HYFTZ,HZFTZ,HZFA,HZTA,HP3FTZ,
     CHP3FA,HP3TA,HAREA,HDAGDS
      REAL*8 HSIGMA,HX,HY,HZ,HDAGE,HXF,HYF,HZF,HXT,HYT,HZT
      REAL*8 ZA,ZB,ZMID
      REAL*8 DDELTA,RTPKK
      ZA=HZ
      ZB=Z
    5 ZMID=.5D0*(ZB+ZA)
      IF(DMIN1(DABS(ZMID-ZA),DABS(ZB-ZMID)).LT.1.D-4) GO TO 100
      CALL AIR(ZMID)
      DDELTA=DBLE(OMEGA)-U*DBLE(P10)-V*DBLE(P20)
      RTPKK=DDELTA/C
      IF(RTPKK-RTPAA0) 10,90,20
   10 ZB=ZMID
      GO TO 5
   20 ZA=ZMID
      GO TO 5
   90 Z=ZMID
      RETURN
  100 Z=ZA
      RETURN
      END
C./       ADD   NAME=RECORD
      SUBROUTINE RECORD(*)
      COMMON /ATMCON/ REARTH,G0,RSTAR,R0M0,R0G0M0
```

```
      COMMON /ACIDNT/ IDENT,ACWT
      REAL*8 IDENT
      COMMON /CLASES/CNAMES(30),NRCURV(2,2),TYPRAY(3,2,2),DIRECT,LOFT,
     A UP,DOWN
      REAL*8 CNAMES
      LOGICAL *1 TYPRAY,DIRECT,LOFT,UP,DOWN
      COMMON /GROUND/ ZGRND,CGRND,UGRND,VGRND,REFLFC
      COMMON /ACSPOT/TIME,XR0,YR0,ZR0,XDOT,YDOT,ZDOT,AIRSPD,ASPDOT,
     AC0,U0,V0,CDOT,XMACH,XMADOT,XMU,XMUDOT,COSMU,SINMU,EK(3,3),
     B EKDOT(3,3),GLOAD,HEADIN,CLIMB,BANK
      COMMON /ATMSPH/ GAM,C,U,V,DCDZ,DUDZ,DVDZ,D2CDZ2,D2UDZ2,D2VDZ2,RHO
      REAL*8 GAM,C,U,V
      COMMON /RAYNIT/ KGMH,NDCRVS,NUCRVS,IUPDWN,T0,PHI0,X0,Y0,Z0,
     AP10,P20,P30,OMEGA,DELTA0,P1F0,P2F0,P3F0,OMEGAF,XT0,YT0,ZT0,
     BP1T0,P2T0,P3T0,OMEGAT,XS0,YS0,ZS0,P3S0,RHO0,PCONST,NAGES,AGES(20)
      INTEGER*2 KGMH,NDCRVS,NUCRVS,IUPDWN
      REAL PK(2),PKF(2),PKT(2)
      EQUIVALENCE (PK(1),P10),(PKF(1),P1F0),(PKT(1),P1T0)
      COMMON /RAYVAR/ZDIR,PKK,RTPAA0,ATTEN,SIGMA,X,Y,Z,DAGE,XF,YF,ZF,
     AXT,YT,ZT,XS,YS,ZS,XSS,YSS,ZSS,XSSS,YSSS,ZSSS,P3,P3F,P3T,P3S,
     BXFS,YFS,XTS,YTS,ZFTP3,XFTZ,YFTZ,ZFTZ,ZFA,ZTA,P3FTZ,P3FA,P3TA,AREA,
     CDAGDS
      REAL*8 SIGMA,X,Y,Z,DAGE,XF,YF,ZF,XT,YT,ZT
      REAL XKS(3),XKFS(2),XKTS(2),XKFTZ(3),XKSS(3),XKSSS(3)
      REAL*8 XK(3),XKF(3),XKT(3)
      EQUIVALENCE (XK(1),X),(XKF(1),XF),(XKT(1),XT),(XKS(1),XS)
      EQUIVALENCE (XKFS(1),XFS),(XKTS(1),XTS),(XKFTZ(1),XFTZ)
      EQUIVALENCE (XKSS(1),XSS),(XKSSS(1),XSSS)
      REAL RX(3),RXF(3),RXT(3)
      DATA DGPRAD/57.295780/
      NCLAS=3
      IF(KGMH.EQ.1) GO TO 10
      NCLAS=2*(NDCRVS+3*(3-KGMH+2*(2-IUPDWN)))+3
      IF(.NOT.TYPRAY(NDCRVS,3-IUPDWN,KGMH-1)) GO TO 20
      GO TO 15
   10 IF(.NOT.DIRECT) GO TO 20
   15 RTPKK=SQRT(PKK)
      PFACT=PCONST*C*SQRT(RHO*RTPKK/(DELTA0*(ABS(AREA)+1.E-12)))
      PFACT=PFACT*ATTEN*(1.+REFLFC)
      VLIFT=ACWT*GLOAD*G0*COSMU*COS((PHI0-BANK)/DGPRAD)/
     A (RHO0*SINMU*AIRSPD**2)
      RECPHI=AMOD(AMOD(PHI0,360.)+450.,360.)-90.
      RSIGM=SNGL(SIGMA)
      DO 17 K=1,3
      RX(K)=XK(K)
      RXF(K)=XKF(K)
      RXT(K)=XKT(K)
   17 CONTINUE
      AGES(NAGES)=DAGE
      WRITE(9) CNAMES(NCLAS),KGMH,NDCRVS,IUPDWN,XMACH,VLIFT,T0,RECPHI,
     ARSIGM,RX,OMEGA,PK,P3,XKS,RXT,RXF,PFACT,NAGES,(AGES(K),K=1,NAGES)
   20 IF(KGMH.EQ.1) GO TO 30
      IF(NDCRVS.GE.NRCURV(3-IUPDWN,KGMH-1)) RETURN
      RETURN 1
   30 IF(LOFT) RETURN 1
      RETURN
      END
C./       ADD   NAME=ARTUBE
      FUNCTION ARTUBE(PZ,RF,RT)
      COMMON /RAYNIT/ KGMH,NDCRVS,NUCRVS,IUPDWN,T0,PHI0,X0,Y0,Z0,
     AP10,P20,P30,OMEGA,DELTA0,P1F0,P2F0,P3F0,OMEGAF,XT0,YT0,ZT0,
     BP1T0,P2T0,P3T0,OMEGAT,XS0,YS0,ZS0,P3S0,RHO0,PCONST,NAGES,AGES(20)
      INTEGER*2 KGMH,NDCRVS,NUCRVS,IUPDWN
      REAL PK(2),PKF(2),PKT(2)
      EQUIVALENCE (PK(1),P10),(PKF(1),P1F0),(PKT(1),P1T0)
      INTEGER INDET(3)/2,3,1/
      REAL*8 RF(3),RT(3)
      ARTUBE=PZ*(RF(1)*RT(2)-RF(2)*RT(1))
      PKK=PZ**2
      DO 10 K=1,2
      ARTUBE=ARTUBE+PK(K)*(RF(INDET(K))*RT(INDET(K+1))-RF(INDET(K+1))*
     A RT(INDET(K)))
      PKK=PKK+PK(K)**2
   10 CONTINUE
      ARTUBE=ARTUBE/SQRT(PKK)
      RETURN
      END
C./       ADD   NAME=RCSPCL
      SUBROUTINE RCSPCL(TYPE,SIG,RK,PZ,RF,RT,RS,AREA)
      LOGICAL*1 TYPE(8)
      REAL*8 RK(3),RF(3),RT(3),SIG
      REAL RS(3),AREA,RF4(3),RT4(3),RK4(3)
      COMMON /CLASES/CNAMES(30),NRCURV(2,2),TYPRAY(3,2,2),DIRECT,LOFT,
     A UP,DOWN
      REAL*8 CNAMES
      LOGICAL *1 TYPRAY,DIRECT,LOFT,UP,DOWN
      COMMON /RAYNIT/ KGMH,NDCRVS,NUCRVS,IUPDWN,T0,PHI0,X0,Y0,Z0,
     AP10,P20,P30,OMEGA,DELTA0,P1F0,P2F0,P3F0,OMEGAF,XT0,YT0,ZT0,
     BP1T0,P2T0,P3T0,OMEGAT,XS0,YS0,ZS0,P3S0,RHO0,PCONST,NAGES,AGES(20)
      INTEGER*2 KGMH,NDCRVS,NUCRVS,IUPDWN
      REAL PK(2),PKF(2),PKT(2)
      EQUIVALENCE (PK(1),P10),(PKF(1),P1F0),(PKT(1),P1T0)
      NCLAS=3
      IF(KGMH.EQ.1) GO TO 10
      NCLAS=2*(NDCRVS+3*(3-KGMH+2*(2-IUPDWN)))+3
   10 RECPHI=AMOD(AMOD(PHI0,360.)+450.,360.)-90.
      RSIGM=SNGL(SIG)
      DO 20 K=1,3
      RK4(K)=RK(K)
      RF4(K)=RF(K)
      RT4(K)=RT(K)
   20 CONTINUE
      WRITE(11)TYPE,CNAMES(NCLAS),NDCRVS,NUCRVS,T0,RECPHI,RSIGM,RK4,RF4,
     ART4,AREA,PK,PZ
      RETURN
      END
C./       ADD   NAME=RDSPCL
C  *****************************************************************
C  *** SIGNATURE CALCULATIONS - RDSPCL,SIGNUR,FREAD,AGING,HILBRT     ***
C  ***                         SIGPRT,CPVAL,SORTEM                  ***
C  ***                         (DREAD,FFA2F,FFA2I)                  ***
C  *****************************************************************
      SUBROUTINE RDSPCL
      REAL*8 PTTYPE,RYCLAS,SIGPRN,TPRN,TIMCVR
      REAL RK(3),XF(3),XT(3),PK(3)
      INTEGER*2 NHIGH,NLOW
      COMMON /PRINTS/ TITLE(30),KTPSIG,CVRTIM,TIMLBL
      REAL*8 TIMLBL
      LOGICAL CVRTIM
      REWIND 11
      WRITE(6,5) TITLE
    5 FORMAT('1',30A4)
      WRITE(6,6) TIMLBL,TIMLBL
    6 FORMAT('0 POINT',T11,'#HIGH #LOW',T22,'RAY',T34,'TIME',T44,'PHI',
     AT54,'TIME',T66,'X',T76,'Y',T86,'Z',T93,'RAY NORMAL',T109,'AREA'/
     BT3,'TYPE',T21,'CLASS',T32,'(INITIAL)',T42,'(INITIAL)',T93,'AZIMUTH
     C ELEV'/T33,A8,T53,A8,T65,'MET',T75,'MET',T85,'MET',T93,'DEG',T100,
     D'DEG',T106,'MET**2/SEC')
   10 READ(11,END=100)PTTYPE,RYCLAS,NHIGH,NLOW,TIM0,PHI0,SIGMA,RK,XF,XT,
     AAREA,PK
      TPRN=TIMCVR(DBLE(TIM0),2)
      SIGPRN=TIMCVR(DBLE(SIGMA),2)
      CALL EAMENU(ELEV,AZIM,PMAG,PK(1),PK(2),PK(3))
      WRITE(6,20)PTTYPE,NHIGH,NLOW,RYCLAS,TPRN,PHI0,SIGPRN,RK,AZIM,ELEV,
     A AREA
   20 FORMAT(1X,A8,2I5,T22,A8,T31,F10.1,F8.2,T50,F10.1,2F10.0,F10.1,
     AF7.0,F7.1,G12.4)
      GO TO 10
  100 RETURN
      END
C./       ADD   NAME=SIGNUR
      SUBROUTINE SIGNUR
      COMMON /FFTAB/ ACIDNT,KRCAC,NSPDS,SPEEDS(11),LOCSPD(10),KTABL,
     A NTAU,TAU(200),FAC(200),FLC(200)
      REAL*8 ACIDNT
      COMMON /BASEAG/ NTERMS,XILEAD(2),XI(500),XITAIL(502),
     A                VLEAD(2),V(500),VTAIL(502)
      DIMENSION XII(1004),VI(1004)
      EQUIVALENCE (XII(1),XILEAD(1)),(VI(1),VLEAD(1))
      COMMON /PRINTS/ TITLE(30),KTPSIG,CVRTIM,TIMLBL
      REAL*8 TIMLBL,TIMCVR
      LOGICAL CVRTIM
      REAL*8 SIGD,TD0
      COMMON /SIGPAR/ RAYNAM,KGMH,NRCURV,IUPDWN,IDENT,XMACH,VLIFT,T0,
     APHI0,SIGMA,XK(3),OMEGA,PK(3),XKS(3),XKT(3),XKF(3),PFACT,NAGES,
     B AGES(20)
      REAL*8 IDENT,RAYNAM
      INTEGER*2 KGMH,NRCURV,IUPDWN
      IF(KTPSIG.LE.0) RETURN
      REWIND 9
      READ(9) IDENT
      WRITE(6,15) TITLE
      IF(KTPSIG.GE.2)CALL FREAD
      IF(KTPSIG.GT.1) GO TO 10
      WRITE(6,16) IDENT
      WRITE(6,20) TIMLBL,TIMLBL
   10 CONTINUE
      READ(9,END=500)RAYNAM,KGMH,NRCURV,IUPDWN,XMACH,VLIFT,T0,PHI0,SIGMA
     A,XK,OMEGA,PK,XKS,XKT,XKF,PFACT,NAGES,(AGES(K),K=1,NAGES)
      CALL EAMENU(ELEV,AZIM,PMAG,PK(1),PK(2),PK(3))
      IF(KTPSIG-2) 25,17,11
   11 WRITE(6,15) TITLE
   15 FORMAT('1',30A4)
      WRITE(6,16) IDENT
   16 FORMAT('0A/C IDENT=',A8)
   17 WRITE(6,20) TIMLBL,TIMLBL
   20 FORMAT(' RAY TYPE MACH#',T20,'TINIT',T28,'PHI0',T37,'TIME',T50,'X'
     A,T60,'Y',T67,'Z',T72,'RAY NORMAL',T84,'TFACT',T91,'PFACT',T104,
     B'VLIFT'/T20,A8,T28,'DEG.',T36,A8,T49,'MET',T59,'MET',T66,'MET',
     CT71,'AZIMUTH ELEV',T83,'MS/MET',T90,'PA/MET**.5',T104,'MET**2')
   25 TFACT=1000./OMEGA
      TD0=TIMCVR(DBLE(T0),2)
      SIGD=TIMCVR(DBLE(SIGMA),2)
      WRITE(6,100)RAYNAM,XMACH,TD0,PHI0,SIGD,XK,AZIM,ELEV,TFACT,
     A PFACT,VLIFT,(AGES(K),K=1,NAGES)
  100 FORMAT('0',A8,1X,F5.3,F10.1,F7.2,F10.1,2F10.0,F6.1,F7.0,F7.1,F6.3,
     A 2G11.4,(/T19,'AGES(M**.5)=',9F9.2))
      IF(KTPSIG.EQ.1) GO TO 10
      CALL NEWTAB
      DO 200 K=1,NTAU
      XI(K)=TAU(K)
      V(K)=FAC(K)+VLIFT*FLC(K)
  200 CONTINUE
      NTERMS=NTAU
      CALL AGING(AGES(1))
      IF(NAGES.LE.1) GO TO 215
      DO 210 K=2,NAGES
      CALL HILBRT
      CALL AGING(AGES(K))
  210 CONTINUE
  215 CALL SIGPRT
      GO TO 10
  500 RETURN
      END
C./       ADD   NAME=FREAD
      SUBROUTINE FREAD
      COMMON /SIGPAR/ RAYNAM,KGMH,NRCURV,IUPDWN,IDENT,XMACH,VLIFT,T0,
     APHI0,SIGMA,XK(3),OMEGA,PK(3),XKS(3),XKT(3),XKF(3),PFACT,NAGES,
     B AGES(20)
      REAL*8 IDENT,RAYNAM
```

```
      INTEGER*2 KGMH,NRCURV,IUPDWN
      COMMON /FFTAB/ ACIDNT,KRCAC,NSPDS,SPEEDS(11),LOCSPD(10),KTABL,
     A NTAU,TAU(200),FAC(200),FLC(200)
      REAL*8 ACIDNT
      REAL*8 BUFFER(10)
      COMMON /FERMSG/MESG(26)
      CALL LJUST(8,1,IDENT,ACIDNT)
      KRCAC=1
   10 CALL DREAD(90,KRCAC,BUFFER,&900)
      IF(ACIDNT.EQ.BUFFER(1)) GO TO 20
      CALL FFA2N(BUFFER,18,5,1,DUMMY,0.,KERR)
      KINCR=DUMMY+.5
      IF(KINCR.EQ.0) GO TO 950
      KRCAC=KRQaA+KINCR
      GO TO 10
   20 CALL FFA2N(BUFFER,26,2,1,DUMMY,1.,KERR)
      NSPDS=DUMMY+.5
      NCARDS=(NSPDS+3)/4
      DO  40 I=1,NCARDS
      K1=1
      K2=MIN0(4,NSPDS-4*(I-1))
      DO  30 K=K1,K2
      KK=K+4*(I-1)
      CALL FFA2N(BUFFER,18+10*K,5,1,SPEEDS(KK),0.,KERR)
      CALL FFA2N(BUFFER,23+10*K,5,1,DUMMY,0.,KERR)
      LOCSPD(KK)=DUMMY+.5
      LOCSPD(KK)=LOCSPD(KK)+KRCAC
   30 CONTINUE
      CALL DREAD(90,KRCAC+I,BUFFER,&900)
   40 CONTINUE
      WRITE(6,50) IDENT,(SPEEDS(K),K=1,NSPDS)
   50 FORMAT('0F-FUNCTION TABLES FOR ',A8,' AIRCRAFT.'/'  TABLES FOR MAC
     AH NUMBERS',20F5.2)
      SPEEDS(NSPDS+1)=SPEEDS(NSPDS)
      DO  60 M=1,NSPDS
      SPEEDS(NSPDS-M+2)=.5*(SPEEDS(NSPDS-M+2)+SPEEDS(NSPDS-M+1))
   60 CONTINUE
      SPEEDS(1)=1.
      LTABL=1
      GO TO 150
      ENTRY NEWTAB
      IF(NSPDS.EQ.1)  RETURN
      DO 100 K=1,NSPDS
      IF(AMIN1(XMACH-SPEEDS(K),SPEEDS(K+1)-XMACH).GE.0.) GO TO 120
  100 CONTINUE
      IF(XMACH.GT.SPEEDS(NSPDS+1)) WRITE(6,110) XMACH,SPEEDS(NSPDS+1)
  110 FORMAT(' MACH NUMBER ',F5.2,' IS GREATER THAN MAXIMUM IN TABLES '
     A,F5.2,'. SUGGEST EXTENDING TABLES.')
      K=NSPDS
  120 LTABL=K
      IF(LTABL.EQ.KTABL) RETURN
  150 KTABL=LTABL
      MREC=LOCSPD(KTABL)
      CALL DREAD(90,MREC,BUFFER,&900)
      CALL FFA2N(BUFFER,16,6,1,XLAC,0.,KERR)
      CALL FFA2N(BUFFER,22,7,1,STEP,0.,KERR)
      CALL FFA2N(BUFFER,13,3,1,DUMMY,0.,KERR)
      NTAU=DUMMY+.5
      XLR=SQRT(XLAC)
      CONST=1./(XLR*XLAC)
      DO 200 K=1,NTAU
      CALL FFA2N(BUFFER,48,3,1,EXP10,0.,KERR)
      CALL FFA2N(BUFFER,35,12,1,FAC(K),0.,KERR)
      FAC(K)=FAC(K)*XLR*(10.**EXP10)
      CALL FFA2N(BUFFER,67,3,1,EXP10,0.,KERR)
      CALL FFA2N(BUFFER,54,12,1,FLC(K),0.,KERR)
      FLC(K)=FLC(K)*CONST*(10.**EXP10)
      TAU(K)=(K-1)*STEP*XLAC
      CALL DREAD(90,MREC+K,BUFFER,&900)
  200 CONTINUE
      RETURN
  900 WRITE(6,910) MESG
  910 FORMAT(' DA/IO ERROR ON UNIT 90.'/1X,Z8,I6,20A4,4Z9)
      STOP 900
  950 WRITE(6,960) IDENT
  960 FORMAT(' AIRCRAFT ID ',A8,' NOT FOUND. PROGRAM TERMINATED.')
      STOP 960
      END
C./       ADD   NAME=AGING
      SUBROUTINE AGING(AGE)
      COMMON /BASEAG/ NTERMS,XILEAD(2),XI(500),XITAIL(502),
     A                          VLEAD(2),V(500),VTAIL(502)
      DIMENSION XII(1004),VI(1004)
      EQUIVALENCE (XII(1),XILEAD(1)),(VI(1),VLEAD(1))
      REAL*8 SA,SB,SC,SD,SE1,SE2
      LOGICAL JUMP
      DO 2 K=1,2
      XII(K)=XI(1)
      XII(NTERMS+K+2)=XI(NTERMS)
      VI(K)=0.
      VI(NTERMS+K+2)=0.
    2 CONTINUE
      LTERMS=2
      K=2
      XIB=XII(2)
      VB=0.
      SB=0.D0
      L=2
      VD=0.
      SD=0.D0
      XID=XII(2)
      JUMP=.FALSE.
    5 K=K+1
      IF(K.GT.NTERMS+4) GO TO 200
      XIA=XIB
      VA=VB
      SA=SB
      VB=VI(K)
      XIB=XII(K)-AGE*VB
      SB=SA+(.5D0*(XIB-XIA))*(VB+VA)
      XII(1)=AMIN1(XII(1),XIB)
      XII(NTERMS+4)=AMAX1(XII(NTERMS+4),XIB)
      IF(JUMP) GO TO 15
      IF(XIB.LT.XIA) GO TO 10
      LTERMS=LTERMS+1
      XII(LTERMS)=XIB
      VI(LTERMS)=VB
      GO TO 5
   10 JUMP=.TRUE.
      GO TO 5
   15 IF(XIB.LE.XIA) GO TO 5
   17 XIC=XII(L-1)
      VC=VI(L-1)
      SC=SD-(.5D0*(VC+VD))*(XID-XIC)
      IF(XIC.LE.XIA) GO TO 21
      L=L-1
      VD=VC
      XID=XIC
      SD=SC
      GO TO 17
   20 L=L+1
      XIC=XID
      VC=VD
      SC=SD
      XID=XII(L)
      VD=VI(L)
      SD=SC+(.5D0*(VC+VD))*(XID-XIC)
   21 IF(XIB.LE.XIC) GO TO 5
      IF(XID.LE.XIC) GO TO 20
      IF(XIA.GT.XID) GO TO 20
      XIE=AMIN1(XIB,XID)
      VE1=(VB*(XIE-XIA)+VA*(XIB-XIE))/(XIB-XIA)
      VE2=(VD*(XIE-XIC)+VC*(XID-XIE))/(XID-XIC)
      SE1=SA+(.5D0*(VE1+VA))*(XIE-XIA)
      SE2=SC+(.5D0*(VE2+VC))*(XIE-XIC)
      C=SE1-SE2
      IF(C) 25,40,30
   25 IF(XID-XIB) 20,5,5
   30 A=(VB-VA)/(XIB-XIA)-(VD-VC)/(XID-XIC)
      B=VE1-VE2
      XIE=XIE-2.*C/(B+SQRT(B**2-2.*A*C))
      VE1=(VB*(XIE-XIA)+VA*(XIB-XIE))/(XIB-XIA)
      VE2=(VD*(XIE-XIC)+VC*(XID-XIE))/(XID-XIC)
      SE2=SC+(.5D0*(VC+VE2))*(XIE-XIC)
   40 SB=SE2+(.5D0*(VE1+VB))*(XIB-XIE)
      XII(L)=XIE
      VI(L)=VE2
      XII(L+1)=XIE
      VI(L+1)=VE1
      XII(L+2)=XIB
      VI(L+2)=VB
      L=L+2
      SD=SB
      XID=XIB
      VD=VB
      LTERMS=L
      JUMP=.FALSE.
      GO TO 5
  200 LL=1
      DO 220 L=3,LTERMS
      IF(XII(LL).EQ.XII(L)) GO TO 220
      IF(XII(LL).LT.XII(L-1)) GO TO 210
      IF(VI(LL).EQ.VI(L-1)) GO TO 220
  210 XII(LL+1)=XII(L-1)
      VI(LL+1)=VI(L-1)
      LL=LL+1
  220 CONTINUE
      XII(LL+1)=XII(LTERMS)
      VI(LL+1)=VI(LTERMS)
      NTERMS=LL-3
      RETURN
      END
C./       ADD   NAME=HILBRT
      SUBROUTINE HILBRT
      COMMON /BASEAG/ NTERMS,XILEAD(2),XI(500),XITAIL(502),
     A                          VLEAD(2),V(500),VTAIL(502)
      DIMENSION XII(1004),VI(1004)
      EQUIVALENCE (XII(1),XILEAD(1)),(VI(1),VLEAD(1))
      COMMON /XISAVE/ NSTRMS,XIS(502),VS(502)
      WEIGHT=0.
      XIMEAN=0.
      XIVAR=0.
      XIS(1)=XII(2)
      VS(1)=0.
      NSTRMS=NTERMS+2
      DO 10 K=2,NSTRMS
      XIS(K)=XI(K-1)
      VS(K)=V(K-1)
      AWAIT1=VS(K-1)**2
      AWAIT2=VS(K-1)*VS(K)
      AWAIT3=VS(K)**2
      DELXI=XIS(K)-XIS(K-1)
      WAV=DELXI*(AWAIT1+AWAIT2+AWAIT3)/3.
      WAVX=XIS(K-1)*WAV+((AWAIT1+2.*AWAIT2+3.*AWAIT3)*DELXI**2)/12.
      WAVX2=(WAV*XIS(K-1)+2.*WAVX)*XIS(K-1)+(AWAIT1+3.*AWAIT2+6.*AWAIT3)
     A*DELXI*DELXI*DELXI/30.
      WEIGHT=WEIGHT+WAV
      XIMEAN=XIMEAN+WAVX
      XIVAR=XIVAR+WAVX2
   10 CONTINUE
```

```
      XIMEAN=XIMEAN/WEIGHT
      XIVAR=XIVAR/WEIGHT-XIMEAN**2
      XILNG=SQRT(XIVAR)
      LTRMHF=40
      LTERMS=LTRMHF*2+1
      NTERMS=0
      DO 100 L=1,LTERMS
      XINEW=XILNG*(LTERMS*(L-LTRMHF)/(L*(LTERMS+1.-L)))+XIMEAN
      CALL CPVAL(XINEW,VV,&100)
      NTERMS=NTERMS+1
      V(NTERMS)=VV
      XI(NTERMS)=XINEW
  100 CONTINUE
      XI(NTERMS+1)=2.*(XI(1)-XIMEAN)+XIMEAN
      XI(NTERMS+2)=2.*(XI(NTERMS)-XIMEAN)+XIMEAN
      V(NTERMS5i&]a9j&x      V(NTERMS+2)=0.
      NTERMS=NTERMS+2
      DO 200 K=2,NSTRMS
      IF(XIS(K).GT.XIS(K-1)) GO TO 200
      IF(VS(K).EQ.VS(K-1)) GO TO 200
      DELXI=XILNG
      DO 190 M=1,10
      DELXI=DELXI*.3
      CALL CPVAL(XIS(K)-DELXI,VV,&180)
      NTERMS=NTERMS+1
      XI(NTERMS)=XIS(K)-DELXI
      V(NTERMS)=VV
  180 CALL CPVAL(XIS(K)+DELXI,VV,&190)
      NTERMS=NTERMS+1
      XI(NTERMS)=XIS(K)+DELXI
      V(NTERMS)=VV
  190 CONTINUE
  200 CONTINUE
      CALL SORTEM
      RETURN
      END
C./       ADD   NAME=SIGPRT
      SUBROUTINE SIGPRT
      COMMON /PRINTS/ TITLE(30),KTPSIG,CVRTIM,TIMLBL
      REAL*8 TIMLBL
      LOGICAL CVRTIM
      COMMON /BASEAG/ NTERMS,XILEAD(2),XI(500),XITAIL(502),
     A                      VLEAD(2),V(500),VTAIL(502)
      DIMENSION XII(1004),VI(1004)
      EQUIVALENCE (XII(1),XILEAD(1)),(VI(1),VLEAD(1))
      COMMON /SIGPAR/ RAYNAM,KGMH,NRCURV,IUPDWN,IDENT,XMACH,VLIFT,T0,
     APHI0,SIGMA,XK(3),OMEGA,PK(3),XKS(3),XKT(3),XKF(3),PFACT,NAGES,
     B AGES(20)
      REAL*8 IDENT,RAYNAM
      INTEGER*2 KGMH,NRCURV,IUPDWN
      DATA DGPRAD/57.295780/
      IF(KTPSIG.LE.1) RETURN
      TFACT=1000./OMEGA
      PMAX=0.
      PMIN=0.
      DO 220 K=1,NTERMS
      V(K)=V(K)*PFACT
      PMAX=AMAX1(PMAX,V(K))
      PMIN=AMIN1(PMIN,V(K))
  220 CONTINUE
      PSIG=.05*(PMAX-PMIN)
      KMAX=1
      KMIN=NTERMS
      DO 225 K=1,NTERMS
      IF(ABS(V(K)).LT.PSIG) GO TO 225
      KMIN=MIN0(KMIN,K)
      KMAX=MAX0(KMAX,K+2)
  225 CONTINUE
      DIR=DGPRAD*ATAN2(PK(1),PK(2))+180.
      PN=SQRT(PK(1)**2+PK(2)**2)
      WRITE(6,230) DIR
  230 FORMAT(' ********** SHOCK WAVE ANALYSIS *************'/' LENGTHS L
     AAID OUT IN DIRECTION -',F5.0,' DEGREES.'/T8,'TIME',T16,'LENGTH',
     BT29,'P1',T39,'P2'/T9,'MS',T17,'MET.',T25,'PASCALS',T35,'PASCALS')
      NSHOCK=0
      NN=NTERMS+1
      DO 250 K=1,NN
      IF(XII(K+1).LT.XII(K+2)) GO TO 250
      IF(VI(K+1).GE.VI(K+2)) GO TO 250
      NSHOCK=NSHOCK+1
      AXI=XII(K+1)
      TPR=AXI*TFACT
      XPR=AXI/PN
      PONE=VI(K+1)
      PTWO=VI(K+2)
      WRITE(6,235) TPR,XPR,PONE,PTWO
  235 FORMAT(1X,4F10.2)
      KMIN=MIN0(KMIN,K)
      KMAX=MAX0(KMAX,K+1)
  250 CONTINUE
      WRITE(6,260) NSHOCK,PMIN,PMAX
  260 FORMAT(T5,I5,' SHOCKS FOUND. PMIN=',F8.2,' PASCALS, PMAX=',F8.2,
     A' PASCALS')
      IF(KTPSIG.LT.3) GO TO 500
      WRITE(6,300)
  300 FORMAT('0TOTAL SIGNATURE'/T8,'TIME',T16,'LENGTH',T29,'P'/T9,'MS',
     AT17,'MET.',T25,'PASCALS')
      DO 320 K=KMIN,KMAX
      TPR=XII(K+1)*TFACT
      XPR=XII(K+1)/PN
      WRITE(6,330) TPR,XPR,VI(K+1)
  320 CONTINUE
  330 FORMAT(1X,3F10.2)
  500 RETURN
      END
C./       ADD   NAME=CPVAL
      SUBROUTINE CPVAL(XIARG,V,*)
      COMMON /XISAVE/ NSTRMS,XIS(502),VS(502)
      REAL*8 SUM,ALPHA,RATIO,VSA,VSB,DIF1,DIF2,DIF3,PI,DIFA,DIFB
      DATA PI/3.14159265358979D+0/
      SUM=0.D0
      DO 50 K=2,NSTRMS
      DIFA=DBLE(XIS(K))-XIARG
      DIFB=DBLE(XIS(K-1))-XIARG
      DIF2=DIFA
      DIF1=DIFB
      IF(DABS(DIF2).GE.DABS(DIF1)) GO TO 5
      DIF3=DIF2
      DIF2=DIF1
      DIF1=DIF3
    5 IF(DIF1.NE.0.) GO TO 15
      IF(DIF2.NE.0.) GO TO 7
      IF(VS(K).EQ.VS(K-1)) GO TO 50
      RETURN 1
    7 ALPHA=-DLOG(DABS(DIF2))/DIF2
      GO TO 30
   15 RATIO=(DIF2-DIF1)/DIF2
      IF(DABS(RATIO).LT..5D-4) GO TO 20
      ALPHA=DLOG(DABS(DIF1/DIF2))/(DIF1-DIF2)
      GO TO 30
   20 ALPHA=(((.25D0*RATIO+1.D0/3.D0)*RATIO+.5D0)*RATIO+
     A 1.D0)/DIF2
   30 VSA=VS(K)
      VSB=VS(K-1)
      SUM=SUM+(-VSA*DIFB+VSB*DIFA)*ALPHA
   50 CONTINUE
      V=SUM/PI
      RETURN
      END
C./       ADD   NAME=SORTEM
      SUBROUTINE SORTEM
      COMMON /BASEAG/ NTERMS,XILEAD(2),XI(1000),XITAIL(2),
     A                      VLEAD(2),V(1000),VTAIL(2)
      LSTRT2=1
      LSIZE=1
   10  LSTRT1=LSTRT2
       LSTOP1=LSTRT1+NTERMS-1
       LSTRT2=NTERMS+2-LSTRT2
       KC=LSTRT2
       KSTOPB=LSTRT1-1
   20   KSTRTA=KSTOPB+1
        KSTOPA=MIN0(KSTRTA+LSIZE-1,LSTOP1)
        KSTRTB=KSTOPA+1
        KSTOPB=MIN0(KSTRTB+LSIZE-1,LSTOP1)
        IF(KSTRTA.GT.KSTOPA) GO TO 90
   30    IF(KSTRTB.GT.KSTOPB) GO TO 70
          IF(KSTRTA.GT.KSTOPA) GO TO 50
          IF(XI(KSTRTA)-XI(KSTRTB)) 36,33,40
   33     IF(V(KSTRTA).GT.V(KSTRTB)) GO TO 40
   36     XI(KC)=XI(KSTRTA)
          V(KC)=V(KSTRTA)
          KC=KC+1
          KSTRTA=KSTRTA+1
          GO TO 30
   40     XI(KC)=XI(KSTRTB)
          V(KC)=V(KSTRTB)
          KC=KC+1
          KSTRTB=KSTRTB+1
          GO TO 30
   50    IF(KSTRTB.GT.KSTOPB) GO TO 20
         DO 60 K=KSTRTB,KSTOPB
         XI(KC)=XI(K)
         V(KC)=V(K)
         KC=KC+1
   60    CONTINUE
         GO TO 20
   70   IF(KSTRTA.GT.KSTOPA) GO TO 20
        DO 80 K=KSTRTA,KSTOPA
        XI(KC)=XI(K)
        V(KC)=V(K)
        KC=KC+1
   80   CONTINUE
        GO TO 20
   90  LSIZE=LSIZE+LSIZE
       IF(LSTRT2.NE.1) GO TO 10
       IF(LSIZE.LT.NTERMS) GO TO 10
      RETURN
      END
C./       ADD   NAME=AIR
C  ****************************************************************
C  *** PHYSICAL UTILITY ROUTINES - AIR,PHELEV,PHAZIM,EAMENU          ***
C  ****************************************************************
      SUBROUTINE AIR(Z)
      REAL*8 Z,H,ZFACT
      REAL*8 DH,H1,H2,T,DHW,H1W,H2W,SPD,THETA,ST,CT,DTHDH
      COMMON /PTH/ NPTH,PRESS(97),TMPMOL(97),GPHC(97),GAMMA(97)
      COMMON /WINDS/ NWINDS,GPHW(80),DIR(80),TURN(79),SPEED(80)
      COMMON /LYRDEF/NLAYER,GMZA(200),INDPTH(200),INDWND(200),
     ALYRPRT(200),KLAYER,ZTOP,ZBOT
      INTEGER*2 INDPTH,INDWND
      LOGICAL*1 LYRPRT
      COMMON /ATMCON/ REARTH,G0,RSTAR,R0M0,R0G0M0
      COMMON /ATMSPH/ GAM,C,U,V,DCDZ,DUDZ,DVDZ,D2CDZ2,D2UDZ2,D2VDZ2,RHO
      REAL*8 GAM,C,U,V
      REAL*8 RADPDG/1.74532925199433D-2/
      F1S(TAU)=(((TAU/5.+1.)*TAU/4.+1.)*TAU/3.+1.)*TAU/2.+1.
      F1A(TAU)=(EXP(TAU)-1.)/TAU
      NLPTH=INDPTH(KLAYER)
      NLWND=INDWND(KLAYER)
      ZFACT=1.D0+Z/REARTH
```

```
      H=Z/ZFACT
      DHDZ=1.D0/ZFACT**2
      D2HDZ2=-2.*DHDZ/(REARTH*ZFACT)
      DH=GPHC(NLPTH+1)-GPHC(NLPTH)
      H1=(H-GPHC(NLPTH))/DH
      H2=(GPHC(NLPTH+1)-H)/DH
      T=H1*TMPMOL(NLPTH+1)+H2*TMPMOL(NLPTH)
      DTDH=(TMPMOL(NLPTH+1)-TMPMOL(NLPTH))/DH
      GAM=H1*GAMMA(NLPTH+1)+H2*GAMMA(NLPTH)
      DGAMDH=(GAMMA(NLPTH+1)-GAMMA(NLPTH))/DH
      C=DSQRT(GAM*R0M0*T)
      DCDH=.5*C*(DTDH/T+DGAMDH/GAM)
      D2CDH2=-.25*C*(DGAMDH/GAM-DTDH/T)**2
      DCDZ=DCDH*DHDZ
      D2CDZ2=DCDH*D2HDZ2+D2CDH2*(DHDZ**2)
      TAU=DLOG(T/TMPMOL(NLPTH))
      IF(TAU.GT..1) GO TO 5
      FACTOR=TMPMOL(NLPTH)*F1S(TAU)
      GO TO 10
    5 FACTOR=TMPMOL(NLPTH)*F1A(TAU)
   10 PRS=PRESS(NLPTH)*DEXP(-H1*DH/(R0G0M0*FACTOR))
      RHO=PRS/(T*R0M0)
      DHW=GPHW(NLWND+1)-GPHW(NLWND)
      H1W=(H-GPHW(NLWND))/DHW
      H2W=(GPHW(NLWND+1)-H)/DHW
      SPD=H1W*SPEED(NLWND+1)+H2W*SPEED(NLWND)
      DSDH=(SPEED(NLWND+1)-SPEED(NLWND))/DHW
      DTHDH=TURN(NLWND)*RADPDG
      THETA=DIR(NLWND)*RADPDG+DTHDH*H1W*DHW
      CT=DCOS(THETA)
      ST=DSIN(THETA)
      U=-SPD*ST
      V=-SPD*CT
      DUDH=-SPD*CT*DTHDH-DSDH*ST
      DVDH=SPD*ST*DTHDH-DSDH*CT
      D2UDH2=DTHDH*(SPD*ST*DTHDH-2.*DSDH*CT)
      D2VDH2=DTHDH*(SPD*CT*DTHDH+2.*DSDH*ST)
      DUDZ=DUDH*DHDZ
      DVDZ=DVDH*DHDZ
      D2UDZ2=DUDH*D2HDZ2+D2UDH2*(DHDZ**2)
      D2VDZ2=DVDH*D2HDZ2+D2VDH2*(DHDZ**2)
      RETURN
      END
C./       ADD   NAME=PHELEV
      FUNCTION PHELEV(DUMMY)
      COMMON /RAYVAR/ZDIR,PKK,RTPAA0,ATTEN,SIGMA,X,Y,Z,DAGE,XF,YF,ZF,
     AXT,YT,ZT,XS,YS,ZS,XSS,YSS,ZSS,XSSS,YSSS,ZSSS,P3,P3F,P3T,P3S,
     BXFS,YFS,XTS,YTS,ZFTP3,XFTZ,YFTZ,ZFTZ,ZFA,ZTA,P3FTZ,P3FA,P3TA,AREA,
     CDAGDS
      REAL*8 SIGMA,X,Y,Z,DAGE,XF,YF,ZF,XT,YT,ZT
      DATA DGPRAD/57.295780/
      PHELEV=DGPRAD*ATAN2(P3,RTPAA0)
      RETURN
      END
C./       ADD   NAME=PHAZIM
      FUNCTION PHAZIM(DUMMY)
      COMMON /RAYNIT/ KGMH,NDCRVS,NUCRVS,IUPDWN,T0,PHI0,X0,Y0,Z0,
     AP10,P20,P30,OMEGA,DELTA0,P1F0,P2F0,P3F0,OMEGAF,XT0,YT0,ZT0,
     BP1T0,P2T0,P3T0,OMEGAT,XS0,YS0,ZS0,P3S0,RHO0,PCONST,NAGES,AGES(20)
      INTEGER*2 KGMH,NDCRVS,NUCRVS,IUPDWN
      DATA DGPRAD/57.295780/
      PHAZIM=DGPRAD*ATAN2(-P10,-P20)+180.
      RETURN
      END
C./       ADD   NAME=EAMENU
      SUBROUTINE EAMENU(ELEV,AZIM,MAG,EAST,NORTH,UP)
      REAL MAG,NORTH,DGPRAD/57.295780/
      HSQ=EAST**2+NORTH**2
      IF(HSQ.NE.0.) GO TO 5
      AZIM=0.
      GO TO 10
    5 AZIM=DGPRAD*ATAN2(-EAST,-NORTH)+180.
      IF(AZIM.LE.0.) AZIM=360.
   10 MAG=SQRT(HSQ+UP**2)
      IF(MAG.LE.0.) GO TO 20
      HORIZ=SQRT(HSQ)
      ELEV=DGPRAD*ATAN2(UP,HORIZ)
      RETURN
   20 ELEV=0.
      RETURN
      END
C./       ADD   NAME=UNITIS
C  *********************************************************************
C  *** GENERAL UTILITY ROUTINES - UNITIS,LOOKUP,FFA2N,LJUST          ***
C  ***                           FNDLYR,GETLYR,TIMCVR                ***
C  *********************************************************************
      SUBROUTINE UNITIS(GIVEN,TABLE,NTABS,LCUNIT,TYPE,IDEFLT)
      REAL*8 GIVEN,TABLE(NTABS),TYPE,BLANK/' '/
      CALL LOOKUP(8,NTABS,TABLE,GIVEN,LCUNIT,&5,&10)
      RETURN
    5 IF(GIVEN.EQ.BLANK) GO TO 15
      WRITE(6,7) GIVEN,TYPE,TABLE(LCUNIT)
    7 FORMAT(' AMBIGUOUS ABBREVIATION ''',A8,''' FOR ',A8,' UNIT. ''',
     AA8,''' ASSUMED.')
      RETURN
   10 WRITE(6,12) TYPE,GIVEN
   12 FORMAT(' INVALID ',A8,' UNIT SPECIFIED -''',A8,'''.')
      STOP 650
   15 LCUNIT=IDEFLT
      RETURN
      END
C./       ADD   NAME=LOOKUP
      SUBROUTINE LOOKUP(NCHAR,NTERMS,KTABL,KTEST,KTERM,*,*)
C   SEARCH TABLE OF CHAR STRINGS 'KTABL' FOR MATCH WITH 'KTEST
C   RETURN 1 FOR AMBIGUOUS ABBREVIATION
C   RETURN 2 FOR NO MATCH FOUND
C   NORMAL RETURN OR RETURN 1, MATCH STRING NUMBER IN 'KTERM'
      LOGICAL*1 KTABL(NCHAR,NTERMS), KTEST(NCHAR)
      LOGICAL*1 LCHECK(4),MCHECK(4)
      INTEGER CHECK1/0/,CHECK2/0/
      EQUIVALENCE (CHECK1,LCHECK(1)),(CHECK2,MCHECK(1))
      KPT1=1
      KPT2=NTERMS
      DO 50 L=1,NCHAR
      LCHECK(4)=KTEST(L)
      IF (CHECK1.LE.192) GO TO 55
      DO 20 KPT=KPT1,KPT2
      MCHECK(4)=KTABL(L,KPT)
      IF(CHECK1-CHECK2) 150,25,20
   20 CONTINUE
      GO TO 150
   25 KPT1=KPT
      DO 30 KPT=KPT1,KPT2
      MCHECK(4)=KTABL(L,KPT)
      IF (CHECK1.LT.CHECK2) GO TO 35
   30 CONTINUE
      KPT=KPT2+1
   35 KPT2=KPT-1
   50 CONTINUE
   55 KTERM=KPT1
      IF(KPT1.EQ.KPT2) RETURN
      MCHECK(4)=KTABL(L,KPT1)
      IF(CHECK2.EQ.64) RETURN
      RETURN 1
  150 RETURN 2
      END
C./       ADD   NAME=FFA2N
      SUBROUTINE FFA2N(STRING,INDEX,KWIDTH,NTERMS,ARRAY,DEFARY,KERR)
C   DECODES ALPHANUMERIC TO REAL*4 VARIABLES IN OUTPUT ARRAY 'ARRAY'.
C   ALPHANUMERIC INPUT IN ARRAY 'STRING', BEGINNING AT BYTE 'INDEX' IN
C   FIELDS OF LENGTH 'KWIDTH', FOR 'NTERMS' FIELDS. INPUT IS OF THE FORM
C   SDDDPDDD WHERE S IS '+' OR '-', DEFAULT TO '+', D IS A DIGIT 0-9,
C   AND P IS AN OPTIONAL '.'. IF '.' IS ABSENT, RESULT WILL BE AN
C   INTEGER. FOR MISSING FIELD (BLANK, OR ONLY DECIMAL POINT) THEN THE
C   CORRESPONDING ARRAY VALUE WILL DEFAULT TO THE DEFARY VALUE.
C   RESTRICTIONS: ONLY ONE SIGN S, WHICH MUST PRECEDE DIGITS AND DECIMAL
C   POINT.  AT MOST ONE DECIMAL POINT. INTERSPERSED BLANKS TREATED AS IF
C   NOT PRESENT (RATHER THAN AS IF ZERO). ALL OTHER CHARACTERS WILL BE
C   TREATED AS IF NOT THERE, EXCEPT KERR WILL BE SET EQUAL TO 1.
C   IF ALL FIELDS VALID DATA OR MISSING FIELDS, THEN KERR IS SET =0.
      REAL ARRAY(NTERMS),DEFARY(NTERMS)
      INTEGER FIELD/0/
      LOGICAL*1 KFLD(4),MISSNG,LSIGN,STRING(INDEX)
      EQUIVALENCE(KFLD(1),FIELD)
      KERR=0
      DO 100 M=1,NTERMS
      FRACT=0.
      XINT=0.
      SIGVAL=1.
      MISSNG=.TRUE.
      LSIGN=.FALSE.
      KPOINT=(M-1)*KWIDTH+INDEX-1
      DO 50 L=1,KWIDTH
      KFLD(4)=STRING(KPOINT+L)
C   FIELD --  64=SPACE  78='+' 96='-'  75='.'
      IF(FIELD.EQ.64) GO TO 50
      IF(FIELD.NE.78) GO TO 10
      IF(LSIGN) GO TO 40
      LSIGN=.TRUE.
      GO TO 50
   10 IF(FIELD.NE.96) GO TO 15
      IF(LSIGN) GO TO 40
      SIGVAL=-1.
      LSIGN=.TRUE.
      GO TO 50
   15 IF(FIELD.EQ.75) GO TO 55
      IF(FIELD.GE.240.AND.FIELD.Ljt250) GO TO 45
   40 KERR=1
      GO TO 50
   45 MISSNG=.FALSE.
      LSIGN=.TRUE.
      XINT=XINT*10.+(FIELD-240)
   50 CONTINUE
      GO TO 75
   55 LPOINT=INDEX+M*KWIDTH
      NPLACE=LPOINT-KPOINT-L-1
      IF(NPLACE.LE.0) GO TO 75
      DO 65 L=1,NPLACE
      KFLD(4)=STRING(LPOINT-L)
      IF(FIELD.EQ.64) GO TO 65
      IF(FIELD.GE.240.AND.FIELD.LT.250) GO TO 60
      KERR=1
      GO TO 65
   60 FRACT=(FRACT+(FIELD-240))/10.
      MISSNG=.FALSE.
   65 CONTINUE
   75 IF(MISSNG) GO TO 80
      ARRAY(M)=SIGN(XINT+FRACT,SIGVAL)
      GO TO 100
   80 ARRAY(M)=DEFARY(M)
      IF(LSIGN) KERR=1
  100 CONTINUE
      RETURN
      END
C./       ADD   NAME=FFN2A
      SUBROUTINE FFN2A(STRING,KPOS,NPLACE,NDEC,NTERMS,RNUM)
C CONVERTS NUMERIC VALUES IN REAL*4 ARRAY RNUM(NTERMS)
C TO ALPHANUMERIC CHARACTERS IN CHARACTER STRING "STRING".
C CHARACTERS BEGIN AT CHARACTER NUMBER "KPOS". FORMAT DEFINED
C AS EITHER FIXED, WITH "NPLACE" CHARACTERS FOR WIDTH, "NDEC"
```

```
C CHARACTERS AFTER DECIMAL POINT, OR FLOATING -
C BY CHOOSING "NPLACE" NEGATIVE, OF MAGNITUDE EQUAL TO THE NUMBER OF
C CHARACTERS AVAILABLE, WILL TRY TO FIT AN INTEGER, PLUS ENOUGH DECIMAL
C DIGITS TO MAKE "NDEC" SIGNIFICANT DIGITS.  IF NECESSARY, WILL TRIM
C DECIMAL PLACES TO FIT IN ABS(NPLACE) SPACES. RESULT WILL BE RIGHT-
C JUSTIFIED, WITH TRAILING ZEROES AND DECIMAL POINT TRIMMED.  IF
C INTEGER PORTION TOO LARGE TO FIT, RESULT IS ASTERISKS.
      INTEGER*2 CONVRT/0/,DIGITS(33)
      REAL*4 RNUM(NTERMS)
      LOGICAL*1 STRING(NTERMS),TEMPLT(2),MINUS/'-'/,BLANK/' '/,FLOAT
      LOGICAL*1 POINT/'.'/,STAR/'*'/,UP
      EQUIVALENCE (TEMPLT(1),CONVRT)
      KPLACE=IABS(NPLACE)
      KDEC=MIN0(16,MAX0(NDEC,0))
      FLOAT=NPLACE.LT.0
      DO 100 NN=1,NTERMS
      KSTART=KPOS+KPLACE*(NN-1)
      KSTOP=KSTART+KPLACE-1
      ANUM=ABS(RNUM(NN))
      BINT=AINT(ANUM)
      BFRACT=ANUM-BINT
C************FRACTION PART
      MSIG=17
      DO 10 K=1,17
      LDIG=BFRACT*10
      DIGITS(16+K)=LDIG
      BFRACT=BFRACT*10-LDIG
      IF(LDIG.GT.0) MSIG=MIN0(MSIG,K)
   10 CONTINUE
C*************INTEGER PART
      TEST=1.
      DO 20 K=1,16
      LDIG=AMOD(BINT,TEST*10.)/TEST
      BINT=BINT-LDIG*TEST
      IF(BINT.GE.0.) GO TO 15
      BINT=BINT+TEST
      LDIG=LDIG-1
   15 DIGITS(17-K)=LDIG
      TEST=TEST*10.
      IF(LDIG.GT.0) MSIG=1-K
   20 CONTINUE
      IF(BINT.GE.TEST) GO TO 90
C*****************ROUNDING
      NROUND=16+KDEC
      IF(FLOAT) NROUND=16+MAX0(0,MIN0(MSIG+KDEC-1,KPLACE-3))
      UP=DIGITS(NROUND+1).GE.5
      MAXK=0
      DO 22 K=1,NROUND
      LOOK=NROUND+1-K
      IF(UP) DIGITS(LOOK)=DIGITS(LOOK)+1
      UP=DIGITS(LOOK).GE.10
      IF(UP) DIGITS(LOOK)=DIGITS(LOOK)-10
      IF(DIGITS(LOOK).GT.0) MAXK=K
   22 CONTINUE
      IF(UP) GO TO 90
      MSIG=NROUND-MAXK-16
      INTPL=MAX0(1,-MSIG)
C***************BUILD STRING
      DO 25 K=KSTART,KSTOP
      STRING(K)=BLANK
   25 CONTINUE
      IF(FLOAT) GO TO 70
C******************FIXED FORMAT
      LPLACE=KDEC+2+INTPL
      IF(LPLACE.GT.KPLACE) GO TO 90
      STRING(KSTOP-KDEC)=POINT
      IF(RNUM(NN).LT.0.) STRING(KSTOP-LPLACE+1)=MINUS
      DO 35 K=1,INTPL
      CONVRT=240+DIGITS(17-K)
      STRING(KSTOP-KDEC-K)=TEMPLT(2)
   35 CONTINUE
      IF(KDEC.EQ.0) GO TO 100
      DO 40 K=1,KDEC
      CONVRT=240+DIGITS(16+K)
      STRING(KSTOP-KDEC+K)=TEMPLT(2)
   40 CONTINUE
      GO TO 100
C******************FLOAT FORMAT
   70 IF(INTPL+1.GT.KPLACE) GO TO 90
      IF(RNUM(NN).LT.0) STRING(KSTART)=MINUS
      DO 75 K=1,INTPL
      CONVRT=240+DIGITS(16+K-INTPL)
      STRING(KSTART+K)=TEMPLT(2)
   75 CONTINUE
      KFRCNZ=-1
      IFRCPL=MIN0(16,KPLACE-INTPL-2,MSIG+KDEC)
      IF(IFRCPL.LE.0) GO TO 82
      STRING(KSTART+INTPL+1)=POINT
      DO 80 K=1,IFRCPL
      CONVRT=240+DIGITS(16+K)
      STRING(KSTART+INTPL+K+1)=TEMPLT(2)
      IF(DIGITS(K+16).GT.0) KFRCNZ=K
   80 CONTINUE
C******************RIGHT JUSTIFY
   82 NSIG=INTPL+2+KFRCNZ
      IF(NSIG.EQ.KPLACE) GO TO 100
      DO 85 K=1,NSIG
      STRING(KSTOP-K+1)=STRING(KSTART+NSIG-K)
   85 CONTINUE
      NBLNK=KPLACE-NSIG
      DO 87 K=1,NBLNK
      STRING(KSTART+K-1)=BLANK
   87 CONTINUE
      GO TO 100
C******************NUMBER TOO LARGE FOR FORMAT
   90 DO 95 K=KSTART,KSTOP
      STRING(K)=STAR
   95 CONTINUE
  100 CONTINUE
      RETURN
      END
C./       ADD   NAME=LJUST
      SUBROUTINE LJUST(NCHAR,NSTR,STRIN,STROUT)
      LOGICAL*1 STRIN(NCHAR,NSTR),STROUT(NCHAR,NSTR)
      LOGICAL*1 LCHECK(4),BLANK/' '/
      INTEGER CHECK/0/
      EQUIVALENCE (CHECK,LCHECK(1))
      DO 50 KSTR=1,NSTR
      K=1
      DO 10 L=1,NCHAR
      LCHECK(4)=STRIN(L,KSTR)
      IF(CHECK.LE.128) GO TO 10
      CHECK=MOD(CHECK,64)+192
      IF(CHECK.EQ.240) GO TO 5
      ICHK=MOD(CHECK,16)
      IF(ICHK.EQ.0.OR.ICHK.GT.9) GO TO 10
    5 STROUT(K,KSTR)=LCHECK(4)
      K=K+1
   10 CONTINUE
      IF (K.GT.NCHAR) GO TO 50
      DO 20 L=K,NCHAR
      STROUT(L,KSTR)=BLANK
   20 CONTINUE
   50 CONTINUE
      RETURN
      END
C./       ADD   NAME=FNDLYR
      SUBROUTINE FNDLYR(Z,*)
      COMMON /LYRDEF/NLAYER,GMZA(200),INDPTH(200),INDWND(200),
     ALYRPRT(200),KLAYER,ZTOP,ZBOT
      INTEGER*2 INDPTH,INDWND
      LOGICAL*1 LYRPRT
      CALL GETLYR(Z,GMZA,NLAYER,KLAYER,&50)
      ZBOT=GMZA(KLAYER)
      ZTOP=GMZA(KLAYER+1)
      RETURN
   50 RETURN 1
      END
C./       ADD   NAME=GETLYR
      SUBROUTINE GETLYR(X,XTABL,NITEMS,NLAYR,*)
      DIMENSION XTABL(NITEMS)
      IF(X.LT.XTABL(1)) RETURN 1
      IF(XTABL(NITEMS).LT.X) RETURN 1
      N1=1
      N2=NITEMS-1
      IF(N2.LT.N1) RETURN 1
    2 NLAYR=(N1+N2+1)/2
      IF(N2.EQ.N1) GO TO 40
      IF(XTABL(NLAYR)-X) 5,40,10
    5 N1=NLAYR
      GO TO 2
   10 N2=NLAYR-1
      GO TO 2
   40 RETURN
      END
C./       ADD   NAME=TIMCVR
      FUNCTION TIMCVR(T,KDIR)
      REAL*8 T,HMS,SS,SSS,HHMMSS
      COMMON /PRINTS/ TITLE(30),KTPSIG,CVRTIM,TIMLBL
      REAL*8 TIMLBL,TIMCVR
      LOGICAL CVRTIM
      REAL*8 ROUND,X,XNEAR
      SSS(HMS)=-2400.D0*DINT(HMS/1E4)-40.D0*DINT(HMS/100.D0)+HMS
      HHMMSS(SS)=4000.D0*DINT(SS/3600.D0)+40.D0*DINT(SS/60.D0)+SS
      ROUND(X,XNEAR)=DSIGN(XNEAR*DINT(DABS(X/XNEAR)+.5D0),X)
      IF (.NOT.CVRTIM) GO TO 50
      IF (KDIR.LE.1) GO TO 30
      TIMCVR=HHMMSS(ROUND(T,.1D0))
      RETURN
   30 TIMCVR=SSS(T)
      RETURN
   50 TIMCVR=T
      RETURN
      END
C  *********************************************************************
C  ***  END OF MAIN TRAPS PROGRAMS                                   ***
C  *********************************************************************
```