NASA TECHNICAL MEMORANDUM

NASA TM X-3323

NASA TM X-3323

A COMPUTER PROGRAM FOR THE DETERMINATION OF THE ACOUSTIC PRESSURE SIGNATURE OF HELICOPTER ROTORS DUE TO BLADE THICKNESS

G. H. Mall and F. Farassat

Langley Research Center
Hampton, Va. 23665

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • JANUARY 1976

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA TM X-3323		

4. Title and Subtitle	5. Report Date
A COMPUTER PROGRAM FOR THE DETERMINATION OF THE ACOUSTIC PRESSURE SIGNATURE OF HELICOPTER ROTORS DUE TO BLADE THICKNESS	January 1976
	6. Performing Organization Code
7. Author(s)	8. Performing Organization Report No.
G. H. Mall and F. Farassat	L-10437
	10. Work Unit No.
9. Performing Organization Name and Address	505-10-26-02
NASA Langley Research Center Hampton, Va. 23665	11. Contract or Grant No.
	13. Type of Report and Period Covered
12. Sponsoring Agency Name and Address	Technical Memorandum
National Aeronautics and Space Administration Washington, D.C. 20546	14. Sponsoring Agency Code

15. Supplementary Notes

G. H. Mall: Computer Sciences Corporation.

F. Farassat: The George Washington University, Joint Institute for Acoustics and Flight Sciences.

16. Abstract

This report presents a computer program for the determination of the thickness noise of helicopter rotors. The results are obtained in the form of an acoustic pressure time history. The parameters of the program are the rotor geometry and the helicopter motion descriptors. The formulation employed is valid in the near and far fields. The blade planform must be rectangular, but the helicopter motion is arbitrary. The observer position is fixed with respect to the ground with a maximum elevation of 45^{o} above or below the rotor plane. With these restrictions, the program can also be used for the calculation of thickness noise of propellers.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Aeroacoustics Aerodynamic noise Helicopter rotor noise	Unclassified – Unlimited Subject Category 71

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	79	$4.75

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

CONTENTS

A COMPUTER PROGRAM FOR THE DETERMINATION OF THE ACOUSTIC PRESSURE SIGNATURE OF HELICOPTER ROTORS DUE TO BLADE THICKNESS

G. H. Mall* and F. Farassat**
Langley Research Center

SUMMARY

This report presents a computer program for the determination of the thickness noise of helicopter rotors. The results are obtained in the form of an acoustic pressure time history. The parameters of the program are the rotor geometry and the helicopter motion descriptors. The formulation employed is valid in the near and far fields. The blade planform must be rectangular, but the helicopter motion is arbitrary. The observer position is fixed with respect to the ground with a maximum elevation of 45° above or below the rotor plane. With these restrictions, the program can also be used for the calculation of thickness noise of propellers.

INTRODUCTION

Thickness noise of helicopter rotors or propellers is the noise generated by the normal velocity distribution on the surface of these rotating bodies. In the past, thickness noise of helicopter rotors and propellers has not been studied thoroughly even after the introduction of some approximations. This noise was investigated by Deming (ref. 1) and Arnoldi (ref. 2); the latter based his analysis on the work of Billing (ref. 3). Deming's and Arnoldi's analyses contain some of the following restrictions which should be removed in applications. These restrictions are of three types:

(1) The rotor or the propeller is required to be stationary or to move in axial direction at uniform speed.

(2) The observer is located in the far field.

(3) The acoustic source distribution must be compact.

---

*Computer Sciences Corporation.

**The George Washington University, Joint Institute for Acoustics and Flight Sciences.

For example, Deming's analysis is developed with restrictions 1 and 2. The expression derived in references 4 and 5 removes restrictions 1, 2, and 3. This paper documents a computer program developed to evaluate this expression.

The program, as presented, is for blades with rectangular planform. A variety of uniform airfoil sections can be defined by the user in the program input data. The observer is fixed with respect to the ground; this condition is preferred when experimental and theoretical results are compared. Numerical examples showing favorable comparison with some experimental results are presented in reference 5.

THE FORMULATION

Let the surface of the rotor system be specified by $f(\vec{Y},\tau) = 0$, where $\vec{Y}$ is a Cartesian frame fixed to the undisturbed medium and τ is the time. If v_n is the local normal velocity of the blade surface and p is the acoustic pressure, then the governing equation for the determination of the thickness noise is (refs. 4 and 5)

$$\frac{1}{c^2}\frac{\partial^2 p}{\partial \tau^2} - \nabla^2 p = \frac{\partial}{\partial \tau}\left[\rho_o v_n \left|\nabla f\right| \delta(f)\right] \tag{1}$$

where c is the speed of sound in the undisturbed medium, $\delta(f)$ is the Dirac delta function, and ρ_o is the density of the undisturbed medium. The solution of equation (1) is presented in reference 4 and in slightly more general form in reference 5.

In the present work, the $\vec{\eta}'$- and $\vec{\eta}$-frames in reference 5 are replaced by the $\vec{\eta}$- and $\overrightarrow{YH}$-frames, respectively. Consider one single blade and a nonrotating coordinate system: the $\overrightarrow{YH}$-frame is fixed to the center of rotation (fig. 1), and a rotating system is the $\vec{\eta}$-frame, fixed to the blade (fig. 2). Let $T(\eta_1,\eta_2)$ be the thickness function of the blade in this frame. Then the solution of equation (1) is given by equation (32) of reference 5 as follows:

$$p(\vec{X},t) = \frac{\rho_o c}{2\pi}\frac{\partial}{\partial t}\int_{\tau_1}^{\tau_2}\int_{\Gamma(D.P.)} \frac{T_1\tilde{V}_1 + T_2\tilde{V}_2}{r\left[1 - \hat{r}_3^{\,2} + T_1^{\,2}\left(1 - \hat{r}_1^{\,2}\right)\right]^{1/2}}\, d\Gamma\, d\tau \tag{2}$$

where

$p(\vec{X},t)$ acoustic pressure

ρ_o density of undisturbed medium

$\vec{X}$,t observer position and time

τ_1,τ_2 source times when the sphere $g = \tau - t + \left|\vec{X} - \vec{Y}\right|/c = 0$ enters and leaves the blade, respectively ($\vec{X}$ and t fixed), $\left|\vec{X} - \vec{Y}\right| = r$

$\vec{Y}$ source location $\vec{Y}$-frame

(VH_1,VH_2) vehicle velocity components along Y_1 and Y_2 axes of $\vec{Y}$-frame

Ω angular velocity of rotor

$(\hat{r}_1,\hat{r}_2,\hat{r}_3)$ unit vector along radiation direction in $\vec{\eta}$-frame

Γ curve of intersection of sphere $g = 0$ and mean surface of the blade

D.P. disk plane

T_1 $= \partial T/\partial\eta_1$, $T_2 = \partial T/\partial\eta_2$

$\tilde{V}_1$ $= -VH_1 + \eta_2\Omega$

$\tilde{V}_2$ $= -VH_2 - \eta_1\Omega$

For several blades, the contribution of each to the acoustic pressure is added linearly.

THE PROGRAM

General Discussion

For this program, the blades are assumed to have a rectangular planform with a uniform airfoil section along the span. The thickness distribution function $T(\eta_1,\eta_2)$ is denoted as $F(\eta_1)$ here since there is no dependence on η_2. The Rotor Thickness Noise program (RTN) evaluates the integral in equation (2) numerically. Evaluation of this double integral involves one integration over the curve of intersection of the sphere $g = 0$ and the rotor blade; the other integration is over source time. The first of these integrals is evaluated using a trapezoidal integration. The second integral is performed with Simpson's rule and a trapezoidal end-point correction where required. The Rotor Thickness Noise program postprocessor (RTNPP) completes the calculation of the acoustic pressure by differentiating the RTN output numerically with respect to observer time. The purpose of this section is to present detailed descriptions of the RTN and RTNPP programs, to

describe the manner in which input data must be prepared, and to describe the program output. Flow charts and FORTRAN listings of the programs are included as appendixes.

The solution technique employed by the RTN program is presented in the section of this report entitled "Method of Solution." Within this section, several key internal program variables are identified parenthetically. The RTN program reads a number of parameters describing the rotor system structure, the amount of detail to be used in performing the integrations, and various program options. A detailed description of the input required by the program is given in the section entitled "Preparation of Input Data." Output from the RTN program includes an echo of the input data and a table of ordered pairs $\left[\varphi(\vec{X},t),t\right]$ where t is the observer time and $\varphi(\vec{X},t)$ is defined by equation (3).

$$\varphi(\vec{X},t) = \frac{\rho_0 c}{2\pi} \int_{\tau_1}^{\tau_2} \int_{\Gamma(\text{D.P.})} \frac{\tilde{V}_1 F'(\eta_1)}{r\left[1 - \hat{r}_3^2 + F'^2(\eta_1)\left(1 - \hat{r}_1^2\right)\right]^{1/2}} \, d\Gamma \, d\tau \tag{3}$$

The section entitled "Postprocessor Description" discusses the numerical differentiation methodology used by the RTNPP program in computing the thickness noise from equation (3).

Method of Solution

Based upon the translational velocity of the rotor system (VH) and the specified initial value of source time (TAU), the location of the center of the rotor system relative to a fixed coordinate system is determined. This fixed coordinate system (Y) is described in figure 1. The origin of this system is the location of the center of the rotor system at time TAU = 0. The Y_1Y_2-plane contains the rotor disk and the Y_1-axis is parallel to the initial orientation of blade number one. The direction from the blade tip to the hub is the positive direction. A new reference frame (YH) is constructed with its origin coinciding with the center of the rotor system at time TAU. This nonrotating coordinate system remains fixed to the helicopter. The position of the observer at time TAU in the YH-frame (X) is computed from the specified location in the Y-frame (X0). The blade-fixed coordinate system is shown in figure 2.

At time TAU, a sphere is constructed with its center at the position of the observer with a radius (RM) equal to the maximum distance between the observer and any blade tip. The observer time (OTIME) is determined by the relation OTIME = TAU - RM/SNDSPD, where SNDSPD is the speed of sound. The radius of this sphere is subsequently permitted to collapse at the speed of sound. The points of intersection of the projection of this

sphere on the rotor system plane and the circle defined by the outer radii of the blades are determined as shown in figure 3. At the position of the observer, the more clockwise of these points (YSTART) defines the initial sweep angle (PHI0). The other point of intersection (YEND) defines the final sweep angle.

The integral over the intersection of the sphere and the rotor blade is performed by sweeping the arc of the intersection of the projection of the sphere and the rotor system plane. The arc is swept in a counterclockwise sense. To execute the sweep of such an arc efficiently, the rotor system is divided into four regions as illustrated in figure 4. These regions are defined as follows:

IREGION = 0 The point lies outside the circle defined by the outer radius of the blades.

= 1 The point lies between the circles defined by the outer and inner radii of the blades, but is not on a blade.

= 3 The point lies inside the circle defined by the inner (hub) radius of the blades.

= 6 The point lies on a blade.

Three sweep increments are used by the program. In region 1, a rather coarse angular increment (DELPHI1) is used to space across the region. When a blade is encountered, the intersection of the arc with the edge of the blade is computed. A smaller angular increment (DELPHI2) is used to sweep across the blade. An even smaller angular increment is allowed for points near the leading edge for blunt leading edges. The values of the integrand and differential increment are calculated for each point on a blade. The line integral (trapezoidal) is computed dynamically by cumulative summing. The individual contributions of each blade are also summed. The hub region is bypassed by using the computed value of the intersection points of the arc with the inner circle. The line integral is completed when the sweep across the arc reaches the final sweep angle.

When a sweep has been completed, the value of the sweep time (SWPTAU) is increased by a small increment (DELTAU). The center position of the rotor system relative to the observer is recomputed. A new sphere is constructed and its line integral is calculated. This process is continued until the sphere no longer intersects the rotor system. As these line integrals are calculated, they are dynamically accumulated into a Simpson's rule sum. The integrands are multiplied by two or four if they are even or odd terms, respectively, and are then added to the sum. If the last term is even, a trapezoidal correction is made.

After the last line integral has been calculated, the value of the double integral and the corresponding observer time are output. The value of the source time is increased

by some multiple of DELTAU and the entire process is repeated until one of three possible termination criteria is satisfied. These criteria are:

(1) TAU exceeds a specified maximum value.

(2) The radius of the contracting sphere becomes less than the outer radius of the blades.

(3) The angle of the observer relative to the center of the rotor system and the rotor system plane exceeds 45^{o}.

The hierarchy of program RTN and its 10 subprograms is illustrated in figure 5. Flow charts and FORTRAN listings of every RTN subprogram are presented in appendix A.

Preparation of Input Data

All input data to RTN are associated with the single NAMELIST name ROTOR. These variables are summarized in table I according to dimension, type, and units and are described in the following paragraphs.

CH — Blade chord.

COEFFS — The coefficients for describing the airfoil shape. The airfoil shape used in the subprogram FOFETA1 is given in nondimensional form by

$$F(E)/CH = T\left(C_1\sqrt{E} + C_2E + C_3E^2 + C_4E^3 + C_5E^4\right)$$

where $E = \eta_1/CH$, η_1 is the distance along the chord from the leading edge, CH is the blade chord, and T is the thickness ratio. The function F(E) is the airfoil thickness function which is the airfoil ordinate at η_1. This equation is suitable for many airfoil shapes, in particular for the NACA four digit airfoil series. The coefficients for this series (from ref. 6) are as follows:

$C_1 = 1.4845$

$C_2 = -0.6300$

$C_3 = -1.7580$

$C_4 = 1.4215$

$C_5 = -0.5075$

DELTET — The angular displacement of the rotor system between consecutive positions of the contracting sphere. This variable is used with the rotor system angular velocity to compute DELTAU, the time required for the rotor to rotate DELTET degrees. DELTAU is then used as the differ-

ential increment in the integration over time. The source time TAU is increased by some multiple of DELTAU between computations of $\Phi(\vec{X},t)$. A useful formula to select DELTET is

$$\text{DELTET} = \frac{(\text{CH})(\text{OMEGA})\cos\psi}{2.5(345 - \text{VH}\cos\psi)}$$

where ψ is the angle that the line from observer to the center of rotation makes with the vertical and VH is the helicopter speed.

DTAUM — A factor to determine the time increment between data points. The time increment derived from the variable DELTET is multiplied by DTAUM to determine the time interval between data points. Consequently, the rotor system rotates DTAUM*DELTET degrees between data points. A useful rule for selecting this factor is

$$\text{DTAUM} = \frac{4 \text{ to } 6}{(\text{NBLADES})(\text{DELTET})}$$

ETAMAX — The maximum η_1 value for which increased resolution is desired near the leading edge for airfoils with blunt leading edges. Generally, a value between 10 to 15 percent of the chord is sufficient.

JFLG — Rotor system translational motion indicator:

JFLG = 0 Hovering helicopter
= 1 Helicopter in forward flight

KFLG — Airfoil section type indicator:

KFLG = 0 Sharp leading edge
= 1 Blunt leading edge

MORDATA — Additional data indicator:

MORDATA = .T. Process another data set
= .F. Terminate

N — A factor to control the spacing along the arc for points not on a blade (i.e., to skip from one blade to the next). The variable N should not be made too small since a large sweep increment near a blade tip might skip over the blade completely. Approximately N points are used to traverse a distance along the arc equal to the blade chord. A value between 3 and 8 is suggested.

NBLADES	The number of blades in the rotor system. The program allows a maximum of eight blades.
NS	A factor to control spacing along the arc for points on a blade. The variable NS is used to compute the sweep angular increment, DELPHI2. The DELPHI2 determines the line integral differential increment. Approximately NS points are used to traverse a distance along the arc equal to the blade chord. A value between 15 and 25 is suggested.
OMEGA	Angular velocity of the rotor system.
PERDM	The number of periods of the blade rotation to be processed. Because of the lengthy computation time, this number should not be given a large value. This variable is used to establish the maximum time termination criterion.
R	Radius of the blades.
RO	Hub radius.
SNDSPD	The speed of sound.
TAUINT	The initial value of TAU, the source time. Nonzero values of this variable provide a restart capability as well as a means of generating high-resolution calculations in TAU regions of interest.
THKRAT	Blade thickness ratio.
X0	The location of the observer relative to (0,0,0), the initial position of the center of the rotor system at time TAU = 0 (i.e., the Y-frame).

Postprocessor Description

The relationship between the RTN program and the postprocessor (RTNPP) is illustrated in figure 6. Sample RTN output which is input to RTNPP is shown in figure 7. No input, other than the information generated on TAPE12 by RTN, is required by the postprocessor.

The Rotor Thickness Noise program postprocessor in use at Langley Research Center (LRC) performs three functions. The RTN output ordered pairs (φ,t) are used to find the thickness noise by differentiating them numerically with respect to t, the observer time. The resulting thickness noise is interpolated to provide a specification at equally spaced observer time values. The results are then plotted.

Because φ is the result of numerical integration, error resulting from finite resolution should be eliminated before attempting to differentiate the RTN output. This elimination is accomplished with a subroutine which fits a smooth cubic spline curve to the set

of data points. This routine requires a specification of the allowed variation of each functional value. For each data point, the difference between the raw data and a value obtained by trigonometric smoothing is used to satisfy this requirement.

The postprocessor output consists of four plots (illustrated in figs. 8 to 11) for each data set. The raw data as output by RTN are displayed in figure 8. The cubic spline curve is presented in figure 9. Figure 10 displays the smoothed data over the raw data to provide a measure of the quality of the cubic spline fit. Figure 11 shows the acoustic pressure time history.

Flow charts and FORTRAN listings of several RTNPP subprograms are presented in appendix B. Several routines required by RTNPP were obtained from the LRC math and graphics libraries. Descriptions of these subroutines are included in appendix B.

CONCLUDING REMARKS

This report presents a computer program for calculating the thickness noise of helicopter rotors with rectangular blades. Favorable comparisons with experimental results containing high-speed helicopter blade slap have been found and are reported in reference 5. The program is written in Control Data Corporation (CDC) FORTRAN and, although several postprocessor subroutines are not fully documented because of proprietary rights, sufficient information is supplied to adapt the program to any facility.

Langley Research Center
National Aeronautics and Space Administration
Hampton, Va. 23665
December 12, 1975

APPENDIX A

ROTOR THICKNESS NOISE PROGRAM

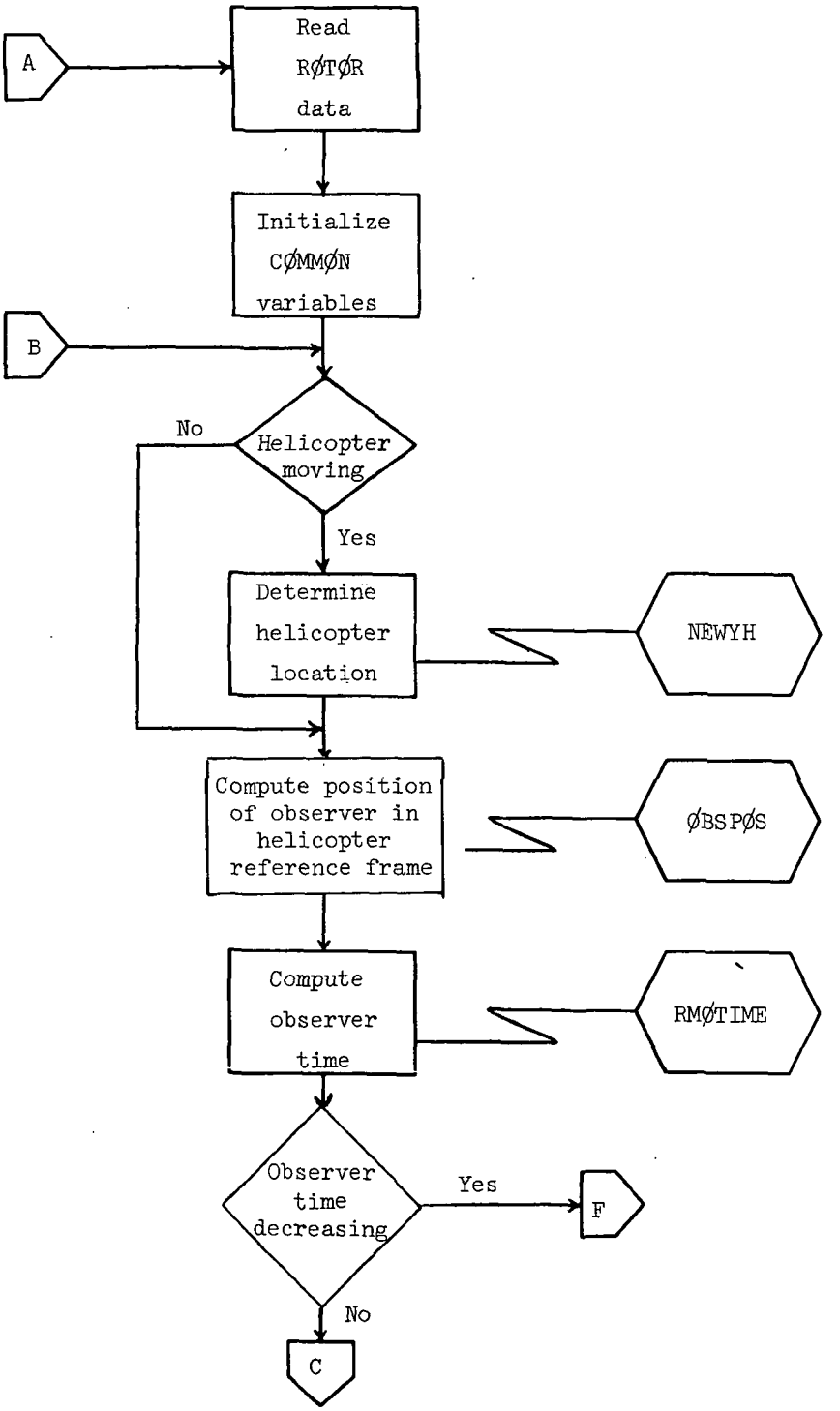
This appendix provides a brief description of the calculations performed by each RTN subprogram. The subprogram flow chart and FORTRAN listing follow the description.

Program RTN

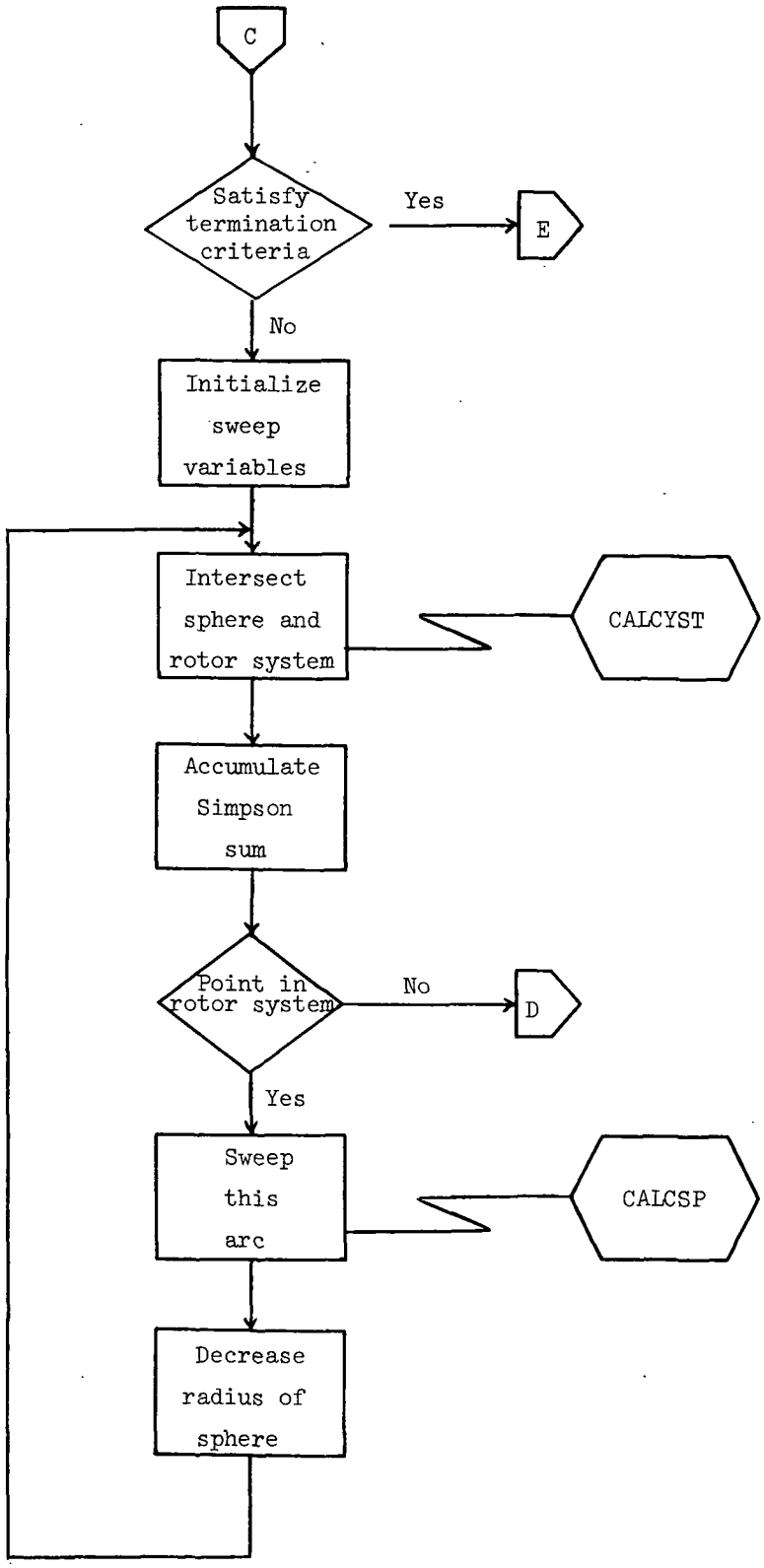
RTN reads and writes the NAMELIST input data and initializes other variables used elsewhere in the program. Following each contraction of the sphere, a check is made to see if any termination criteria have been satisfied or if the sphere has completed its pass through the rotor system. If the sphere is still intersecting the rotor system, an additional contribution is made to the Simpson's rule integral. If the sphere has passed out of the rotor system, the source time is incremented, the computed value of the double integral is output, and an additional sphere contraction is initiated. If one of the termination criteria has been satisfied, the program terminates unless an additional data set is specified.

APPENDIX A

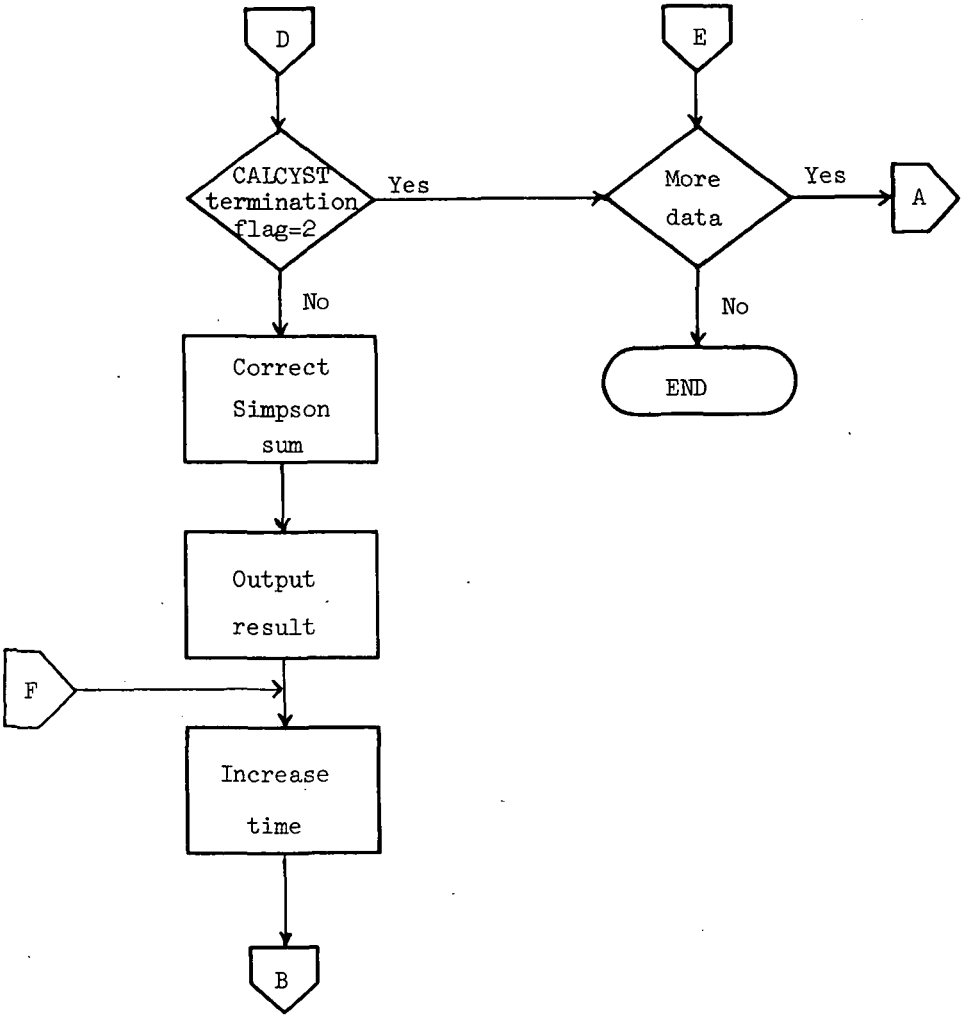
Flow Chart of Program RTN



APPENDIX A


C
Satisfy termination criteria
Yes
E
No
Initialize sweep variables
Intersect sphere and rotor system
CALCYST
Accumulate Simpson sum
Point in rotor system
No
D
Yes
Sweep this arc
CALCSP
Decrease radius of sphere

APPENDIX A


D
E
CALCYST termination flag=2
Yes
More data
Yes
A
No
No
END
Correct Simpson sum
Output result
F
Increase time
B

APPENDIX A

Listing of Program RTN

```
      PROGRAM RTN(OUTPUT=101B,INPUT=101B,TAPE12=OUTPUT,TAPE5=INPUT)      RTN      2
C*****************************************************************     RTN      3
C*                                                               *     RTN      4
C*            R O T O R   T H I C K N E S S   N O I S E          *     RTN      5
C*                                                               *     RTN      6
C*****************************************************************     RTN      7
C*                                                               *     RTN      8
C*                                                               *     RTN      9
C*    PURPOSE:                                                   *     RTN     10
C*                PROGRAM RTN DETERMINES THE ACOUSTIC PRESSURE   *     RTN     11
C*                GENERATED BY A ROTATING THICK BODY, SUCH AS AN *     RTN     12
C*                IDEALIZED RIGID ROTOR, WITH SOME SPECIFIED MOTION *  RTN     13
C*                RELATIVE TO AN OBSERVER AT SOME INITIAL DISTANCE *   RTN     14
C*                FROM THE BODY. THE PRESENT PROGRAM INCLUDES    *     RTN     15
C*                ROUTINES DEVELOPED SOLELY FOR ROTOR SYSTEMS WITH *   RTN     16
C*                RECTANGULAR BLADES AND UNIFORM AIRFOIL SECTION *     RTN     17
C*                ACROSS THE SPAN OF THE BLADES.                 *     RTN     18
C*                                                               *     RTN     19
C*                                                               *     RTN     20
C*     METHOD                                                    *     RTN     21
C*        OF                                                     *     RTN     22
C*   SOLUTION:                                                   *     RTN     23
C*                THE FORM OF THE SOLUTION UTILIZED BY THIS PROGRAM *  RTN     24
C*                INCLUDES A DOUBLE INTEGRAL WHERE ONE INTEGRATION *   RTN     25
C*                IS OVER THE INTERSECTION OF THE CONTRACTING    *     RTN     26
C*                SPHERE C*(TAU-T)+R=0 (CENTERED AT OBSERVER)    *     RTN     27
C*                AND THE BODY, AND THE OTHER INTEGRATION IS OVER *    RTN     28
C*                SOURCE TIME, TAU. THE FIRST OF THESE INTEGRALS IS *  RTN     29
C*                EVALUATED USING A TRAPEZOIDAL INTEGRATION. THE *     RTN     30
C*                SECOND INTEGRAL IS PERFORMED WITH SIMPSONS     *     RTN     31
C*                RULE AND A TRAPEZOIDAL END POINT CORRECTION    *     RTN     32
C*                WHERE REQUIRED.                                *     RTN     33
C*                                                               *     RTN     34
C*                BEGINNING WITH AN INITIAL VALUE OF TIME        *     RTN     35
C*                (TAU), THE TRANSLATIONAL VELOCITY OF THE ROTOR *     RTN     36
C*                SYSTEM IS USED TO LOCATE THE POSITION OF ITS   *     RTN     37
C*                CENTER. THE LOCATION OF THE OBSERVER RELATIVE  *     RTN     38
C*                TO THIS CENTER POSITION IS DETERMINED AND A    *     RTN     39
C*                SPHERE IS CONSTRUCTED (CENTERED AT THE OBSERVER) *   RTN     40
C*                WITH A RADIUS EQUAL TO THE MAXIMUM DISTANCE    *     RTN     41
C*                BETWEEN THE OBSERVER AND ANY BLADE TIP. THIS   *     RTN     42
C*                TIME REPRESENTS THE LOWER LIMIT ON THE TIME    *     RTN     43
C*                INTEGRAL AND THIS SPHERE REPRESENTS THE INITIAL *    RTN     44
C*                POSITION OF THE CONTRACTING SPHERE. THE POINTS *     RTN     45
C*                OF INTERSECTION OF THE PROJECTION OF THIS      *     RTN     46
C*                SPHERE ON THE ROTOR SYSTEM PLANE AND THE CIRCLE *    RTN     47
C*                DEFINED BY THE BLADES OUTER RADIUS ARE DETERMINED. * RTN     48
C*                AT THE OBSERVERS POSITION THE MORE CLOCKWISE OF *    RTN     49
C*                THESE POINTS DEFINES THE INITIAL SWEEP ANGLE.  *     RTN     50
C*                THE ARC THROUGH THE ROTOR SYSTEM PLANE COR-    *     RTN     51
C*                RESPONDING TO THIS POSITION OF THE SPHERE IS   *     RTN     52
C*                SWEPT IN A COUNTERCLOCKWISE SENSE TO COMPUTE   *     RTN     53
C*                THE LINE INTEGRAL FOR THIS VALUE OF SOURCE TIME. *   RTN     54
C*                                                               *     RTN     55
C*                TO DECREASE EXECUTION TIME, TWO SWEEP INCREMENTS *   RTN     56
C*                ARE USED BY THE PROGRAM. FOR POINTS ON THE ARC *     RTN     57
C*                WHICH ARE NOT ON A BLADE A LARGE ANGULAR INCREMENT * RTN     58
C*                IS USED TO SPACE ACROSS THESE REGIONS. SMALLER *     RTN     59
C*                ANGULAR INCREMENTS ARE AUTOMATICALLY USED BY THE *   RTN     60
C*                PROGRAM WHEN A BLADE IS ENCOUNTERED. FOR BLUNT *     RTN     61
C*                EDGES A SMALLER INCREMENT MAY BE GENERATED FOR *     RTN     62
C*                POINTS BETWEEN THE LEADING EDGE AND A MAXIMUM VALUE * RTN     63
C*                OF THE DISTANCE FROM THE LEADING EDGE. FOR EACH *    RTN     64
C*                POINT ON A BLADE THE VALUE OF THE INTEGRAND AND *    RTN     65
C*                DIFFERENTIAL INCREMENT ARE COMPUTED. THE LINE  *     RTN     66
```

```
C*                INTEGRAL (TRAPEZOIDAL) IS COMPUTED DYNAMICALLY      * RTN      67
C*                BY CUMULATIVE SUMMING. WHEN A POINT ON THE ARC      * RTN      68
C*                IS REACHED WHICH IS OUTSIDE THE CIRCLE DEFINED      * RTN      69
C*                BY THE ROTOR SYSTEM, THE VALUE OF THE RESULTANT     * RTN      70
C*                INTEGRAL IS MULTIPLIED BY THE APPROPRIATE PHYSICAL  * RTN      71
C*                CONSTANT.                                           * RTN      72
C*                                                                    * RTN      73
C*                THE VALUE OF SOURCE TIME IS INCREASED SLIGHTLY.     * RTN      74
C*                THE CENTER POSITION OF THE ROTOR SYSTEM RELATIVE    * RTN      75
C*                TO THE OBSERVER IS RECOMPUTED. A NEW SPHERE IS      * RTN      76
C*                CONSTRUCTED AND THIS LINE INTEGRAL IS CALCULATED.   * RTN      77
C*                THIS PROCESS IS CONTINUED UNTIL THE PROJECTION      * RTN      78
C*                OF THE SPHERE HAS COMPLETELY PASSED THROUGH THE     * RTN      79
C*                ROTOR SYSTEM. THE FINAL VALUE OF SOURCE TIME        * RTN      80
C*                REPRESENTS THE UPPER LIMIT ON THE TIME INTEGRAL.    * RTN      81
C*                                                                    * RTN      82
C*                SINCE THE SOURCE TIME DIFFERENTIAL INCREMENT IS     * RTN      83
C*                CONSTANT, A SIMPSON RULE INTEGRATION IS POSSIBLE.   * RTN      84
C*                THIS INTEGRAL IS PERFORMED DYNAMICALLY AS A         * RTN      85
C*                CUMULATIVE SUM. AS LINE INTEGRALS ARE CALCULATED    * RTN      86
C*                THEY ARE MULTIPLIED BY TWO(FOUR) IF THEY ARE        * RTN      87
C*                EVEN(ODD) TERMS AND ADDED TO THE SUM. IF THE LAST   * RTN      88
C*                TERM IS EVEN, A TRAPEZOIDAL CORRECTION IS MADE.     * RTN      89
C*                                                                    * RTN      90
C*                THE VALUE OF TAU IS INCREASED AND THE ENTIRE        * RTN      91
C*                PROCESS IS REPEATED UNTIL ONE OF THREE POSSIBLE     * RTN      92
C*                TERMINATION CRITERIA IS SATISFIED. THESE CRITERIA   * RTN      93
C*                ARE:                                                * RTN      94
C*                                                                    * RTN      95
C*                  1-TAU EXCEEDS A SPECIFIED MAXIMUM VALUE           * RTN      96
C*                                                                    * RTN      97
C*                  2-THE RADIUS OF THE CONTRACTING SPHERE            * RTN      98
C*                    BECOMES LESS THAN THE RADIUS OF THE BLADES      * RTN      99
C*                    OUTER RADIUS                                    * RTN     100
C*                                                                    * RTN     101
C*                  3-THE ANGLE OF THE OBSERVER RELATIVE TO           * RTN     102
C*                    THE CENTER OF THE ROTOR SYSTEM AND THE          * RTN     103
C*                    ROTOR SYSTEM PLANE EXCEEDS 45 DEGREES           * RTN     104
C*                                                                    * RTN     105
C*                ONCE ONE OF THE TERMINATION CRITERIA IS SATISFIED   * RTN     106
C*                THE PROGRAM TERMINATES UNLESS AN ADDITIONAL DATA    * RTN     107
C*                SET IS SPECIFIED. NUMERICAL DIFFERENTIATION OF THE  * RTN     108
C*                OUTPUT, AS WELL AS PLOTTING THE RESULTS AND         * RTN     109
C*                FORMATTING THE RESULTS FOR SUBSEQUENT ANALYSIS      * RTN     110
C*                IS PERFORMED IN A COMPANION POST-PROCESSOR.         * RTN     111
C*                                                                    * RTN     112
C*                                                                    * RTN     113
C* PARAMETERS:                                                        * RTN     114
C*                                                                    * RTN     115
C*          XO    THE LOCATION OF THE OBSERVER RELATIVE TO (0,0,0),   * RTN     116
C*                THE INITIAL POSITION OF THE ROTOR SYSTEM AT TIME    * RTN     117
C*                TAU = 0. (METERS)                                   * RTN     118
C*                                                                    * RTN     119
C*      NBLADES   THE NUMBER OF BLADES IN THE ROTOR SYSTEM. THE       * RTN     120
C*                PROGRAM ALLOWS A MAXIMUM OF 8 BLADES.               * RTN     121
C*                                                                    * RTN     122
C*           R    RADIUS OF THE BLADES. (METERS)                      * RTN     123
C*                                                                    * RTN     124
C*          CH    BLADE CHORD. (METERS)                               * RTN     125
C*                                                                    * RTN     126
C*       OMEGA    ANGULAR VELOCITY OF THE ROTOR SYSTEM. (REVOLUTIONS  * RTN     127
C*                PER MINUTE)                                         * RTN     128
C*                                                                    * RTN     129
C*          RO    HUB RADIUS. (METERS)                                * RTN     130
C*                                                                    * RTN     131
C*      THKRAT    BLADE THICKNESS RATIO.                              * RTN     132
C*                                                                    * RTN     133
```

```
C*      DELTET      ANGULAR DISPLACEMENT OF THE ROTOR SYSTEM BETWEEN    * RTN      134
C*                  CONSECUTIVE POSITIONS OF THE CONTRACTING SPHERE.    * RTN      135
C*                  THIS VARIABLE IS USED WITH THE ROTOR SYSTEM ANGULAR * RTN      136
C*                  VELOCITY TO COMPUTE THE SOURCE TIME DIFFERENTIAL    * RTN      137
C*                  INCREMENT, DELTAU. (DEGREES)                       * RTN      138
C*                                                                     * RTN      139
C*            N     FACTOR TO CONTROL SPACING ALONG ARC BETWEEN BLADES  * RTN      140
C*                  (I.E., TO SKIP FROM ONE BLADE TO THE NEXT).        * RTN      141
C*                  APPROXIMATELY N POINTS ARE USED TO TRAVERSE A      * RTN      142
C*                  DISTANCE ALONG THE ARC EQUAL TO THE BLADE CHORD.   * RTN      143
C*                                                                     * RTN      144
C*           NS     FACTOR TO CONTROL SPACING ALONG ARC ON A BLADE.     * RTN      145
C*                  APPROXIMATELY NS POINTS ARE USED TO TRAVERSE A     * RTN      146
C*                  DISTANCE ALONG THE ARC EQUAL TO THE BLADE CHORD.   * RTN      147
C*                  THIS VARIABLE DETERMINES THE LINE INTEGRAL         * RTN      148
C*                  DIFFERENTIAL INCREMENTS.                           * RTN      149
C*                                                                     * RTN      150
C*         KFLG     AIRFOIL SECTION TYPE INDICATOR.                    * RTN      151
C*                                                                     * RTN      152
C*                    KFLG = 0:  SHARP LEADING EDGE                    * RTN      153
C*                                                                     * RTN      154
C*                         = 1:  BLUNT LEADING EDGE                    * RTN      155
C*                                                                     * RTN      156
C*       ETAMAX     MAXIMUM ETA VALUE FOR WHICH INCREASED RESOLUTION   * RTN      157
C*                  IS DESIRED NEAR THE LEADING EDGE FOR BLUNT LEADING * RTN      158
C*                  EDGES.                                             * RTN      159
C*                                                                     * RTN      160
C*        BLNTN     FACTOR TO GENERATE FINE SPACING NEAR THE LEADING   * RTN      161
C*                  EDGE FOR BLUNT LEADING EDGES. FOR ETA VALUES       * RTN      162
C*                  BETWEEN ZERO AND ETAMAX, THE SPACING FACTOR NS IS  * RTN      163
C*                  MULTIPLIED BY BLNTN.                               * RTN      164
C*                                                                     * RTN      165
C*         JFLG     ROTOR SYSTEM TRANSLATIONAL MOTION INDICATOR.       * RTN      166
C*                                                                     * RTN      167
C*                    JFLG = 0:  HOVERING HELICOPTER.                  * RTN      168
C*                                                                     * RTN      169
C*                         = 1:  HELICOPTER IN FORWARD FLIGHT.         * RTN      170
C*                                                                     * RTN      171
C*       COEFFS     COEFFICIENTS IN THE FUNCTION DESCRIBING THE AIRFOIL * RTN      172
C*                  SECTION. A MINUS SIGN MUST APPEAR EXPLICITLY FOR   * RTN      173
C*                  NEGATIVE TERMS.                                    * RTN      174
C*                                                                     * RTN      175
C*       TAUINT     THE INITIAL VALUE OF SOURCE TIME (TAU). (SECONDS)  * RTN      176
C*                                                                     * RTN      177
C*        PERDM     THE NUMBER OF PERIODS OF THE ROTOR SYSTEM TO BE    * RTN      178
C*                  PROCESSED. THIS VARIABLE IS USED TO ESTABLISH      * RTN      179
C*                  THE MAXIMUM TIME TERMINATION CRITERION.            * RTN      180
C*                                                                     * RTN      181
C*        DTAUM     TIME INCREMENT BETWEEN DATA POINTS. THE SOURCE     * RTN      182
C*                  TIME DIFFERENTIAL INCREMENT DERIVED FROM DELTET    * RTN      183
C*                  IS MULTIPLIED BY DTAUM TO DETERMINE THIS INCREMENT. * RTN      184
C*                                                                     * RTN      185
C*       SNDSPD     SPEED OF SOUND. (METERS/SEC)                       * RTN      186
C*                                                                     * RTN      187
C*      MORDATA     A LOGICAL VARIABLE TO SPECIFY IF AN ADDITIONAL     * RTN      188
C*                  DATA SET IS TO BE PROCESSED.                       * RTN      189
C*                                                                     * RTN      190
C*                                                                     * RTN      191
C*  REQUIRED                                                           * RTN      192
C*  ROUTINES:                                                          * RTN      193
C*                  SUBROUTINE NEWVH                                   * RTN      194
C*                  SUBROUTINE CALCVH (SEE ADDITIONAL DETAILS)         * RTN      195
C*                  SUBROUTINE ORSPOS                                  * RTN      196
C*                  SUBROUTINE RMOTIME                                 * RTN      197
C*                  SUBROUTINE CALCYST                                 * RTN      198
C*                  SUBROUTINE CALCSP                                  * RTN      199
C*                  FUNCTION FOFETA1 (SEE ADDITIONAL DETAILS)          * RTN      200
```

APPENDIX A

```
C*                FUNCTION PFUNCN                                    * RTN      201
C*                SUBROUTINE TEST (SEE ADDITIONAL DETAILS)           * RTN      202
C*                SUBROUTINE LEETA                                   * RTN      203
C*                                                                   * RTN      204
C*                                                                   * RTN      205
C* ADDITIONAL                                                        * RTN      206
C*    DETAILS:                                                       * RTN      207
C*                FOR A MOVING ROTOR SYSTEM (JFLG = 1), A SUBROUTINE * RTN      208
C*                MUST BE PROVIDED TO DESCRIBE THE MOTION.           * RTN      209
C*                                                                   * RTN      210
C*                  SUBROUTINE CALCVH(VH,TIME)                       * RTN      211
C*                                                                   * RTN      212
C*                    SUBROUTINE CALCVH DEFINES THE VECTOR VH(3)     * RTN      213
C*                    WHICH MUST CONTAIN THE THREE COMPONENTS OF     * RTN      214
C*                    THE ROTOR SYSTEM TRANSLATIONAL VELOCITY AT     * RTN      215
C*                    T = TIME. THESE COMPONENTS ARE TO BE SPECIFIED * RTN      216
C*                    IN THE ROTOR SYSTEM REFERENCE FRAME. THE THREE * RTN      217
C*                    AXES IN THIS COORDINATE SYSTEM ARE DEFINED AS  * RTN      218
C*                    FOLLOWS.                                       * RTN      219
C*                                                                   * RTN      220
C*                      THE Y1 AXIS IS PARALLEL TO THE INITIAL       * RTN      221
C*                      ORIENTATION OF BLADE NUMBER ONE WITH THE     * RTN      222
C*                      DIRECTION FROM THE BLADE TIP TO THE CENTER   * RTN      223
C*                      OF THE ROTOR SYSTEM TAKEN AS POSITIVE.       * RTN      224
C*                                                                   * RTN      225
C*                      THE Y2 AXIS IS PERPENDICULAR TO Y1 PASSING   * RTN      226
C*                      THROUGH THE CENTER OF THE ROTOR SYSTEM AND   * RTN      227
C*                      LYING IN THE ROTOR SYSTEM PLANE. THE         * RTN      228
C*                      DIRECTION TO THE LEFT OF AN OBSERVER FACING  * RTN      229
C*                      TOWARD POSITIVE Y1 IS TAKEN AS POSITIVE.     * RTN      230
C*                                                                   * RTN      231
C*                      THE Y3 AXIS IS PERPENDICULAR TO THE PLANE    * RTN      232
C*                      OF THE ROTOR SYSTEM WITH THE POSITIVE        * RTN      233
C*                      DIRECTION DETERMINED BY THE USUAL CONVENTION * RTN      234
C*                      FOR A RIGHT-HANDED COORDINATE SYSTEM.        * RTN      235
C*                                                                   * RTN      236
C*                THE PRESENT VERSION OF FUNCTION FOFETA1 IMPOSES    * RTN      237
C*                A LIMITATION TO UNIFORM AIRFOIL SECTION ACROSS THE * RTN      238
C*                SPAN OF THE BLADES. FOR NONUNIFORM AIRFOIL SECTION * RTN      239
C*                THIS FUNCTION MUST BE MODIFIED.                    * RTN      240
C*                                                                   * RTN      241
C*                THE PRESENT VERSION OF SUBROUTINE TEST IMPOSES A   * RTN      242
C*                LIMITATION TO ROTOR SYSTEMS WITH RECTANGULAR       * RTN      243
C*                BLADES. FOR OTHER BLADE SHAPES THIS SUBROUTINE     * RTN      244
C*                MUST BE MODIFIED.                                  * RTN      245
C*                                                                   * RTN      246
C*                                                                   * RTN      247
C*     SOURCE:                                                       * RTN      248
C*                RTN WAS PROGRAMMED AT LANGLEY RESEARCH CENTER BY   * RTN      249
C*                COMPUTER SCIENCES CORPORATION BASED UPON A DESIGN  * RTN      250
C*                BY F. FARASSAT.                                    * RTN      251
C*                                                                   * RTN      252
C*                                                                   * RTN      253
C*   LANGUAGE:                                                       * RTN      254
C*                FORTRAN                                            * RTN      255
C*                                                                   * RTN      256
C*                                                                   * RTN      257
C*      DATE                                                         * RTN      258
C*   RELEASED:                                                       * RTN      259
C*                MAY 1, 1974                                        * RTN      260
C*                                                                   * RTN      261
C*                                                                   * RTN      262
C*     LATEST                                                        * RTN      263
C*   REVISION:                                                       * RTN      264
C*                DECEMBER 16, 1975                                  * RTN      265
C*                                                                   * RTN      266
C*                                                                   * RTN      267
C*******************************************************************  RTN      268
```

```
      LOGICAL MORDATA                                                   RTN      269
      DIMENSION    BETA(8),        COEFFS(5),      ETA(2,8),            RTN      270
     1             X(3),           XO(3),          Y(3),                RTN      271
     2             YEND(2),        YH(3),          YSTART(2),           RTN      272
     3             VH(3)                                                RTN      273
      COMMON        ALFA,      BETA,      BLADES,    BLNTN,     CBETA,  RTN      274
     1              CH,        COEFFS,    DELTAU,    ETA,       ETA2,   RTN      275
     2              ETAMAX,    FAC,       FN,        FNS,       FETA1,  RTN      276
     3              IFLAG,     IREGION,   J2,        JFLG,      KFLG,   RTN      277
     4              NBLADES,   OMEGAR,    OTIME,     PI,        R,      RTN      278
     5              RM,        RMIN,      S,         SBETA,     SNOSPD, RTN      279
     6              SRCOST,    SWPCST,    SWPTAU,    TAU,       THKRAT, RTN      280
     7              TIMTHRU,   VH,        X,         XO,        YEND,   RTN      281
     8              YH,        YSTART,    DELETA1,   DELF               RTN      282
      NAMELIST/ROTOR/XO,NBLADES,R,CH,OMEGA,RO,THKRAT,DELTET,N,NS,KFLG,  RTN      283
     1 ETAMAX,BLNTN,JFLG,COEFFS,TAUINT,PERDM,DTAUM,SNOSPD,MORDATA       RTN      284
C                                                                       RTN      285
C        READ NAMELIST DATA                                             RTN      286
C                                                                       RTN      287
   10 READ(5,ROTOR)                                                     RTN      288
      WRITE(12,ROTOR)                                                   RTN      289
C                                                                       RTN      290
C        INITALIZE VARIABLES                                            RTN      291
C                                                                       RTN      292
      PI = 3.14159265358979                                             RTN      293
      OMEGAR = 2.*OMEGA*PI/60.                                          RTN      294
      TESTIM = 0.                                                       RTN      295
      YH(1) = 0.                                                        RTN      296
      YH(2) = 0.                                                        RTN      297
      YH(3) = 0.                                                        RTN      298
      Y(1) = 0.                                                         RTN      299
      Y(2) = 0.                                                         RTN      300
      Y(3) = 0.                                                         RTN      301
      VH(1) = 0.                                                        RTN      302
      VH(2) = 0.                                                        RTN      303
      VH(3) = 0.                                                        RTN      304
      ALFA = ASIN(0.5*CH/R)                                             RTN      305
      RMIN = R                                                          RTN      306
      RSQ=R*R                                                           RTN      307
      ROSQ=RO*RO                                                        RTN      308
      TAU = TAUINT                                                      RTN      309
      IF(KFLG.EQ.0) BLNTN = 1.                                          RTN      310
      NPTS=0                                                            RTN      311
      PKEP1 = 0.                                                        RTN      312
      PKEP2 = 0.                                                        RTN      313
      SCRIP1 = 0.                                                       RTN      314
      SCRIPTP=0.0                                                       RTN      315
      BLADES = FLOAT(NBLADES)                                           RTN      316
C                                                                       RTN      317
C        COMPUTE SOURCE TIME DIFFERENTIAL INCREMENT AND MAXIMUM TIME    RTN      318
C          TERMINATION CRITERION                                        RTN      319
C                                                                       RTN      320
      DELTAU = DELTET/(6.*OMEGA)                                        RTN      321
      PERIOD = 60./(OMEGA*BLADES)                                       RTN      322
      TAUMAX = TAUINT + PERDM*PERIOD                                    RTN      323
C                                                                       RTN      324
C        S. N. 20 BEGINS A NEW DATA POINT                               RTN      325
C                                                                       RTN      326
   20 IF(JFLG.NE.0) CALL NEWYH(TAU)                                     RTN      327
C                                                                       RTN      328
C        DETERMINE RELATIVE LOCATIONS OF ROTOR SYSTEM AND OBSERVER      RTN      329
C                                                                       RTN      330
      CALL OBSPOS                                                       RTN      331
C                                                                       RTN      332
C        CONSTRUCT INITIAL SPHERE                                       RTN      333
C                                                                       RTN      334
      CALL RMOTIME                                                      RTN      335
      DTIM = OTIME - TESTIM                                             RTN      336
C                                                                       RTN      337
```

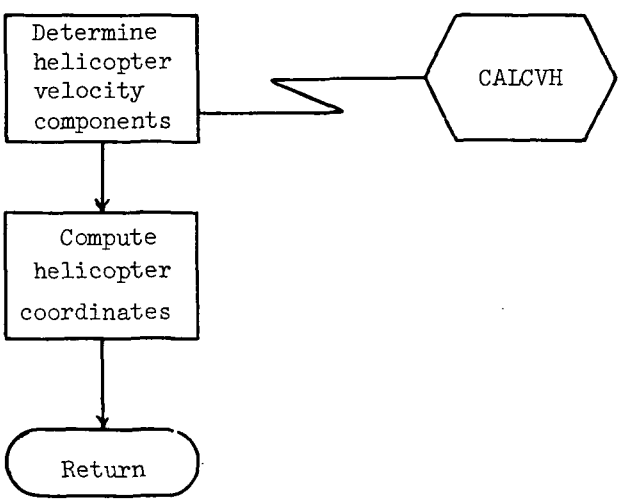
```
C        IF OBSERVER TIME IS DECREASING SKIP TO NEXT DATA POINT          RTN      338
C                                                                        RTN      339
      IF(DTIM.LE.0.) GO TO 40                                            RTN      340
      TESTIM = OTIME                                                     RTN      341
      OBSANGL = ASIN(X(3)/SQRT(X(1)**2 + X(2)**2 + X(3)**2))             RTN      342
C                                                                        RTN      343
C        IF ANY TERMINATION CRITERION IS SATISFIED SKIP TO THE           RTN      344
C          CHECK FOR ADDITIONAL DATA                                     RTN      345
C                                                                        RTN      346
      IF(OBSANGL.GT.(PI/4.)) GO TO 60                                    RTN      347
      IF(RM.LT.RMIN) GO TO 60                                            RTN      348
      IF(TAU.GT.TAUMAX) GO TO 60                                         RTN      349
C                                                                        RTN      350
C        INITIALIZE SWEEP VARIABLES                                      RTN      351
C                                                                        RTN      352
      SWPTAU=TAU                                                         RTN      353
      TIMTHRU=0.0                                                        RTN      354
      IEVEN = -1                                                         RTN      355
C                                                                        RTN      356
C        S. N. 30 BEGINS A NEW POSITION OF THE SPHERE                    RTN      357
C                                                                        RTN      358
   30 TIMTHRU = TIMTHRU + 1.                                             RTN      359
      IEVEN = -IEVEN                                                     RTN      360
      FN = N                                                             RTN      361
      FNS = NS                                                           RTN      362
      CALL CALCYST(RSQ,PHI0)                                             RTN      363
      PHI = PHI0                                                         RTN      364
C                                                                        RTN      365
C        ADD TO CUMULATIVE SIMPSON SUM                                   RTN      366
C                                                                        RTN      367
      PKEP2 = PKEP1                                                      RTN      368
      PKEP1 = SCRIP1                                                     RTN      369
      PDELT = 2.*SCRIP1                                                  RTN      370
      IF(IEVEN.GT.0) PDELT = 2.*PDELT                                    RTN      371
      IF(TIMTHRU.LT.2.5) PDELT = SCRIP1                                  RTN      372
      SCRIPTP = SCRIPTP + PDELT                                          RTN      373
C                                                                        RTN      374
C        IF IFLAG=0, WE ARE IN THE ROTOR SYSTEM                          RTN      375
C                                                                        RTN      376
      IF(IFLAG.EQ.0) GO TO 50                                            RTN      377
C                                                                        RTN      378
C        IF IFLAG=2, A TERMINATION CRITERION WAS DETECTED IN CALCYST     RTN      379
C                                                                        RTN      380
      IF(IFLAG.EQ.2) GO TO 60                                            RTN      381
C                                                                        RTN      382
C        OTHERWISE, WE HAVE COMPLETED THIS DATA POINT. CORRECT THE       RTN      383
C          SIMPSON INTEGRATION AND OUTPUT THE RESULTS                    RTN      384
C                                                                        RTN      385
      PDELT = -PKEP1                                                     RTN      386
      IF(IEVEN.GT.0) PDELT = (PKEP2 - 5.*PKEP1)/2.                       RTN      387
      SCRIPTP = SCRIPTP + PDELT                                          RTN      388
      SCRIPTP = SCRIPTP/3.                                               RTN      389
      NPTS=NPTS+1                                                        RTN      390
      P = SCRIPTP*.415./(2.*PI)                                          RTN      391
      WRITE(12,70) NPTS,TAU,OTIME,P                                      RTN      392
      SCRIP1 = 0.                                                        RTN      393
      PKEP1 = 0.                                                         RTN      394
      PKEP2 = 0.                                                         RTN      395
      SCRIPTP=0.0                                                        RTN      396
C                                                                        RTN      397
C        INCREASE THE TIME AND BEGIN A NEW DATA POINT                    RTN      398
C                                                                        RTN      399
   40 TAU = TAU + DTAUM*DELTAU                                           RTN      400
      GO TO 20                                                           RTN      401
C                                                                        RTN      402
C        WE ARE IN THE ROTOR SYSTEM. SWEEP THE ARC                       RTN      403
```

```
C                                                                  RTN     404
   50 CALL CALCSP(RSQ,POSQ,PHI,Y,SCRIP1)                           RTN     405
      GO TO 30                                                     RTN     406
C                                                                  RTN     407
C        CHECK FOR ADDITIONAL DATA                                 RTN     408
C                                                                  RTN     409
   60 NPTS = 0                                                     RTN     410
      WRITE(12,70) NPTS                                            RTN     411
      IF(MORDATA) GO TO 10                                         RTN     412
C                                                                  RTN     413
C        TERMINATION.                                              RTN     414
C                                                                  RTN     415
   70 FORMAT(I5,3E25.15)                                           RTN     416
      END                                                          RTN     417
```

Subroutine NEWYH

This routine calculates YH, the three coordinates of the center of the rotor system relative to its initial location. This routine is called only for moving (translational) rotor systems.

Flow Chart of Subroutine NEWYH



APPENDIX A

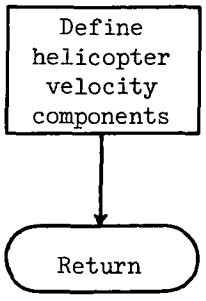
Listing of Subroutine NEWYH

```
      SUBROUTINE NEWYH(TIME)                                                NEWYH      2
C*****************************************************************************NEWYH      3
C*                                                                            *NEWYH      4
C*       SUBROUTINE NEWYH IS CALLED BY PROGRAM RTN AND SUBROUTINE             *NEWYH      5
C*         CALCYST. THIS ROUTINE RETURNS YH(3), THE THREE                     *NEWYH      6
C*         COORDINATES OF THE CENTER OF THE ROTOR SYSTEM RELATIVE             *NEWYH      7
C*         TO ITS INITIAL LOCATION (0,0,0). SUBROUTINE NEWYH IS               *NEWYH      8
C*         CALLED ONLY FOR MOVING (TRANSLATIONAL) ROTOR SYSTEMS TO            *NEWYH      9
C*         DETERMINE THE POSITION AT THE TIME SPECIFIED BY THE                *NEWYH     10
C*         VARIABLE TIME (M/SEC). THE ROTOR SYSTEM VELOCITY (POSSIBLY         *NEWYH     11
C*         TIME-DEPENDENT) IS OBTAINED FROM SUBROUTINE CALCVH.                *NEWYH     12
C*                                                                            *NEWYH     13
C*****************************************************************************NEWYH     14
      DIMENSION     BETA(8),        COEFFS(5),      ETA(2,8),                 NEWYH     15
     1              X(3),           X0(3),                                    NEWYH     16
     2              YEND(2),        YH(3),          YSTART(2),                NEWYH     17
     3              VH(3)                                                     NEWYH     18
      COMMON         ALFA,     BETA,     BLADES,   BLNTN,    CBETA,          NEWYH     19
     1               CH,       COEFFS,   DELTAU,   ETA,      ETA2,           NEWYH     20
     2               ETAMAX,   FAC,      FN,       FNS,      FETA1,          NEWYH     21
     3               IFLAG,    IREGION,  J2,       JFLG,     KFLG,           NEWYH     22
     4               NBLADES,  OMEGAR,   OTIME,    PI,       R,              NEWYH     23
     5               RM,       RMIN,     S,        SBETA,    SNDSPD,         NEWYH     24
     6               SRCDST,   SWPDST,   SWPTAU,   TAU,      THKRAT,         NEWYH     25
     7               TIMTHRU,  VH,       X,        X0,       YEND,           NEWYH     26
     8               YH,       YSTART,   DELETA1,  DELF                      NEWYH     27
      CALL CALCVH(TIME)                                                       NEWYH     28
      DO 10 I=1,3                                                             NEWYH     29
      YH(I) = VH(I)*TIME                                                      NEWYH     30
   10 CONTINUE                                                                NEWYH     31
      RETURN                                                                  NEWYH     32
      END                                                                     NEWYH     33
```

Subroutine CALCVH

CALCVH returns the three components of the translational velocity of the rotor system. In general, this subroutine is a user-supplied subroutine which must be provided for any case other than for that of a hovering helicopter. The routine included with the current version of RTN provides for a constant velocity of 61.73 m/sec (120 knots) directed along the Y_1-axis.

Flow Chart of Subroutine CALCVH



APPENDIX A

Listing of Subroutine CALCVH

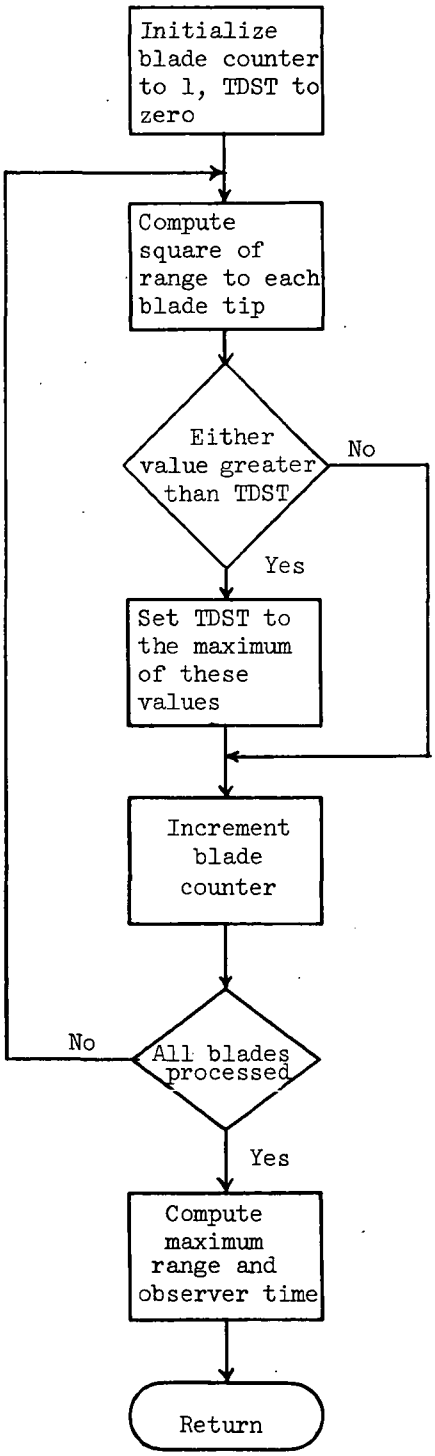
```
      SUBROUTINE CALCVH(TIME)                                                  CALCVH     2
C*************************************************************************** CALCVH     3
C*                                                                          * CALCVH     4
C*       SUBROUTINE CALCVH IS CALLED BY SUBROUTINE NEWVH. CALCVH            * CALCVH     5
C*         RETURNS VH(3), THE THREE COMPONENTS OF THE VELOCITY              * CALCVH     6
C*         (TRANSLATIONAL) OF THE ROTOR SYSTEM. THE VARIABLE TIME           * CALCVH     7
C*         MAY BE USED TO SPECIFY A TIME-DEPENDENT VELOCITY.                * CALCVH     8
C*                                                                          * CALCVH     9
C*************************************************************************** CALCVH    10
      DIMENSION    BETA(8),       COEFFS(5),     ETA(2,8),                   CALCVH    11
     1             X(3),          XO(3),                                     CALCVH    12
     2             YEND(2),       YH(3),         YSTART(2),                  CALCVH    13
     3             VH(3)                                                     CALCVH    14
      COMMON        ALFA,     BETA,     BLADES,   BLNTN,    CBETA,          CALCVH    15
     1              CH,       COEFFS,   DELTAU,   ETA,      ETA2,           CALCVH    16
     2              ETAMAX,   FAC,      FN,       FNS,      FETA1,          CALCVH    17
     3              IFLAG,    IREGION,  J2,       JFLG,     KFLG,           CALCVH    18
     4              NBLADES,  OMEGAR,   OTIME,    PI,       R,              CALCVH    19
     5              RM,       RMIN,     S,        SBETA,    SNDSPD,         CALCVH    20
     6              SRCDST,   SWPOST,   SWPTAU,   TAU,      THKRAT,         CALCVH    21
     7              TIMTHRU,  VH,       X,        XO,       YEND,           CALCVH    22
     8              YH,       YSTART,   DELETA1,  DELF                      CALCVH    23
      VH(1) = 61.73                                                          CALCVH    24
      VH(2)=0.0                                                              CALCVH    25
      VH(3)=0.0                                                              CALCVH    26
      RETURN                                                                 CALCVH    27
      END                                                                    CALCVH    28
```

Subroutine RMOTIME

This subroutine determines the range from the observer to the most distant blade tip in the rotor system. This range (RM) is the initial radius of the contracting sphere. Corresponding to this range and the current value of source time, the value of the observer time (OTIME) is also computed.

APPENDIX A

Flow Chart of Subroutine RMOTIME



APPENDIX A

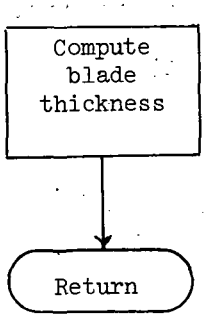
Listing of Subroutine RMOTIME

```
      SUBROUTINE RMOTIME                                                    RMOTIME     2
C*************************************************************************** RMOTIME     3
C*                                                                          * RMOTIME     4
C*       SUBROUTINE RMOTIME IS CALLED BY PROGRAM RTN. THIS RCUTINE          * RMOTIME     5
C*         DETERMINES THE RANGE TO THE MOST DISTANT TIP IN THE POTOR        * RMOTIME     6
C*         SYSTEM, WHICH IS USED AS THE MAXIMUM SWEEPING DISTANCE           * RMOTIME     7
C*         FOR A GIVEN TAU. THE OBSERVERS TIME (CORRESPONDING TO TAU)       * RMOTIME     8
C*         IS ALSO RETURNED.                                                * RMOTIME     9
C*                                                                          * RMOTIME    10
C*************************************************************************** RMOTIME    11
      DIMENSION     BETA(8),        COEFFS(5),      ETA(2,8),               RMOTIME    12
     1              X(3),           XG(3),                                  RMOTIME    13
     2              YEND(2),        YH(3),          YSTART(2),              RMOTIME    14
     3              VH(3)                                                   RMOTIME    15
      COMMON         ALFA,     BETA,     BLADES,   BLNTN,    CBETA,        RMOTIME    16
     1               CH,       COEFFS,   DELTAU,   ETA,      ETA2,         RMOTIME    17
     2               ETAMAX,   FAC,      FN,       FNS,      FETA1,        RMOTIME    18
     3               IFLAG,    IREGION,  J2,       JFLG,     KFLG,         RMOTIME    19
     4               NBLADES,  OMEGAR,   OTIME,    PI,       R,            RMOTIME    20
     5               RM,       RMIN,     S,        SBETA,    SNDSPD,       RMOTIME    21
     6               SRCOST,   SWPOST,   SWPTAU,   TAU,      THKRAT,       RMOTIME    22
     7               TIMTHRU,  VH,       X,        XG,       YEND,         RMOTIME    23
     8               YH,       YSTART,   DELETA1,  DELF                    RMOTIME    24
      DIST(A,B,C,D,E) = (A - D*COS(E))**2 + (B - D*SIN(E))**2 + C*C         RMOTIME    25
      TOST = 0.                                                             RMOTIME    26
      XBETA = OMEGAR*TAU                                                    RMOTIME    27
      DO 10 I=1,NBLADES                                                     RMOTIME    28
      TIP1 = XBETA + PI*(1. + 2.*FLOAT(I-1)/BLADES) - ALFA                  RMOTIME    29
      TIP2 = TIP1 + 2.*ALFA                                                 RMOTIME    30
      DI = DIST(X(1),X(2),X(3),R,TIP1)                                      RMOTIME    31
      IF(DI.GE.TOST) TOST = DI                                              RMOTIME    32
      DI = DIST(X(1),X(2),X(3),R,TIP2)                                      RMOTIME    33
      IF(DI.GE.TOST) TOST = DI                                              RMOTIME    34
   10 CONTINUE                                                              RMOTIME    35
      RM = SQRT(TOST)                                                       RMOTIME    36
      OTIME = TAU + RM/SNDSPD                                               RMOTIME    37
      RETURN                                                                RMOTIME    38
      END                                                                   RMOTIME    39
```

Function FOFETA1

This subprogram computes the blade thickness for specified distance along the blade chord. The function included with the current version of RTN is limited to uniform airfoil section across the span.

Flow Chart of Function FOFETAL



APPENDIX A

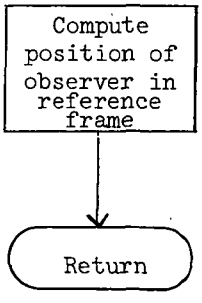
Listing of Function FOFETA1

```
      FUNCTION FOFETA1(XETA)                                                  FOFETA1     2
C*************************************************************************  FOFETA1     3
C*                                                                       *  FOFETA1     4
C*       FUNCTION FOFETA1 COMPUTES THE BLADE THICKNESS FOR SPECIFIED     *  FOFETA1     5
C*         DISTANCE ALONG THE BLADE CHORD                                *  FOFETA1     6
C*                                                                       *  FOFETA1     7
C*************************************************************************  FOFETA1     8
      DIMENSION     BETA(8),        COEFFS(5),      ETA(2,8),               FOFETA1     9
     1              X(3),           XO(3),                                  FOFETA1    10
     2              YEND(2),        YH(3),          YSTART(2),              FOFETA1    11
     3              VH(3)                                                   FOFETA1    12
      COMMON         ALFA,     BETA,     BLADES,   BLNTN,    CBETA,        FOFETA1    13
     1               CH,       COEFFS,   DELTAU,   ETA,      ETA2,         FOFETA1    14
     2               ETAMAX,   FAC,      FN,       FNS,      FETA1,        FOFETA1    15
     3               IFLAG,    IREGION,  J2,       JFLG,     KFLG,         FOFETA1    16
     4               NBLADES,  OMEGAR,   OTIME,    PI,       R,            FOFETA1    17
     5               RM,       RMIN,     S,        SBETA,    SNDSPD,       FOFETA1    18
     6               SRCOST,   SWPOST,   SWPTAU,   TAU,      THKRAT,       FOFETA1    19
     7               TIMTHRU,  VH,       X,        XO,       YEND,         FOFETA1    20
     8               YH,       YSTART,   DELETA1,  DELF                    FOFETA1    21
      E = XETA/CH                                                           FOFETA1    22
      A = COEFFS(1)*SQRT(E)                                                 FOFETA1    23
      A1 = COEFFS(5)*E + COEFFS(4)                                          FOFETA1    24
      A1 = A1*E + COEFFS(3)                                                 FOFETA1    25
      A1 = (A1*E + COEFFS(2))*E                                             FOFETA1    26
      FOFETA1 = (A + A1)*THKRAT                                             FOFETA1    27
   10 RETURN                                                                FOFETA1    28
      END                                                                   FOFETA1    29
```

Subroutine OBSPOS

This subroutine returns X, the three coordinates of the observer relative to the YH-frame fixed to the helicopter.

Flow Chart of Subroutine OBSPOS



APPENDIX A

Listing of Subroutine OBSPOS

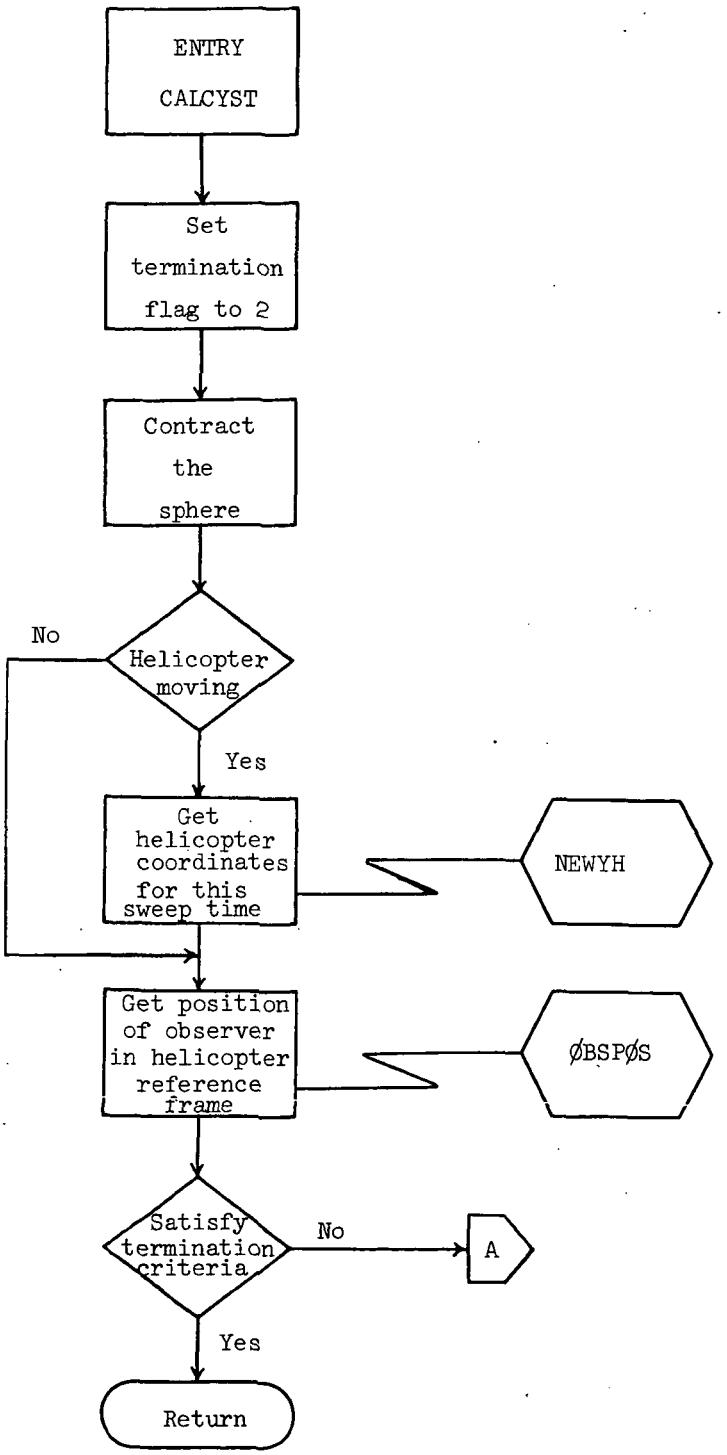
```
      SUBROUTINE OBSPOS                                                  OBSPOS     2
C*************************************************************************OBSPOS     3
C*                                                                       *OBSPOS     4
C*       SUBROUTINE OBSPOS IS CALLED BY PROGRAM RTN AND BY SUBROUTINE    *OBSPOS     5
C*         CALCYST. THIS ROUTINE RETURNS X(3),  THE THREE                *OBSPOS     6
C*         COORDINATES OF THE OBSERVER IN THE CURRENT ROTOR SYSTEM        *OBSPOS     7
C*         REFERENCE FRAME.                                              *OBSPOS     8
C*                                                                       *OBSPOS     9
C*************************************************************************OBSPOS    10
      DIMENSION    BETA(8),       COEFFS(5),     ETA(2,8),               OBSPOS    11
     1             X(3),          XQ(3),                                 OBSPOS    12
     2             YEND(2),       YH(3),         YSTART(2),              OBSPOS    13
     3             VH(3)                                                 OBSPOS    14
      COMMON        ALFA,     BETA,     BLADES,   BLNTN,    CBETA,      OBSPOS    15
     1              CH,       COEFFS,   DELTAU,   ETA,      ETA2,       OBSPOS    16
     2              ETAMAX,   FAC,      FN,       FNS,      FETA1,      OBSPOS    17
     3              IFLAG,    IREGION,  J2,       JFLG,     KFLG,       OBSPOS    18
     4              NBLADES,  OMEGAR,   OTIME,    PI,       R,          OBSPOS    19
     5              RM,       RMIN,     S,        SBETA,    SNOSPD,     OBSPOS    20
     6              SRCOST,   SWPOST,   SWPTAU,   TAU,      THKRAT,     OBSPOS    21
     7              TIMTHRU,  VH,       X,        XQ,       YEND,       OBSPOS    22
     8              YH,       YSTART,   DELETA1,  DELF                  OBSPOS    23
      X(1)=XQ(1)-YH(1)                                                   OBSPOS    24
      X(2)=XQ(2)-YH(2)                                                   OBSPOS    25
      X(3)=XQ(3)-YH(3)                                                   OBSPOS    26
      RETURN                                                             OBSPOS    27
      END                                                                OBSPOS    28
```

Subroutine CALCYST

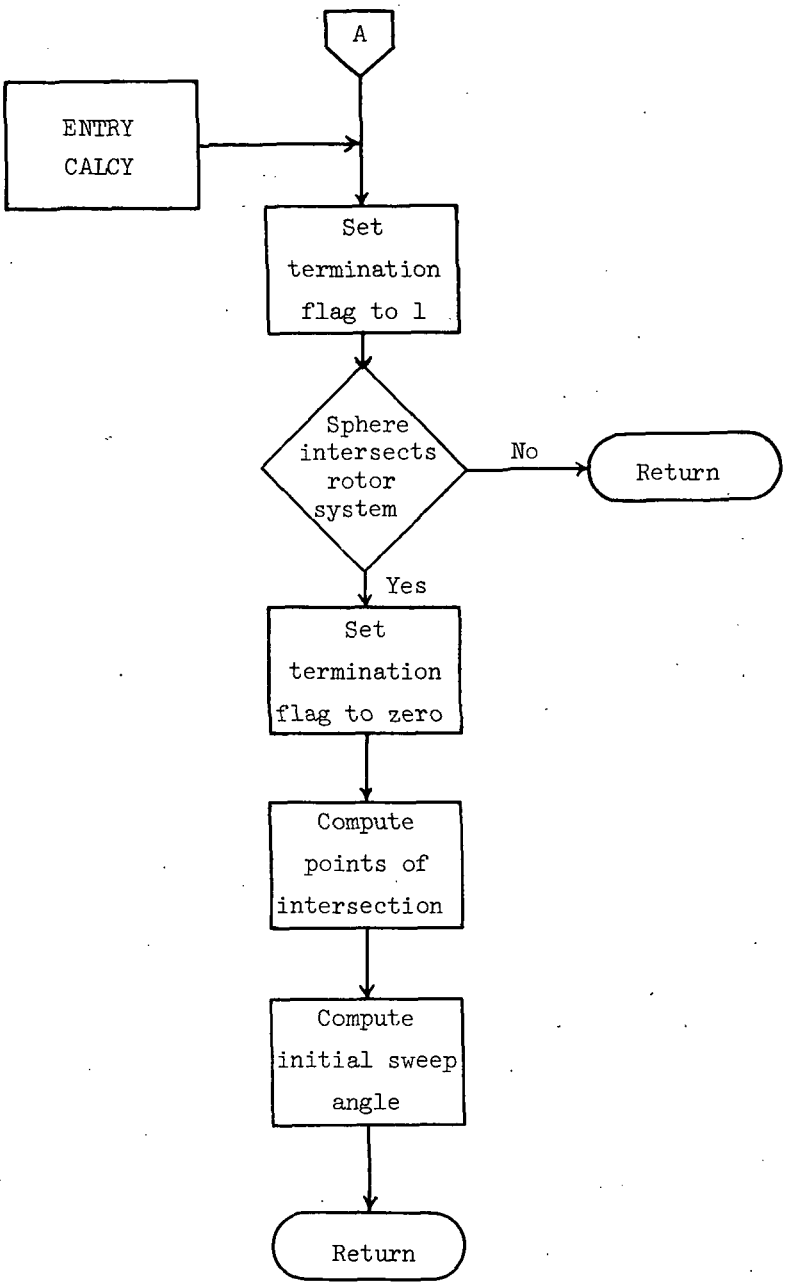
Subroutine CALCYST is called by RTN. A second entry point, CALCY, is used by subroutine CALCSP. This subprogram calculates the points of intersection of the projection of the sphere on the rotor system plane with the circles(s) defined by the rotor system. The calculation is made for either the circle defined by the outer blade radius or for that defined by the hub radius. The two components of the more clockwise point of intersection (relative to the observer) are returned in YSTART. The more counterclockwise point is returned in YEND. The angle (at the observer's position) defined by YSTART is returned in PHI0. The CALCYST entry contracts the sphere. The CALCY entry does not.

APPENDIX A

Flow Chart of Subroutine CALCYST



APPENDIX A


A
ENTRY
CALCY
Set
termination
flag to 1
Sphere
intersects
rotor
system
No
Return
Yes
Set
termination
flag to zero
Compute
points of
intersection
Compute
initial sweep
angle
Return

APPENDIX A

Listing of Subroutine CALCYST

```
      SUBROUTINE CALCYST(RSQ,PHIO)                                      CALCYST    2
C*********************************************************************** CALCYST    3
C*                                                                     * CALCYST    4
C*       SUBROUTINE CALCYST IS CALLED BY PROGRAM RTN. A SECOND ENTRY   * CALCYST    5
C*         POINT, CALCY, IS USED BY SUBROUTINE CALCSP. THIS ROUTINE    * CALCYST    6
C*         CALCULATES THE POINTS OF INTERSECTION OF THE PROJECTION     * CALCYST    7
C*         OF THE CONTRACTING SPHERE ON THE ROTOR SYSTEM PLANE WITH     * CALCYST    8
C*         THE CIRCLE(S) DEFINED BY THE ROTOR SYSTEM. WHEN THE ROUTINE * CALCYST    9
C*         IS ENTERED WITH RSQ EQUAL TO THE OUTER BLADE RADIUS, THE    * CALCYST   10
C*         OUTER CIRCLE IS USED. WHEN ENTERED WITH RSQ EQUAL TO THE    * CALCYST   11
C*         INNER BLADE RADIUS, THE INNER CIRCLE IS USED. THE X AND Y   * CALCYST   12
C*         COMPONENTS OF THE POINT OF INTERSECTION MORE CLOCKWISE      * CALCYST   13
C*         (RELATIVE TO THE OBSERVER) ARE RETURNED IN YSTART(2). THE   * CALCYST   14
C*         MORE COUNTERCLOCKWISE POINT IS RETURNED IN YEND(2). THE     * CALCYST   15
C*         ANGLE (AT THE OBSERVERS POSITION) DEFINED BY YSTART IS      * CALCYST   16
C*         RETURNED IN PHIO. THE CALCYST ENTRY CONTRACTS THE SPHERE.   * CALCYST   17
C*         THE CALCY ENTRY DOES NOT.                                   * CALCYST   18
C*                                                                     * CALCYST   19
C*********************************************************************** CALCYST   20
      DIMENSION    BETA(8),       COEFFS(5),     ETA(2,8),              CALCYST   21
     1             X(3),          XO(3),                                CALCYST   22
     2             YEND(2),       YH(3),         YSTART(2),             CALCYST   23
     3             VH(3)                                                CALCYST   24
      COMMON        ALFA,     BETA,     BLADES,   BLNTN,    CBETA,      CALCYST   25
     1              CH,       COEFFS,   DELTAU,   ETA,      ETA2,       CALCYST   26
     2              ETAMAX,   FAC,      FN,       FNS,      FETA1,      CALCYST   27
     3              IFLAG,    IREGION,  J2,       JFLG,     KFLG,       CALCYST   28
     4              NBLADES,  OMEGAR,   OTIME,    PI,       R,          CALCYST   29
     5              RM,       RMIN,     S,        SBETA,    SNDSPD,     CALCYST   30
     6              SRCOST,   SWPOST,   SWPTAU,   TAU,      THKRAT,     CALCYST   31
     7              TIMTHRU,  VH,       X,        XC,       YEND,       CALCYST   32
     8              YH,       YSTART,   DELETA1,  DELF                  CALCYST   33
      IFLAG = 2                                                         CALCYST   34
      SWPTAU = SWPTAU + DELTAU                                          CALCYST   35
C                                                                       CALCYST   36
C        CONTRACT THE SPHERE                                            CALCYST   37
C                                                                       CALCYST   38
      SWPOST = RM - SNDSPD*TIMTHRU*DELTAU                               CALCYST   39
      IF(JFLG.NE.0) CALL NEWYH(SWPTAU)                                  CALCYST   40
      CALL OBSPOS                                                       CALCYST   41
      OBSANGL = ASIN(X(3)/SQRT(X(1)**2 + X(2)**2 + X(3)**2))            CALCYST   42
C                                                                       CALCYST   43
C        TEST TERMINATION CRITERIA                                      CALCYST   44
C                                                                       CALCYST   45
      IF(OBSANGL.GT.(PI/4.)) GO TO 10                                   CALCYST   46
      IF(RM.LT.RMIN) GO TO 10                                           CALCYST   47
      ENTRY CALCY                                                       CALCYST   48
      IFLAG = 1                                                         CALCYST   49
      SURDSQ = X(1)*X(1) + X(2)*X(2)                                    CALCYST   50
      OBSDSQ = SURDSQ + X(3)*X(3)                                       CALCYST   51
      DELTA = 0.5*(OBSDSQ + RSQ - SWPOST*SWPOST)                        CALCYST   52
      DISC = RSQ*SURDSQ - DELTA*DELTA                                   CALCYST   53
      IF(DISC.LE.0.) GO TO 10                                           CALCYST   54
      IFLAG = 0                                                         CALCYST   55
C                                                                       CALCYST   56
C        COMPUTE INTERSECTIONS AND INITIAL SWEEP ANGLE                  CALCYST   57
C                                                                       CALCYST   58
      FAC = 1.0                                                         CALCYST   59
      IF(X(1).LT.0.) FAC = -1.0                                         CALCYST   60
      IF(ABS(X(1)).LT.P) GO TO 5                                        CALCYST   61
      YSTART(2) = (X(2)*DELTA + X(1)*SQRT(DISC))/SURDSQ                 CALCYST   62
      YSTART(1) = (DELTA - X(2)*YSTART(2))/X(1)                         CALCYST   63
      YEND(2) = (X(2)*DELTA - X(1)*SQRT(DISC))/SURDSQ                   CALCYST   64
      YEND(1) = (DELTA - X(2)*YEND(2))/X(1)                             CALCYST   65
      GO TO 7                                                           CALCYST   66
    5 YSTART(1) = (X(1)*DELTA - FAC*X(2)*DISC)/SURDSQ                   CALCYST   67
      YSTART(2) = (DELTA - X(1)*YSTART(1))/X(2)                         CALCYST   68
      YEND(1) = (X(1)*DELTA + FAC*X(2)*DISC)/SURDSQ                     CALCYST   69
      YEND(2) = (DELTA - X(1)*YEND(1))/X(2)                             CALCYST   70
    7 SPCOST = SQRT(SWPOST**2 - X(3)**2)                                CALCYST   71
      PHIO = ACOS(0.9999999999*(YSTART(1) - X(1))/SPCOST)               CALCYST   72
      IF(YSTART(2).LT.X(2)) PHIO = 2.*PI - PHIO                         CALCYST   73
   10 RETURN                                                            CALCYST   74
      END                                                               CALCYST   75
```

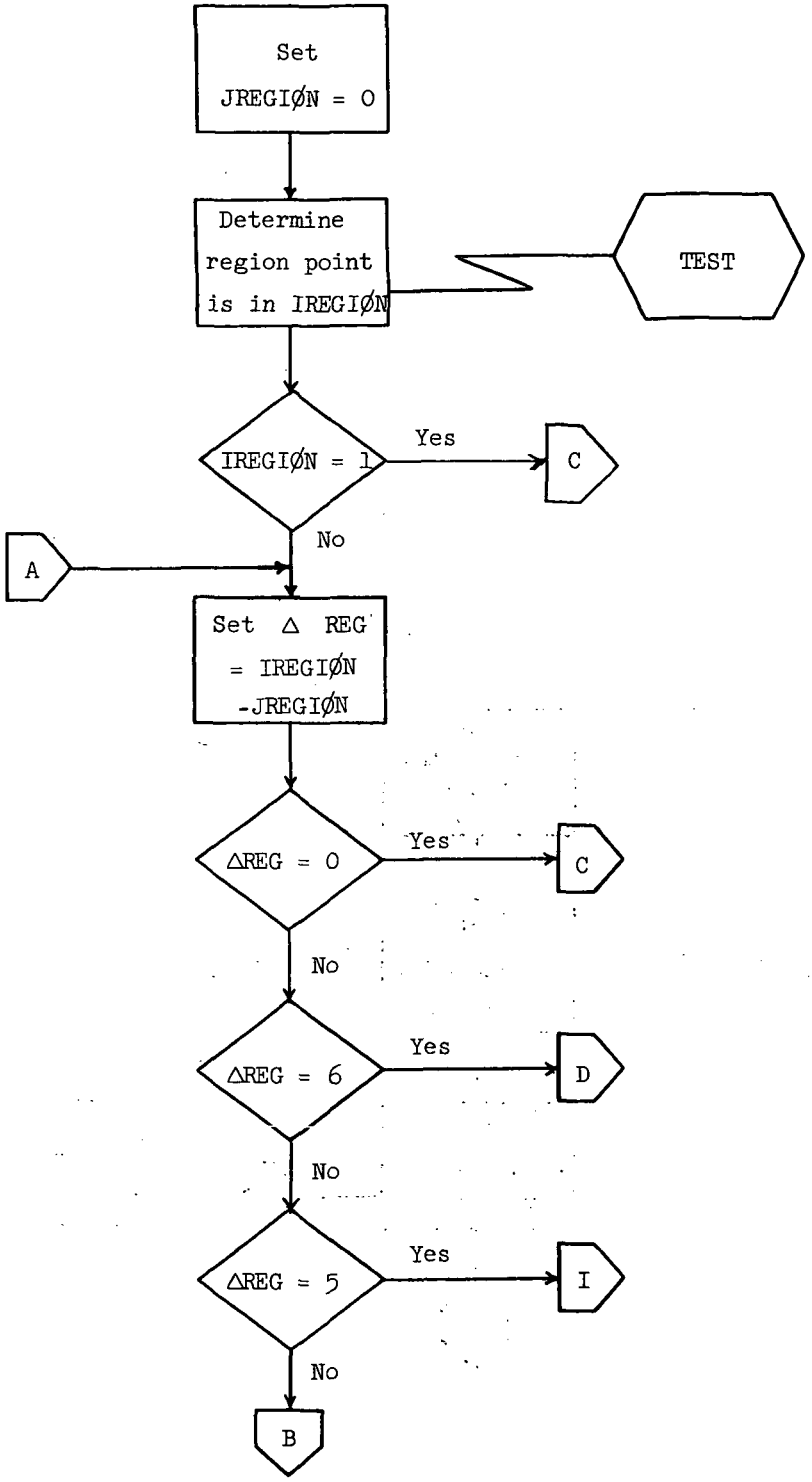
APPENDIX A

Subroutine CALCSP

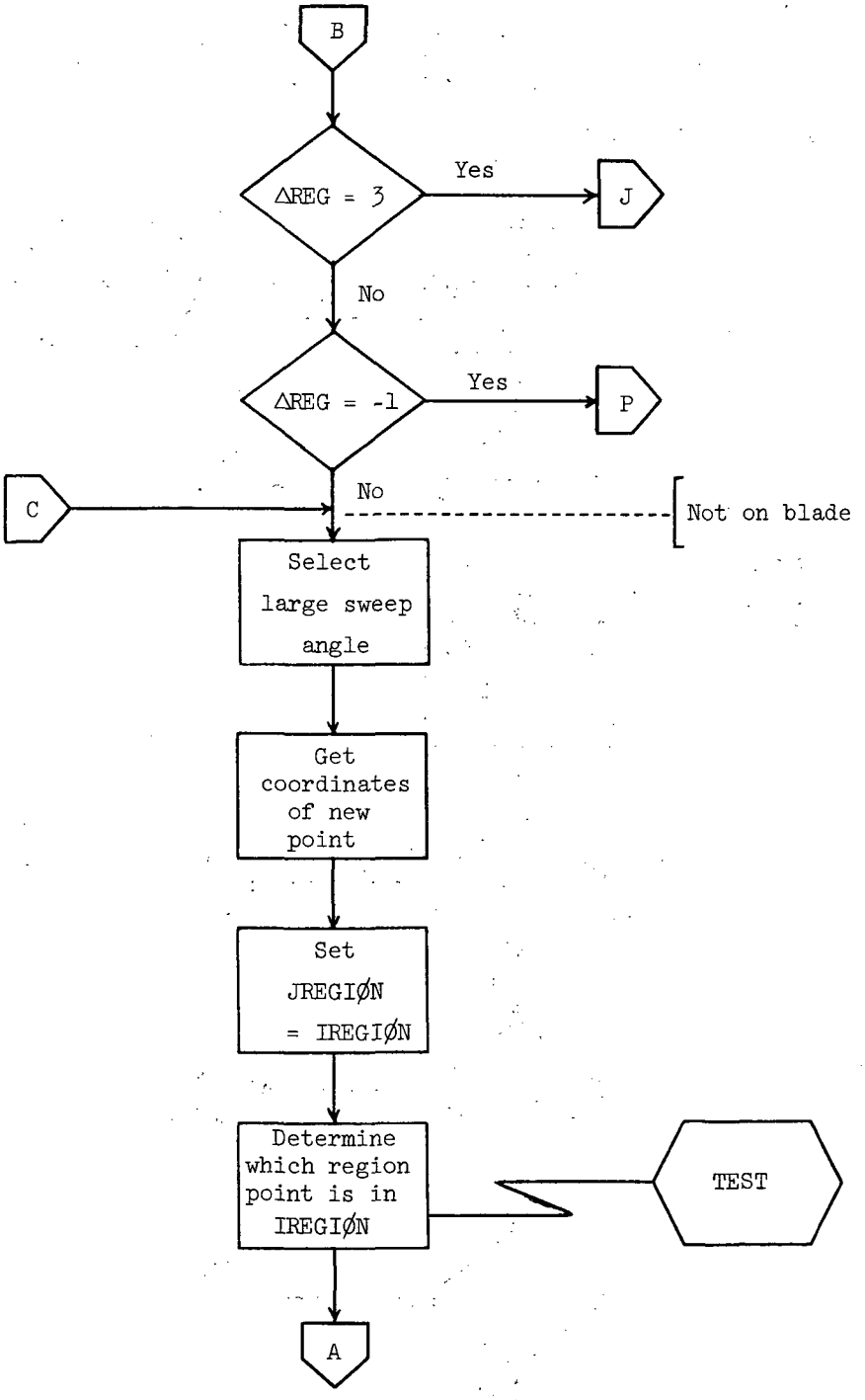
This routine controls the sweeping across the arc defined by a fixed position of the contracting sphere. The sweep is in a counterclockwise direction beginning at PHI0 and terminating when a point is reached outside the circle defined by the rotor system. A trapezoidal integration over the intersection of the arc with each blade is performed dynamically. The coarseness of the sweep increment is determined as a function of whether or not a point is on a blade. Even finer sweep increments are used near the leading edge of blades with blunt leading edges.

APPENDIX A

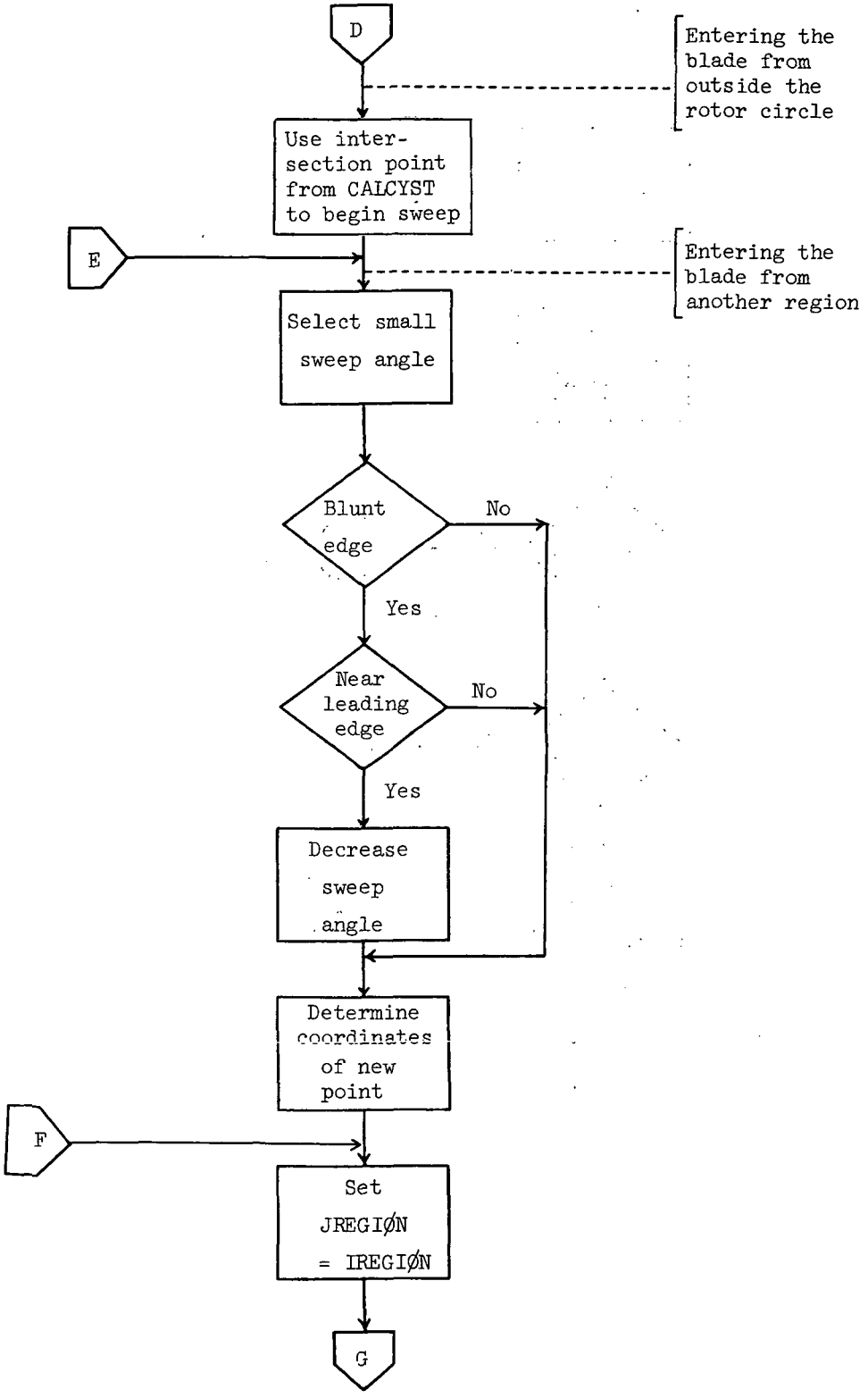
Flow Chart of Subroutine CALCSP



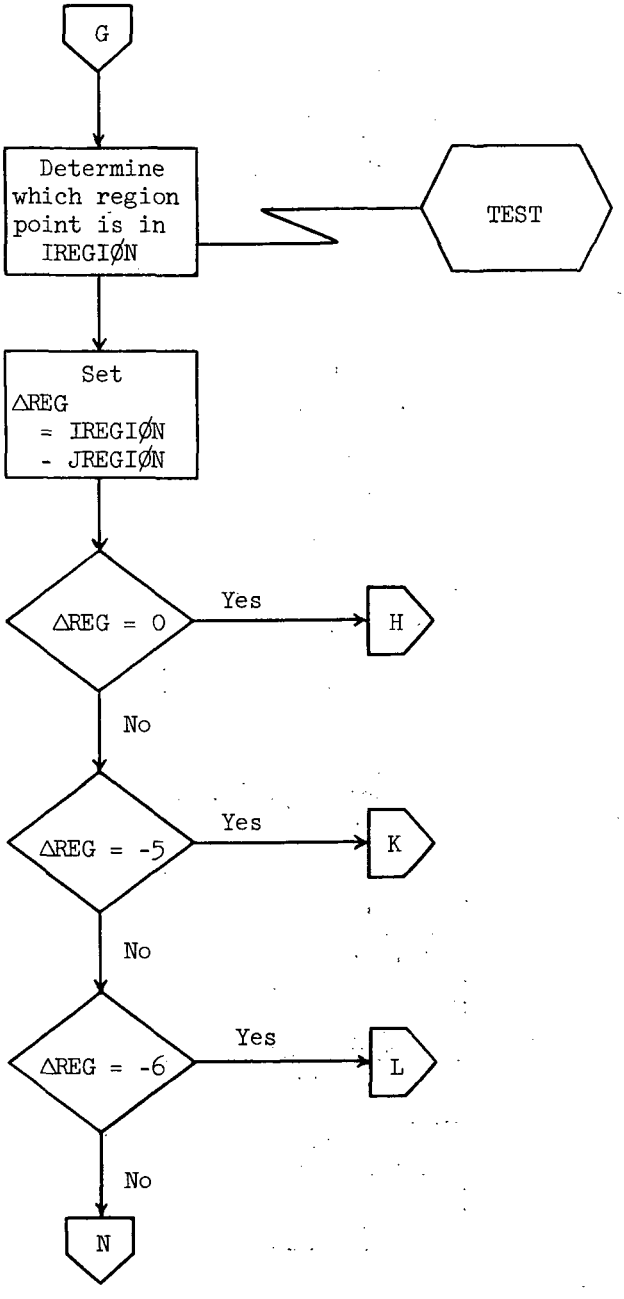
APPENDIX A


B
ΔREG = 3
Yes
J
No
ΔREG = -1
Yes
P
C
No
Not on blade
Select large sweep angle
Get coordinates of new point
Set JREGIØN = IREGIØN
Determine which region point is in IREGIØN
TEST
A

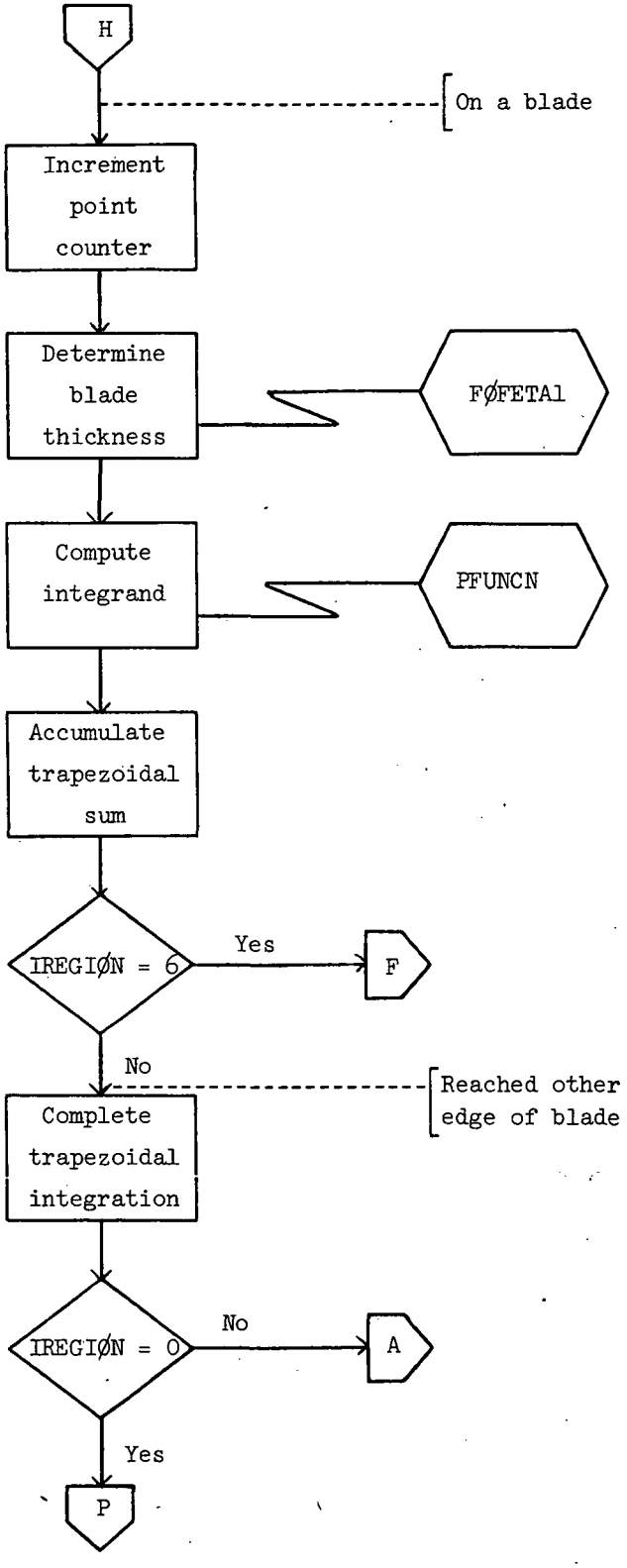
APPENDIX A


D
Entering the blade from outside the rotor circle
Use inter-section point from CALCYST to begin sweep
E
Entering the blade from another region
Select small sweep angle
Blunt edge
No
Yes
Near leading edge
No
Yes
Decrease sweep angle
Determine coordinates of new point
F
Set JREGIØN = IREGIØN
G

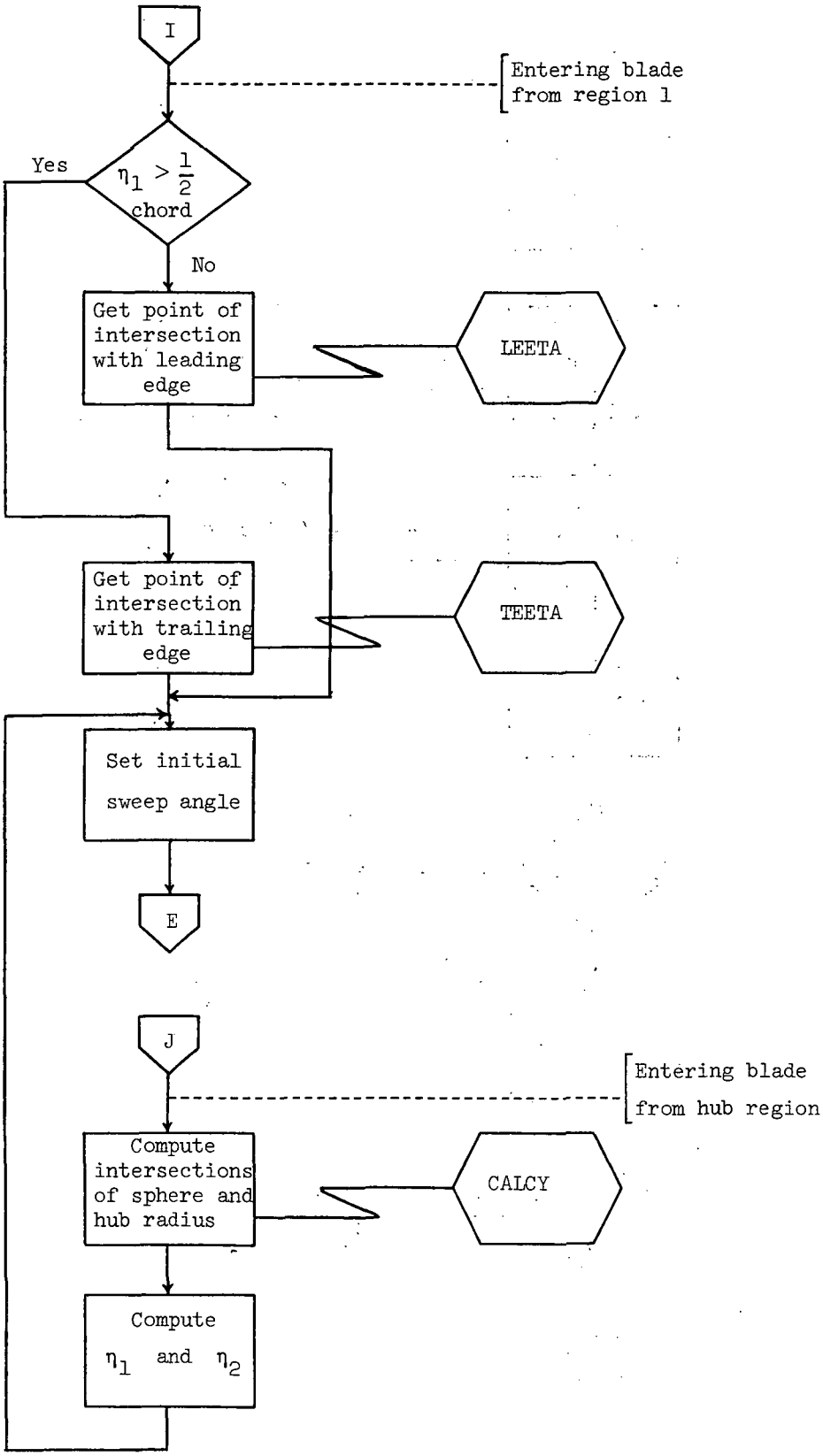
APPENDIX A


G
Determine which region point is in IREGIØN
TEST
Set ΔREG = IREGIØN - JREGIØN
ΔREG = 0
Yes
H
No
ΔREG = -5
Yes
K
No
ΔREG = -6
Yes
L
No
N

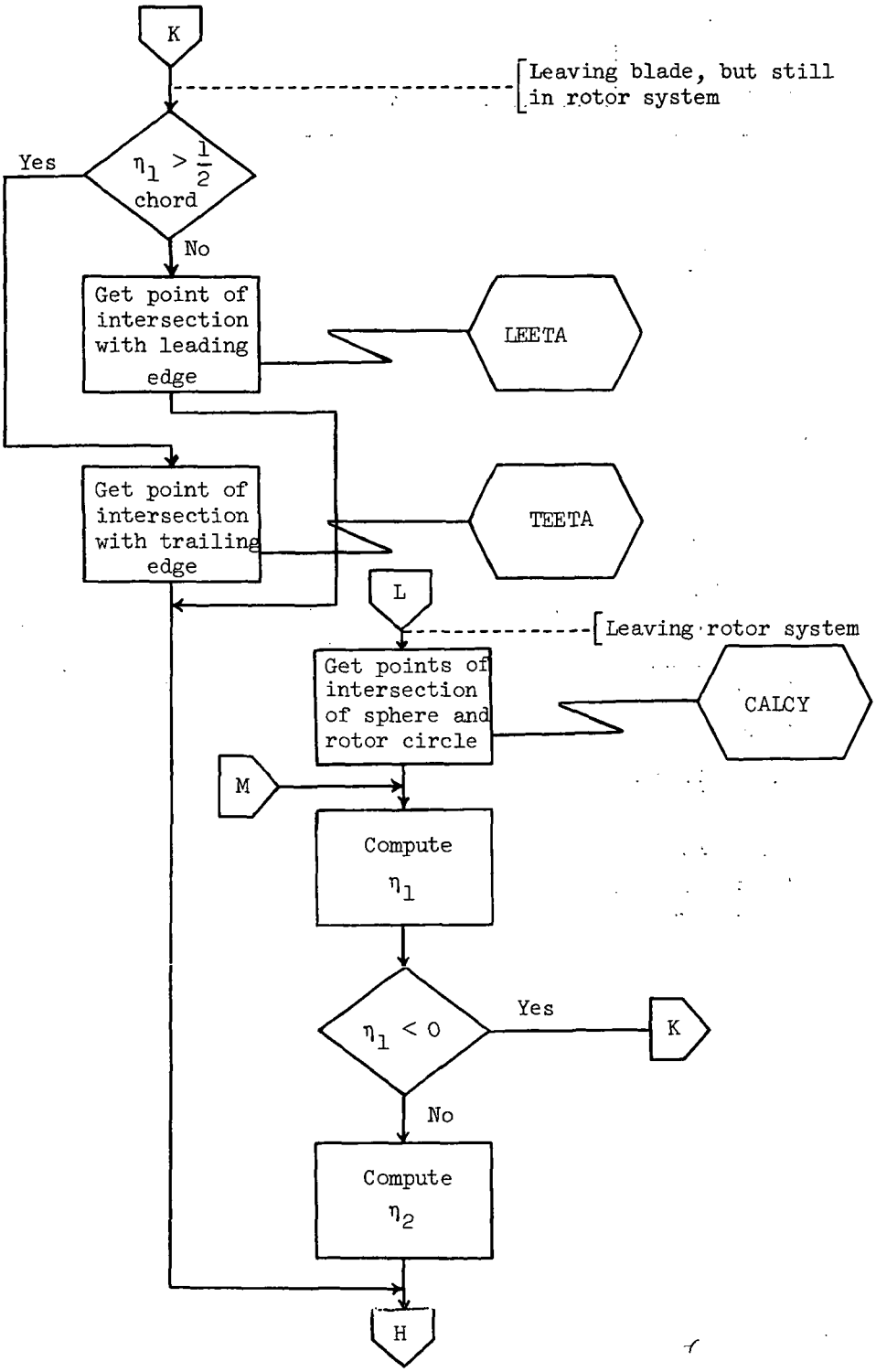
APPENDIX A


H
On a blade
Increment point counter
Determine blade thickness
FØFETA1
Compute integrand
PFUNCN
Accumulate trapezoidal sum
IREGIØN = 6
Yes
F
No
Reached other edge of blade
Complete trapezoidal integration
IREGIØN = 0
No
A
Yes
P

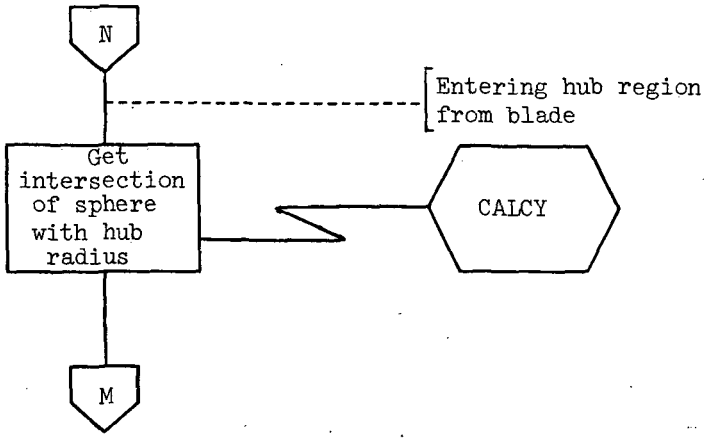
APPENDIX A

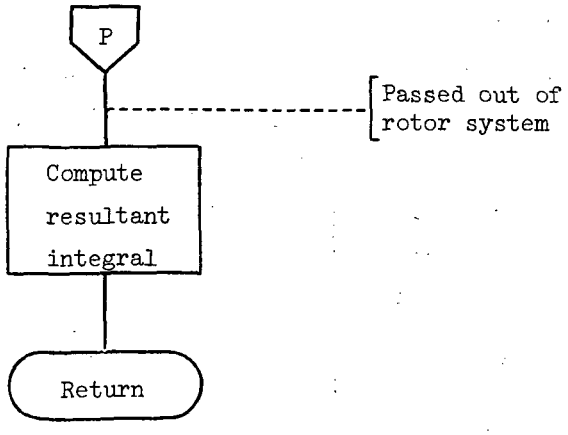

I
Entering blade from region 1
Yes
$\eta_1 > \frac{1}{2}$ chord
No
Get point of intersection with leading edge
LEETA
Get point of intersection with trailing edge
TEETA
Set initial sweep angle
E
J
Entering blade from hub region
Compute intersections of sphere and hub radius
CALCY
Compute η_1 and η_2

APPENDIX A


K
Leaving blade, but still in rotor system
Yes
$\eta_1 > \frac{1}{2}$ chord
No
Get point of intersection with leading edge
LEETA
Get point of intersection with trailing edge
TEETA
L
Leaving rotor system
Get points of intersection of sphere and rotor circle
CALCY
M
Compute η_1
$\eta_1 < 0$
Yes
K
No
Compute η_2
H

APPENDIX A


N
Entering hub region from blade
Get intersection of sphere with hub radius
CALCY
M


P
Passed out of rotor system
Compute resultant integral
Return

APPENDIX A

Listing of Subroutine CALCSP

```
      SUBROUTINE CALCSP(RSQ,ROSQ,PHI,Y,SCRIPTP)                          CALCSP     2
C*************************************************************************CALCSP     3
C*                                                                       *CALCSP     4
C*       SUBROUTINE CALCSP IS CALLED BY PROGRAM RTN. THIS ROUTINE        *CALCSP     5
C*         CONTROLS THE SWEEPING (COUNTERCLOCKWISE) ACROSS THE ARC       *CALCSP     6
C*         DEFINED BY A FIXED POSITION OF THE CONTRACTING SPHERE.        *CALCSP     7
C*         TWO SWEEPING INCREMENTS ARE USED: DELPHI2 FOR SWEEPING        *CALCSP     8
C*         ACROSS A ROTOR BLADE AND DELPHI1 FOR THE SPACE BETWEEN        *CALCSP     9
C*         BLADES, AS DETERMINED BY SUBROUTINE TEST. THE LINE INTEGRAL   *CALCSP    10
C*         REPRESENTING THE CONTRIBUTION OF EACH BLADE IS A              *CALCSP    11
C*         TRAPEZOIDAL INTEGRATION. POINTS ON THE EDGE OF ANY BLADE      *CALCSP    12
C*         ARE OBTAINED FROM SUBROUTINES LEFTA AND TEETA.                *CALCSP    13
C*                                                                       *CALCSP    14
C*************************************************************************CALCSP    15
      DIMENSION    BETA(8),        COEFFS(5),      ETA(2,8),             CALCSP    16
     1             X(3),           X0(3),          Y(3),                 CALCSP    17
     2             YEND(2),        YH(3),          YLST(3),              CALCSP    18
     3             YSTART(2),      VH(3)                                 CALCSP    19
      COMMON        ALFA,     BETA,     BLADES,   BLNTN,    CBETA,      CALCSP    20
     1              CH,       COEFFS,   DELTAU,   ETA,      ETA2,       CALCSP    21
     2              ETAMAX,   FAC,      FN,       FNS,      FETA1,      CALCSP    22
     3              IFLAG,    IREGION,  J2,       JFLG,     KFLG,       CALCSP    23
     4              NBLADES,  OMEGAR,   OTIME,    PI,       R,          CALCSP    24
     5              RM,       RMIN,     S,        SBETA,    SNDSPD,     CALCSP    25
     6              SRCOST,   SWPOST,   SWPTAU,   TAU,      THKRAT,     CALCSP    26
     7              TIMTHRU,  VH,       X,        X0,       YEND,       CALCSP    27
     8              YH,       YSTART,   DELETA1,  DELF                  CALCSP    28
C                                                                        CALCSP    29
C        INITIALIZE VARIABLES FOR THIS SWEEP                             CALCSP    30
C                                                                        CALCSP    31
      DELPHI1 = CH/(FN*SRCOST)                                           CALCSP    32
      DELPHI2 = CH/(FNS*SRCOST)                                          CALCSP    33
      DELETA1 = 0.                                                       CALCSP    34
      DELF = 0.                                                          CALCSP    35
      YLST(3) = 0.                                                       CALCSP    36
      JREGION = 0                                                        CALCSP    37
      SUMSP = 0.                                                         CALCSP    38
      OGLST = 0.                                                         CALCSP    39
      SCRIPTP = 0.                                                       CALCSP    40
      J = 1                                                              CALCSP    41
      J2 = 1                                                             CALCSP    42
C                                                                        CALCSP    43
C        DETERMINE WHICH REGION WE ARE IN                                CALCSP    44
C                                                                        CALCSP    45
      CALL TEST(YSTART,RSQ,ROSQ,SWPTAU)                                  CALCSP    46
      IF(IREGION.EQ.1) GO TO 20                                          CALCSP    47
   10 IDELREG = IREGION - JREGION                                        CALCSP    48
      IF(IDELREG.EQ.0) GO TO 20                                          CALCSP    49
      IF(IDELREG.EQ.6) GO TO 30                                          CALCSP    50
      IF(IDELREG.EQ.5) GO TO 90                                          CALCSP    51
      IF(IDELREG.EQ.3) GO TO 130                                         CALCSP    52
      IF(IDELREG.EQ.-1)GO TO 80                                          CALCSP    53
C                                                                        CALCSP    54
C        NOT ON A BLADE. SELECT LARGE ANGLE AND SEARCH THE ARC           CALCSP    55
C                                                                        CALCSP    56
   20 DELPHI = DELPHI1                                                   CALCSP    57
      PHI = PHI + FAC*DELPHI                                             CALCSP    58
      Y(1) = X(1) + SRCOST*COS(PHI)                                      CALCSP    59
      Y(2) = X(2) + SRCOST*SIN(PHI)                                      CALCSP    60
      JREGION = IREGION                                                  CALCSP    61
      CALL TEST(Y,RSQ,ROSQ,SWPTAU)                                       CALCSP    62
      GO TO 10                                                           CALCSP    63
C                                                                        CALCSP    64
C        ENTERING REGION 6 FROM REGION 0. USE THE INTERSECTION DATA      CALCSP    65
C                                                                        CALCSP    66
   30 Y(1) = YSTART(1)                                                   CALCSP    67
      Y(2) = YSTART(2)                                                   CALCSP    68
C                                                                        CALCSP    69
```

```
C        ENTERING REGION 6 FROM ANOTHER REGION. SELECT SMALL ANGLE      CALCSP    70
C          AND PROCESS THE BLADE                                        CALCSP    71
C                                                                       CALCSP    72
   40 DELPHI = DELPHI2                                                  CALCSP    73
      IF(ETA(1,J2).LT.ETAMAX) DELPHI = DELPHI/BLNTN                     CALCSP    74
      FETA1 = FOFETA1(ETA(1,J2))                                        CALCSP    75
   50 JREGION = IREGION                                                 CALCSP    76
C                                                                       CALCSP    77
C        INCREASE THE SWEEP ANGLE AND TEST THE NEXT POINT               CALCSP    78
C                                                                       CALCSP    79
      PHI = PHI + FAC*DELPHI                                            CALCSP    80
      ETA11 = ETA(1,J2)                                                 CALCSP    81
      ETA22 = ETA(2,J2)                                                 CALCSP    82
      FETA11 = FETA1                                                    CALCSP    83
      SAVETA = DELETA1                                                  CALCSP    84
      SAVEF = DELF                                                      CALCSP    85
      YLST(1) = Y(1)                                                    CALCSP    86
      YLST(2) = Y(2)                                                    CALCSP    87
      Y(1) = X(1) + SRCDST*COS(PHI)                                     CALCSP    88
      Y(2) = X(2) + SRCDST*SIN(PHI)                                     CALCSP    89
      CALL TEST(Y,RSQ,ROSQ,SWPTAU)                                      CALCSP    90
      IDELREG = IREGION - JREGION                                       CALCSP    91
      IF(IDELREG.EQ.0) GO TO 60                                         CALCSP    92
      IF(IDELREG.EQ.-5) GO TO 140                                       CALCSP    93
      IF(IDELREG.EQ.-6) GO TO 170                                       CALCSP    94
      GO TO 180                                                         CALCSP    95
C                                                                       CALCSP    96
C        ON A BLADE                                                     CALCSP    97
C                                                                       CALCSP    98
   60 J = J + 1                                                         CALCSP    99
      DELETA1 = ETA(1,J2) - ETA11                                       CALCSP   100
      DELETA2 = ETA(2,J2) - ETA22                                       CALCSP   101
      FETA1 = FOFETA1(ETA(1,J2))                                        CALCSP   102
      DELF = FETA1 - FETA11                                             CALCSP   103
      DELGAM = SQRT(DELF**2 + DELETA1**2 + DELETA2**2)                  CALCSP   104
      IF(DELETA1.LT.0.) DELGAM = - DELGAM                               CALCSP   105
      SP = PFUNCN(ETA22,DELF,DELETA1,YLST)                              CALCSP   106
      SUMSP = SUMSP + SP*DELGAM                                         CALCSP   107
      SP = PFUNCN(ETA22,SAVEF,SAVETA,YLST)                              CALCSP   108
      SUMSP = SUMSP + SP*DGLST                                          CALCSP   109
      DGLST = DELGAM                                                    CALCSP   110
      IF(IREGION.EQ.6) GO TO 50                                         CALCSP   111
C                                                                       CALCSP   112
C        JUST LEFT REGION 6. PERFORM THE TRAPEZOIDAL INTEGRATION        CALCSP   113
C                                                                       CALCSP   114
      SP = PFUNCN(ETA(2,J2),DELF,DELETA1,Y)                             CALCSP   115
      SUMSP = SUMSP + SP*DELGAM                                         CALCSP   116
      IF(IREGION.EQ.0) GO TO 80                                         CALCSP   117
      DGLST = 0.                                                        CALCSP   118
      DELETA1 = 0.                                                      CALCSP   119
      DELF = 0.                                                         CALCSP   120
      J = 1                                                             CALCSP   121
      J2 = 1                                                            CALCSP   122
      GO TO 20                                                          CALCSP   123
C                                                                       CALCSP   124
C        PASSED OUT OF THE ROTOR SYSTEM FOR THIS ARC                    CALCSP   125
C                                                                       CALCSP   126
   80 SCRIPTP = 0.5*DELTAU*SUMSP/SWPDST                                 CALCSP   127
      RETURN                                                            CALCSP   128
C                                                                       CALCSP   129
C        ENTERING REGION 6 FROM REGION 1                                CALCSP   130
C                                                                       CALCSP   131
   90 IF(ETA(1,J2).GT.CH/2.) GO TO 100                                  CALCSP   132
C                                                                       CALCSP   133
C        THROUGH THE LEADING EDGE                                       CALCSP   134
C                                                                       CALCSP   135
      CALL LEETA(ETA(2,J2),Y)                                           CALCSP   136
      ETA(1,J2) = 0.                                                    CALCSP   137
      GO TO 110                                                         CALCSP   138
C                                                                       CALCSP   139
C        THPOUGH THE TRAILING EDGE                                      CALCSP   140
```

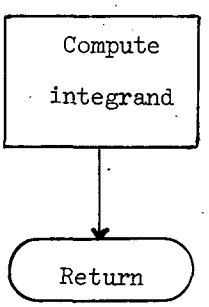
```
C                                                                       CALCSP   141
  100 CALL TEETA(ETA(2,J2),Y)                                           CALCSP   142
      ETA(1,J2) = CH                                                    CALCSP   143
  110 ETA(2,J2) = ETA2                                                  CALCSP   144
C                                                                       CALCSP   145
C        SET ANGLE AND RETURN TO BLADE LOGIC                            CALCSP   146
C                                                                       CALCSP   147
  120 PHI = ACOS(0.9999999999*(Y(1) - X(1))/SRCOST)                     CALCSP   148
      IF(Y(2).LT.X(2)) PHI = 2.*PI - PHI                                CALCSP   149
      GO TO 40                                                          CALCSP   150
C                                                                       CALCSP   151
C        ENTERING REGION 6 FROM REGION 3. START FROM INTERSECTION       CALCSP   152
C          WITH INNER RADIUS                                            CALCSP   153
C                                                                       CALCSP   154
  130 CALL CALCY(ROSQ,PHX)                                              CALCSP   155
      Y(1) = YEND(1)                                                    CALCSP   156
      Y(2) = YEND(2)                                                    CALCSP   157
      ETA(1,J2) = CH/2. - Y(1)*SBETA + Y(2)*CBETA                       CALCSP   158
      IF(ETA(1,J2).LT.0.) GO TO 90                                      CALCSP   159
      ETA(2,J2) = -Y(1)*CBETA - Y(2)*SBETA                              CALCSP   160
      GO TO 120                                                         CALCSP   161
C                                                                       CALCSP   162
C        ENTERING REGION 1 FROM REGION 6                                CALCSP   163
C                                                                       CALCSP   164
  140 IF(ETA(1,J2).GT.CH/2.) GO TO 150                                  CALCSP   165
C                                                                       CALCSP   166
C        THROUGH LEADING EDGE                                           CALCSP   167
C                                                                       CALCSP   168
      CALL LEETA(ETA(2,J2),Y)                                           CALCSP   169
      ETA(1,J2) = 0.                                                    CALCSP   170
      GO TO 160                                                         CALCSP   171
C                                                                       CALCSP   172
C        THROUGH TRAILING EDGE                                          CALCSP   173
C                                                                       CALCSP   174
  150 CALL TEETA(ETA(2,J2),Y)                                           CALCSP   175
      ETA(1,J2) = CH                                                    CALCSP   176
  160 ETA(2,J2) = ETA2                                                  CALCSP   177
      GO TO 60                                                          CALCSP   178
C                                                                       CALCSP   179
C        LEAVING THE ROTOR SYSTEM FOR THIS ARC. GET INTERSECTION WITH   CALCSP   180
C          OUTER RADIUS                                                 CALCSP   181
C                                                                       CALCSP   182
  170 CALL CALCY(RSQ,PHX)                                               CALCSP   183
      Y(1) = YEND(1)                                                    CALCSP   184
      Y(2) = YEND(2)                                                    CALCSP   185
      GO TO 190                                                         CALCSP   186
C                                                                       CALCSP   187
C        ENTERING REGION 3 FROM REGION 6. GET INTERSECTION WITH         CALCSP   188
C          INNER RADIUS                                                 CALCSP   189
C                                                                       CALCSP   190
  180 CALL CALCY(ROSQ,PHX)                                              CALCSP   191
      Y(1) = YSTART(1)                                                  CALCSP   192
      Y(2) = YSTART(2)                                                  CALCSP   193
  190 ETA(1,J2) = CH/2. - Y(1)*SBETA + Y(2)*CBETA                       CALCSP   194
      IF(ETA(1,J2).LT.0.) GO TO 140                                     CALCSP   195
      ETA(2,J2) = -Y(1)*CBETA - Y(2)*SBETA                              CALCSP   196
      GO TO 60                                                          CALCSP   197
      END                                                               CALCSP   198
```

APPENDIX A

Function PFUNCN

This function subprogram computes the integrand of the line integral at each sweep position.

Flow Chart of Function PFUNCN



Listing of Function PFUNCN

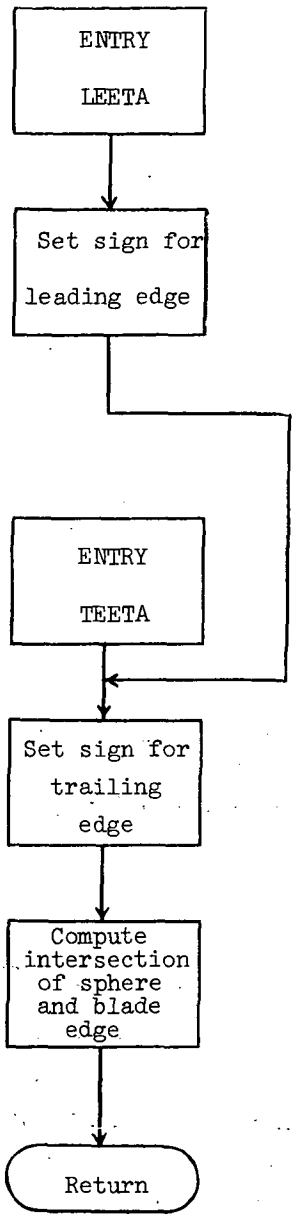
```
      FUNCTION PFUNCN(XETA,DELF,DELETA1,Y)                                  PFUNCN     2
C*************************************************************************  PFUNCN     3
C*                                                                       *  PFUNCN     4
C*       FUNCTION PFUNCN IS CALLED BY SUBROUTINE CALCSP. THIS FUNCTION   *  PFUNCN     5
C*         COMPUTES THE INTEGRAND OF THE LINE INTEGRAL AT EACH           *  PFUNCN     6
C*         SWEEP POSITION.                                               *  PFUNCN     7
C*                                                                       *  PFUNCN     8
C*************************************************************************  PFUNCN     9
      DIMENSION    BETA(8),        COEFFS(5),      ETA(2,8),               PFUNCN    10
     1             X(3),           XQ(3),                                  PFUNCN    11
     2             YEND(2),        YH(3),          YSTART(2),              PFUNCN    12
     3             VH(3)                                                   PFUNCN    13
      DIMENSION Y(3)                                                        PFUNCN    14
      COMMON        ALFA,      BETA,      BLADES,    BLNTN,     CBETA,     PFUNCN    15
     1              CH,        COEFFS,    DELTAU,    ETA,       ETA2,      PFUNCN    16
     2              ETAMAX,    FAC,       FN,        FNS,       FETA1,     PFUNCN    17
     3              IFLAG,     IREGION,   J2,        JFLG,      KFLG,      PFUNCN    18
     4              NBLADES,   OMEGAR,    OTIME,     PI,        R,         PFUNCN    19
     5              RM,        RMIN,      S,         SBETA,     SNDSPD,    PFUNCN    20
     6              SPCOST,    SWPOST,    SWPTAU,    TAU,       THKRAT,    PFUNCN    21
     7              TIMTHRU,   VH,        X,         XC,        YEND,      PFUNCN    22
     8 YH,YSTART,DELETAX,DELFX                                              PFUNCN    23
      PFUNCN = 0.                                                           PFUNCN    24
      S = (((X(1) - Y(1))*SBETA + (Y(2) - X(2))*CBETA)*DELF                 PFUNCN    25
     1 + (X(3) - Y(3))*DELETA1)/SWPOST                                      PFUNCN    26
      Q = DELF**2 + DELETA1**2 - S**2                                       PFUNCN    27
      IF(Q.LT.1.E-10) RETURN                                                PFUNCN    28
      PFUNCN = DELF*(OMEGAR*XETA - VH(2)*CBETA + VH(1)*SBETA)/SQRT(Q)       PFUNCN    29
      RETURN                                                                PFUNCN    30
      END                                                                   PFUNCN    31
```

Subroutine LEETA

Subroutine LEETA returns the coordinates (in the YH frame) of the point of intersection of a fixed arc and the leading edge of a blade. The second ETA component (see fig. 2) is also returned. A second entry point, TEETA, does the same thing for a trailing edge.

APPENDIX A

Flow Chart of Subroutine LEETA



APPENDIX A

Listing of Subroutine LEETA

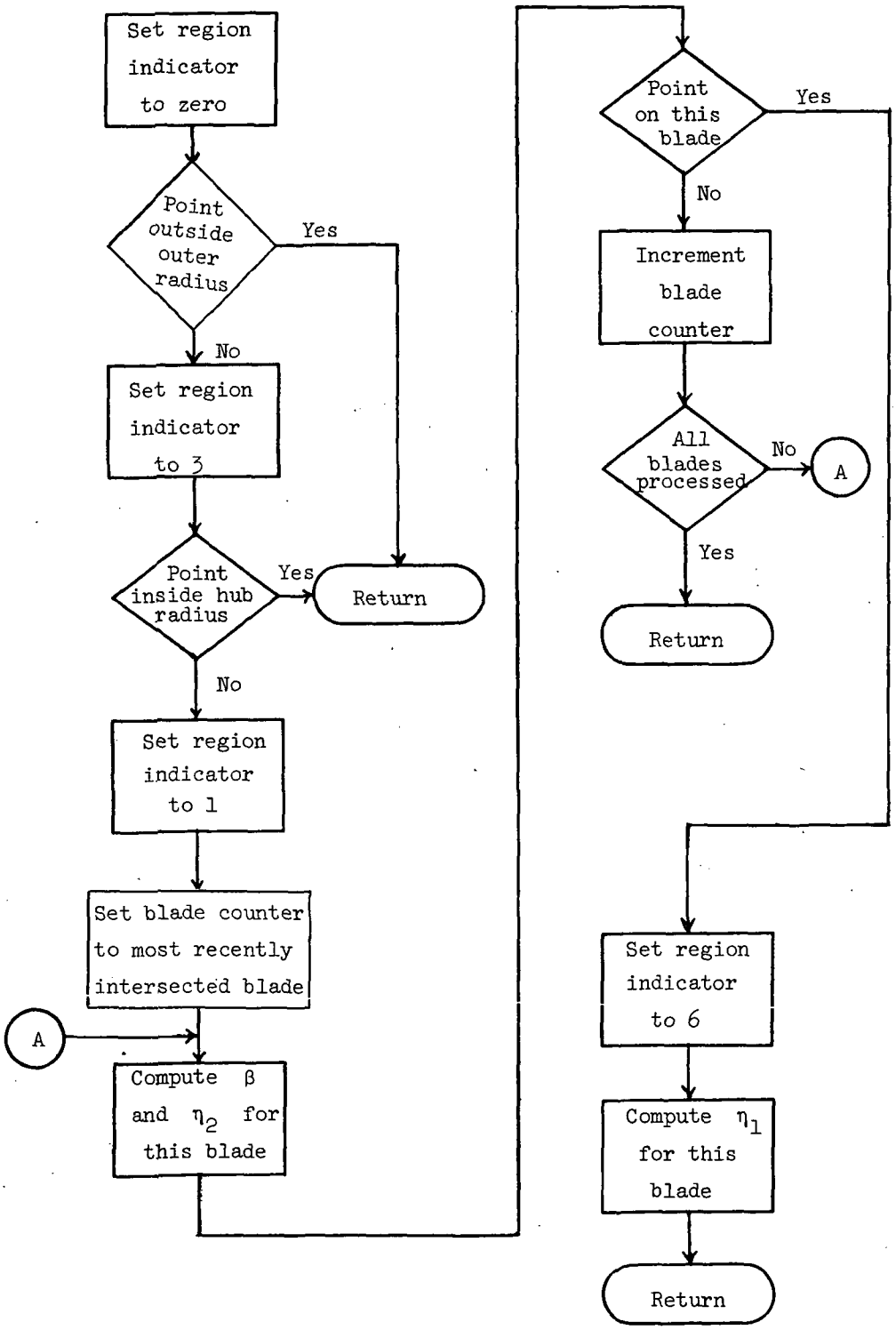
```
      SUBROUTINE LEETA(XETA,Y)                                                  LEETA      2
C*************************************************************************** LEETA      3
C*                                                                          * LEETA      4
C*       SUBROUTINE LEETA IS CALLED BY SUBROUTINE CALCSP. THIS              * LEETA      5
C*         ROUTINE RETURNS THE COORDINATES (IN THE Y REFERENCE              * LEETA      6
C*         FRAME) OF THE INTERSECTION OF A FIXED SWEEPING ARC AND           * LEETA      7
C*         THE LEADING EDGE OF A BLADE. A SECOND ENTRY POINT, TEETA,        * LEETA      8
C*         DOES THE SAME FOR THE TRAILING EDGE. THE SECOND ETA              * LEETA      9
C*         COMPONENT IS ALSO RETURNED.                                      * LEETA     10
C*                                                                          * LEETA     11
C*************************************************************************** LEETA     12
      DIMENSION    BETA(8),       COEFFS(5),     ETA(2,8),                    LEETA     13
     1             X(3),          XC(3),         Y(3),                        LEETA     14
     2             YEND(2),       YH(3),         YSTART(2),                   LEETA     15
     3             VH(3)                                                      LEETA     16
      COMMON        ALFA,     BETA,     BLADES,   BLNTN,    CBETA,           LEETA     17
     1              CH,       COEFFS,   DELTAU,   ETA,      ETA2,            LEETA     18
     2              ETAMAX,   FAC,      FN,       FNS,      FETA1,           LEETA     19
     3              IFLAG,    IREGION,  J2,       JFLG,     KFLG,            LEETA     20
     4              NBLADES,  OMEGAR,   OTIME,    PI,       R,               LEETA     21
     5              RM,       RMIN,     S,        SBETA,    SNDSPD,          LEETA     22
     6              SRCOST,   SWPDST,   SWPTAU,   TAU,      THKRAT,          LEETA     23
     7              TIMTHRU,  VH,       X,        XC,       YEND,            LEETA     24
     8              YH,       YSTART,   DELETA1,  DELF                       LEETA     25
      A = -0.5*CH                                                             LEETA     26
      GO TO 10                                                                LEETA     27
      ENTRY TEETA                                                             LEETA     28
      A = 0.5*CH                                                              LEETA     29
   10 B = -A                                                                  LEETA     30
      XBAR1 = X(1) + A*SBETA                                                  LEETA     31
      XBAR2 = X(2) + B*CBETA                                                  LEETA     32
      XBAR = X(1)*CBETA + X(2)*SBETA                                          LEETA     33
      XTILDE = XBAR1*XBAR1 + XBAR2*XBAR2 - SRCOST*SRCOST                      LEETA     34
      DISC1 = SQRT(XBAR*XBAR - XTILDE)                                        LEETA     35
      ETA2 = -XBAR + DISC1                                                    LEETA     36
      IF(ABS(XETA - ETA2).GE.CH/FN) ETA2 = ETA2 - 2.*DISC1                    LEETA     37
      Y(1) = B*SBETA -ETA2*CBETA                                              LEETA     38
      Y(2) = A*CBETA -ETA2*SBETA                                              LEETA     39
      RETURN                                                                  LEETA     40
      END                                                                     LEETA     41
```

Subroutine TEST

TEST returns the region index (IREGION) for a given pair of coordinates in the YH reference frame at a given time. The region indices correspond to points outside the outer radius, inside the hub radius, within the rotor system but not a blade, and on a blade. When a point is found to be on a blade, the routine returns the blade number, the angle made by that blade, the sine and cosine of this angle, and the ETA coordinates of the point. The subroutine included with the current version of RTN is limited to rotor systems with rectangular blades.

APPENDIX A

Flow Chart of Subroutine TEST



APPENDIX A

Listing of Subroutine TEST

```
      SUBROUTINE TEST(Y,RSQ,ROSQ,TIME)                                  TEST      2
C*************************************************************************TEST      3
C*                                                                       * TEST      4
C*       SUBROUTINE TEST IS CALLED BY SUBROUTINE CALCSP. THIS            * TEST      5
C*         ROUTINE RETURNS THE REGION INDEX FOR A GIVEN TIME AND         * TEST      6
C*         A GIVEN PAIR OF COORDINATES IN THE Y REFERENCE FRAME.         * TEST      7
C*         THE REGION INDEX HAS FOUR POSSIBLE VALUES.                    * TEST      8
C*                                                                       * TEST      9
C*           IREGION = 0:  THE POINT LIES OUTSIDE THE CIRCLE             * TEST     10
C*                         DEFINED BY THE BLADES OUTER RADIUS            * TEST     11
C*                                                                       * TEST     12
C*                   = 1:  THE POINT LIES BETWEEN THE CIRCLES            * TEST     13
C*                         DEFINED BY THE BLADES OUTER AND               * TEST     14
C*                         INNER RADII,BUT IS NOT ON A BLADE             * TEST     15
C*                                                                       * TEST     16
C*                   = 3:  THE POINT LIES INSIDE THE CIRCLE              * TEST     17
C*                         DEFINED BY THE BLADES INNER RADIUS            * TEST     18
C*                                                                       * TEST     19
C*                   = 6:  THE POINT LIES ON A BLADE.                    * TEST     20
C*                                                                       * TEST     21
C*       WHEN THE REGION INDEX IS 6, TEST RETURNS THE BLADE NUMBER,      * TEST     22
C*         J2, THE ANGLE MADE BY THAT BLADE, THE SIN AND COSINE OF       * TEST     23
C*         THIS ANGLE, SBETA AND CBETA, AND THE ETA COORDINATES OF       * TEST     24
C*         THE POINT.                                                    * TEST     25
C*                                                                       * TEST     26
C*************************************************************************TEST     27
      DIMENSION    BETA(8),      COEFFS(5),     ETA(2,8),                TEST     28
     1             X(3),         X0(3),         Y(3),                    TEST     29
     2             YEND(2),      YH(3),         YSTART(2),               TEST     30
     3             VH(3)                                                 TEST     31
      COMMON       ALFA,     BETA,     BLADES,   BLNTN,    CBETA,       TEST     32
     1             CH,       COEFFS,   DELTAU,   ETA,      ETA2,        TEST     33
     2             ETAMAX,   FAC,      FN,       FNS,      FETA1,       TEST     34
     3             IFLAG,    IREGION,  J2,       JFLG,     KFLG,        TEST     35
     4             NBLADES,  OMEGAR,   OTIME,    PI,       R,           TEST     36
     5             RM,       RMIN,     S,        SBETA,    SNDSPD,      TEST     37
     6             SRCDST,   SWPDST,   SWPTAU,   TAU,      THKRAT,      TEST     38
     7             TIMTHRU,  VH,       X,        X0,       YEND,        TEST     39
     8             YH,       YSTART,   DELETA1,  DELF                   TEST     40
      RYSQ = Y(1)**2 + Y(2)**2                                          TEST     41
      IREGION = 0                                                       TEST     42
      IF(RYSQ.GT.RSQ) GO TO 3                                           TEST     43
      IREGION = 3                                                       TEST     44
      IF(RYSQ.LT.ROSQ) GO TO 3                                          TEST     45
      IREGION = 1                                                       TEST     46
      LIMTOP = J2 + NBLADES - 1                                         TEST     47
      DO 1 J=J2,LIMTOP                                                  TEST     48
      I = J                                                             TEST     49
      IF(J.GT.NBLADES) I = J - NBLADES                                  TEST     50
      BETA(I) = OMEGAR*TIME + FLOAT(I-1)*6.28318530B/FLOAT(NBLADES)     TEST     51
      CBETA = COS(BETA(I))                                              TEST     52
      SBETA = SIN(BETA(I))                                              TEST     53
      IF((Y(2)*CBETA - Y(1)*SBETA)**2.GT.CH*CH/4.) GO TO 1              TEST     54
      ETA(2,I) = -Y(1)*CBETA - Y(2)*SBETA                               TEST     55
      IF(ETA(2,I).GT.0.) GO TO 2                                        TEST     56
    1 CONTINUE                                                          TEST     57
      CBETA = COS(BETA(J2))                                             TEST     58
      SBETA = SIN(BETA(J2))                                             TEST     59
      GO TO 3                                                           TEST     60
    2 IREGION = 6                                                       TEST     61
      J2 = I                                                            TEST     62
      ETA(1,J2) = CH/2. - Y(1)*SBETA + Y(2)*CBETA                       TEST     63
    3 RETURN                                                            TEST     64
      END                                                               TEST     65
```

APPENDIX B

ROTOR THICKNESS NOISE POSTPROCESSOR PROGRAM

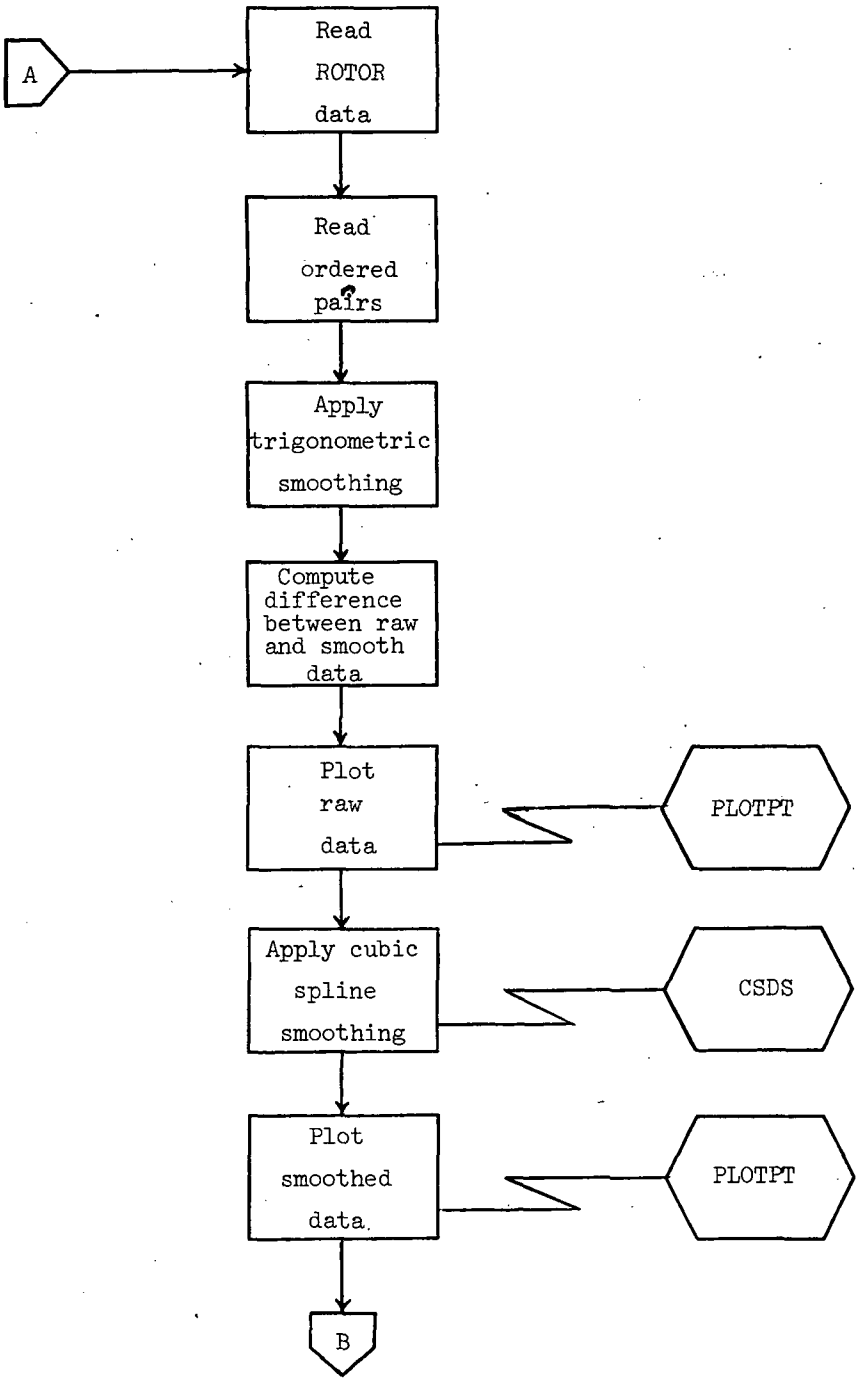
This appendix includes brief descriptions of each of the RTNPP subprograms. Where possible, each description is followed by a flow chart and FORTRAN listing of the subprogram. The CSDS subroutine is the property of International Mathematical and Statistical Library (IMSL). Four other subroutines, SPLDER, PSEUDO, INFOPLT, and CALPLT, are the property of Langley Research Center. For these five subroutines, FORTRAN listings are not included, but the more detailed descriptions (ref. 7) provided should permit their replacement by similar routines which might be available at other facilities.

Program RTNPP

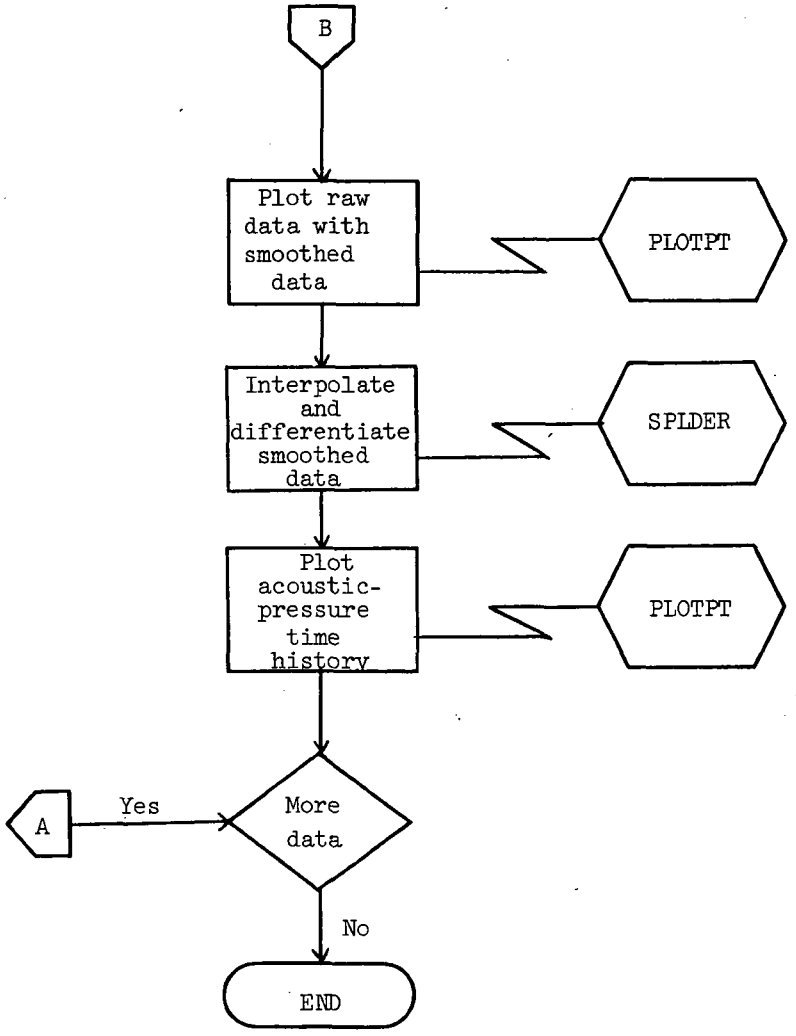
Program RTNPP reads the output from the RTN program. If the contents of TAPE12 were disposed to the card punch following execution of the RTN program, this deck should be placed in the RTNPP input stream. If TAPE12 is a disk file, it should be rewound prior to executing RTNPP. No other input is necessary for the postprocessor program. Once the data has been read and stored into arrays, trigonometric smoothing is used to obtain an estimate of the validity of each data point in the $\mathcal{P}(\vec{X},t)$ plotted against t spectrum. These estimates are then passed to a cubic spline smoothing routine. Interpolation is performed on the smoothed data to provide values at equally spaced values of the observer time. The smoothed data are numerically differentiated with respect to the observer time and the results are plotted.

APPENDIX B

Flow Chart of Program RTNPP



APPENDIX B


B
Plot raw data with smoothed data
PLOTPT
Interpolate and differentiate smoothed data
SPLDER
Plot acoustic-pressure time history
PLOTPT
A
Yes
More data
No
END

APPENDIX B

Listing of Program RTNPP

```
      PROGRAM RTNPP(INPUT=101B,OUTPUT=101B,TAPE5=INPUT,TAPE6=OUTPUT)      RTNPP      2
      DIMENSION SCRIPTP(350),OTIME(350),F(500),T(500),DER1(500),          RTNPP      3
     1 DER2(500),COEF(350,4),WK(3850),DF(350)                             RTNPP      4
      DIMENSION X0(3),COEFFS(5)                                           RTNPP      5
      LOGICAL MORDATA                                                     RTNPP      6
      NAMELIST/ROTOR/XC,NBLADES,R,CH,OMEGA,RO,THKRAT,DELTET,N,NS,KFLG,    RTNPP      7
     1 ETAMAX,BLNTN,JFLG,COEFFS,TAUINT,PERDM,DTAUM,SNDSPD,MORDATA         RTNPP      8
    1 FORMAT(1HC,*IERR = *,I5)                                            RTNPP      9
    2 FORMAT(I5,3E25.15)                                                  RTNPP     10
      M = 0                                                               RTNPP     11
      INIT = 0                                                            RTNPP     12
   10 READ(5,ROTOR)                                                       RTNPP     13
      WRITE(6,ROTOR)                                                      RTNPP     14
   20 CONTINUE                                                            RTNPP     15
      M = M + 1                                                           RTNPP     16
   30 CONTINUE                                                            RTNPP     17
      READ(5,2) I,Z,OTIME(M),SCRIPTP(M)                                   RTNPP     18
      OTIME(M) = 1000.*OTIME(M)                                           RTNPP     19
      IF(I.NE.0) GO TO 20                                                 RTNPP     20
      M = M - 1                                                           RTNPP     21
      MNPTS = 350                                                         RTNPP     22
      NCVS = 1                                                            RTNPP     23
      M = M/3                                                             RTNPP     24
      M = 3*M                                                             RTNPP     25
      DT = (OTIME(M) - OTIME(1))/499.                                     RTNPP     26
      DO 40 IJ=1,500                                                      RTNPP     27
      T(IJ) = OTIME(1) + FLOAT(IJ-1)*DT                                   RTNPP     28
   40 CONTINUE                                                            RTNPP     29
      MM = 500                                                            RTNPP     30
      S = FLOAT(M) + SQRT(FLOAT(2*M))                                     RTNPP     31
      PI = 3.141592654                                                    RTNPP     32
      NM1 = M - 1                                                         RTNPP     33
      DO 50 I=1,M                                                         RTNPP     34
      DER1(I) = 0.                                                        RTNPP     35
      WK(I) = 0.                                                          RTNPP     36
      DF(I) = SCRIPTP(I) - SCRIPTP(1) - FLOAT(I-1)*(SCRIPTP(M) - SCRIPTP  RTNPP     37
     1 (1))/FLOAT(NM1)                                                    RTNPP     38
   50 CONTINUE                                                            RTNPP     39
      NNMM= MIN0(85,M)                                                    RTNPP     40
      DO 70 J=1,NNMM                                                      RTNPP     41
      DO 60 I=2,NM1                                                       RTNPP     42
      WK(J) = WK(J) + DF(I)*SIN(FLOAT(J*(I-1))*PI/FLOAT(NM1))             RTNPP     43
   60 CONTINUE                                                            RTNPP     44
      WK(J) = 2.*WK(J)/FLOAT(NM1)                                         RTNPP     45
   70 CONTINUE                                                            RTNPP     46
      DO 90 I=2,NM1                                                       RTNPP     47
      DO 80 J=1,NNMM                                                      RTNPP     48
      DER1(I) = DER1(I) + WK(J)*SIN(FLOAT(J*(I-1))*PI/FLOAT(NM1))         RTNPP     49
   80 CONTINUE                                                            RTNPP     50
      COEF(I,1) = DER1(I) + SCRIPTP(1) + FLOAT(I-1)*(SCRIPTP(M)           RTNPP     51
     1 - SCRIPTP(1))/FLOAT(NM1)                                           RTNPP     52
   90 CONTINUE                                                            RTNPP     53
      COEF(1,1) = SCRIPTP(1)                                              RTNPP     54
      COEF(M,1) = SCRIPTP(M)                                              RTNPP     55
      DO 100 I=1,M                                                        RTNPP     56
      DF(I) =1.5*ABS(COEF(I,1) - SCRIPTP(I))                              RTNPP     57
  100 CONTINUE                                                            RTNPP     58
      ML = 0                                                              RTNPP     59
      IF(INIT.GT.0) ML = 1                                                RTNPP     60
      INIT = INIT + 1                                                     RTNPP     61
      CALL PLOTPT(OTIME,SCRIPTP,M,ML,1)                                   RTNPP     62
      ML = 1                                                              RTNPP     63
      IW = -1                                                             RTNPP     64
```

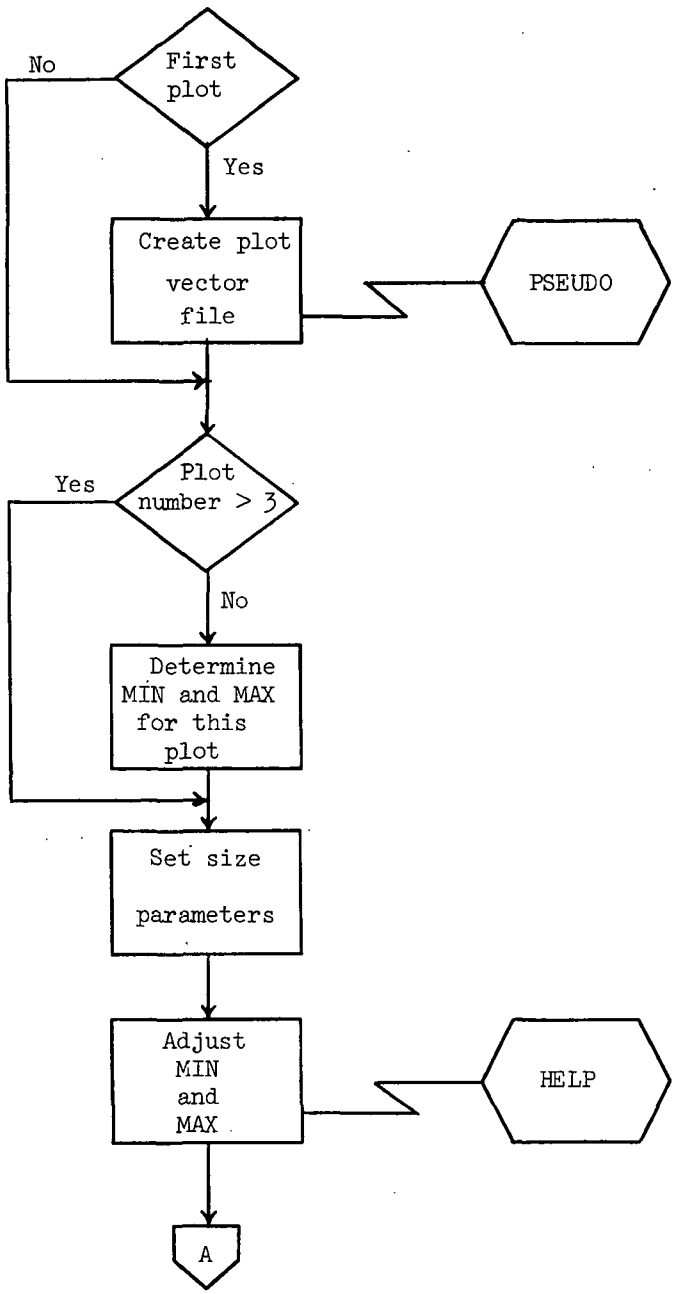
```
      CALL CSDS(MNPTS,M,OTIME,SCRIPTP,DF,S,IW,COEF,WK,IERR)          RTNPP      65
      WRITE(6,1) IERR                                                RTNPP      66
      CALL PLOTPT(OTIME,COEF,M,ML,2)                                 RTNPP      67
      XMN = OTIME(1)                                                 RTNPP      68
      XMX = OTIME(M)                                                 RTNPP      69
      YMN = 1.E10                                                    RTNPP      70
      YMX = -1.E10                                                   RTNPP      71
      DO 110 I=1,M                                                   RTNPP      72
      IF(SCRIPTP(I).LT.YMN) YMN = SCRIPTP(I)                         RTNPP      73
      IF(SCRIPTP(I).GT.YMX) YMX = SCRIPTP(I)                         RTNPP      74
      IF(COEF(I,1).LT.YMN) YMN = COEF(I,1)                           RTNPP      75
      IF(COEF(I,1).GT.YMX) YMX = COEF(I,1)                           RTNPP      76
  110 CONTINUE                                                       RTNPP      77
      OTIME(M+1) = XMN                                               RTNPP      78
      OTIME(M+2) = XMX                                               RTNPP      79
      SCRIPTP(M+1) = YMN                                             RTNPP      80
      SCRIPTP(M+2) = YMX                                             RTNPP      81
      COEF(M+1,1) = YMN                                              RTNPP      82
      COEF(M+2,1) = YMX                                              RTNPP      83
      CALL PLOTPT(OTIME,SCRIPTP,M,ML,4)                              RTNPP      84
      CALL PLOTPT(OTIME,COEF,M,ML,5)                                 RTNPP      85
      IW = -1                                                        RTNPP      86
      CALL SPLDER(MNPTS,M,NCVS,MM,MM,OTIME,COEF,T,F,DER1,DER2,IW,WK, RTNPP      87
     1 IERR)                                                         RTNPP      88
      WRITE(6,1) IERR                                                RTNPP      89
      DO 120 I=1,MM                                                  RTNPP      90
      DER1(I) = 1000.*DER1(I)                                        RTNPP      91
  120 CONTINUE                                                       RTNPP      92
      ML = 2                                                         RTNPP      93
      IF(MORDATA) ML = 1                                             RTNPP      94
      CALL PLOTPT(T,DER1,MM,ML,3)                                    RTNPP      95
      M = 0                                                          RTNPP      96
      IF(MORDATA) GO TO 10                                           RTNPP      97
      STOP                                                           RTNPP      98
      END                                                            RTNPP      99
```

Subroutine PLOTPT

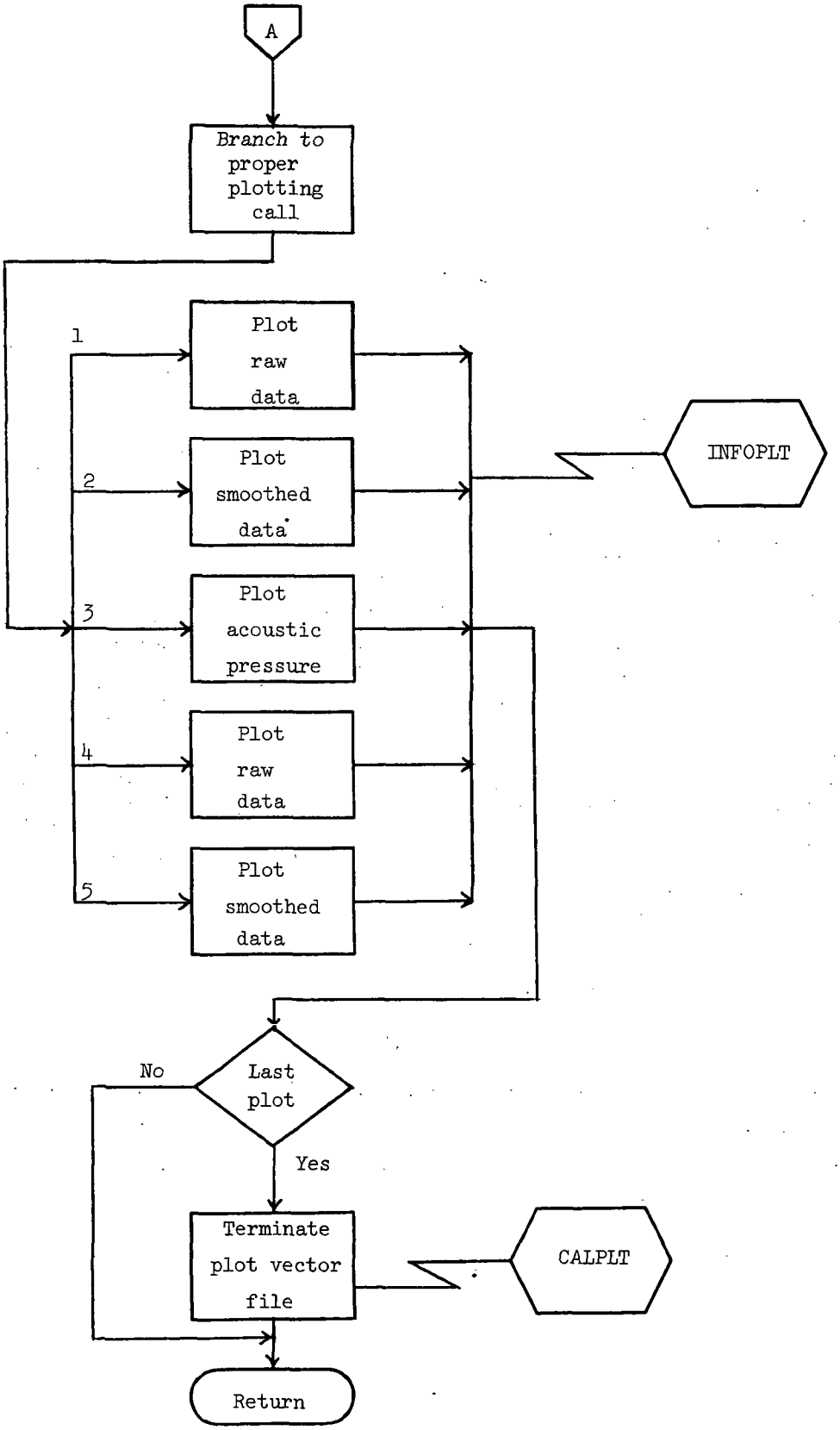
This subroutine serves as an interface between RTNPP and the graphics output library. Plot size and captions are provided for each plot.

APPENDIX B

Flow Chart of Subroutine PLOTPT



APPENDIX B


A
Branch to proper plotting call
1
Plot raw data
2
Plot smoothed data
3
Plot acoustic pressure
4
Plot raw data
5
Plot smoothed data
INFOPLT
No
Last plot
Yes
Terminate plot vector file
CALPLT
Return

APPENDIX B

Listing of Subroutine PLOTPT

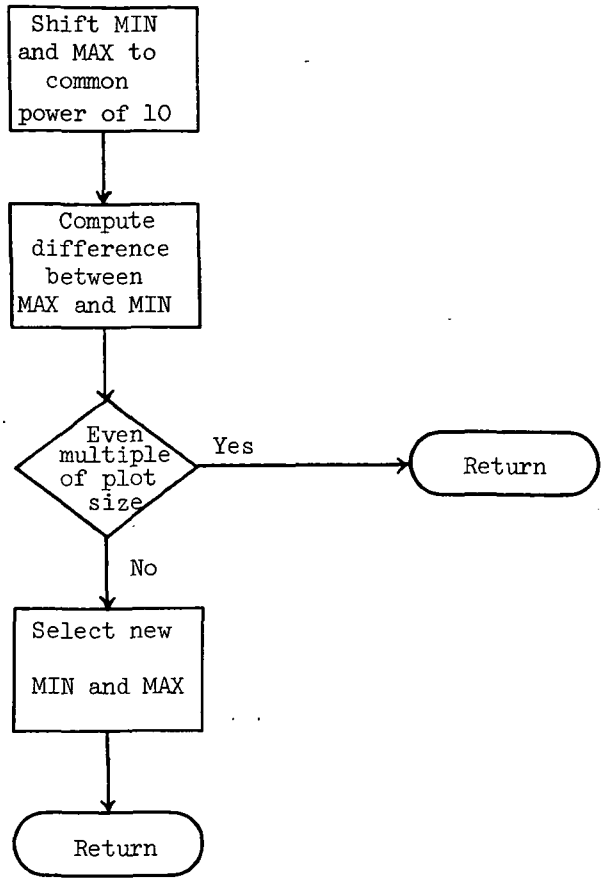
```
      SUBROUTINE PLOTPT(X,Y,N,ML,J)                                      PLOTPT     2
      DIMENSION X(1),Y(1)                                                PLOTPT     3
      IF(ML.EQ.0) CALL PSEUDO                                            PLOTPT     4
      IF(J.GT.3) GO TO 15                                                PLOTPT     5
      XMIN = 1.E10                                                       PLOTPT     6
      YMIN = 1.E10                                                       PLOTPT     7
      XMAX =-1.E10                                                       PLOTPT     8
      YMAX =-1.E10                                                       PLOTPT     9
      DO 10 I=1,N                                                        PLOTPT    10
      IF(X(I).LT.XMIN) XMIN = X(I)                                       PLOTPT    11
      IF(Y(I).LT.YMIN) YMIN = Y(I)                                       PLOTPT    12
      IF(X(I).GT.XMAX) XMAX = X(I)                                       PLOTPT    13
      IF(Y(I).GT.YMAX) YMAX = Y(I)                                       PLOTPT    14
   10 CONTINUE                                                           PLOTPT    15
      GO TO 16                                                           PLOTPT    16
   15 CONTINUE                                                           PLOTPT    17
      XMIN = X(N+1)                                                      PLOTPT    18
      XMAX = X(N+2)                                                      PLOTPT    19
      YMIN = Y(N+1)                                                      PLOTPT    20
      YMAX = Y(N+2)                                                      PLOTPT    21
   16 CONTINUE                                                           PLOTPT    22
      SH = 8.                                                            PLOTPT    23
      SV = 5.                                                            PLOTPT    24
      ISH = 8                                                            PLOTPT    25
      ISV = 5                                                            PLOTPT    26
      CALL HELP(XMIN,XMAX,ISH,XMIN,XMAX)                                 PLOTPT    27
      CALL HELP(YMIN,YMAX,ISV,YMIN,YMAX)                                 PLOTPT    28
      GO TO (20,30,40,60,70),J                                           PLOTPT    29
   20 CALL INFOPLT(1,N,X,1,Y,1,XMIN,XMAX,YMIN,YMAX,1.,                   PLOTPT    30
     1 13,13HOBSERVER TIME,18,18HSCRIPTP (RAW DATA),0,SH,SV,1.,1.)       PLOTPT    31
      GO TO 50                                                           PLOTPT    32
   30 CALL INFOPLT(1,N,X,1,Y,1,XMIN,XMAX,YMIN,YMAX,1.,                   PLOTPT    33
     1 13,13HOBSERVER TIME,16,16HSCRIPTP (SPLINE),0,SH,SV,1.,1.)         PLOTPT    34
      GO TO 50                                                           PLOTPT    35
   40 CALL INFOPLT(1,N,X,1,Y,1,XMIN,XMAX,YMIN,YMAX,1.,                   PLOTPT    36
     1 28,28HOBSERVER TIME (MILLISECONDS),17,17HPRESSURE (N/M**2),0,     PLOTPT    37
     2 SH,SV,1.,1.)                                                     PLOTPT    38
      GO TO 50                                                           PLOTPT    39
   60 CALL INFOPLT(0,N,X,1,Y,1,XMIN,XMAX,YMIN,YMAX,                      PLOTPT    40
     1 1.,13,13HOBSERVER TIME,7,7HSCRIPTP,0,SH,SV,1.,1.)                 PLOTPT    41
      GO TO 50                                                           PLOTPT    42
   70 CALL INFOPLT(1,N,X,1,Y,1,XMIN,XMAX,YMIN,YMAX,                      PLOTPT    43
     1 1.,13,13HOBSERVER TIME,7,7HSCRIPTP,0,SH,SV,1.,1.)                 PLOTPT    44
   50 IF(ML.EQ.2) CALL CALPLT(0.,0.,999)                                 PLOTPT    45
      RETURN                                                             PLOTPT    46
      END                                                                PLOTPT    47
```

Subroutine HELP

This routine serves to eliminate the problem of rounding associated with the annotation of the values of the variable at the tic marks. If the specified values for the minimum and maximum result in rounding, a smaller minimum and larger maximum are substituted so that rounding does not occur.

APPENDIX B

Flow Chart of Subroutine HELP



Listing of Subroutine HELP

```
      SUBROUTINE HELP(X1,X2,D,XMIN,XMAX)                          HELP      2
      INTEGER D                                                   HELP      3
      P1 = ALOG10(ABS(X1))                                        HELP      4
      P2 = ALOG10(ABS(X2))                                        HELP      5
      P = AMAX1(P1,P2)                                            HELP      6
      IF(P.LT.0.) P = P - 1.                                      HELP      7
      IP = P - 2.                                                 HELP      8
      X1P = X1/10.**IP                                            HELP      9
      X2P = X2/10.**IP                                            HELP     10
      IX1P = X1P - 1.                                             HELP     11
      IX2P = X2P + 1.                                             HELP     12
      IDIF = IX2P - IX1P                                          HELP     13
      N = IDIF/(D+1)                                              HELP     14
      JDIF = N*(D+1)                                              HELP     15
      IF(IDIF.EQ.JDIF) GO TO 10                                   HELP     16
      N = N + 1                                                   HELP     17
      KDIF = N*(D+1)                                              HELP     18
      LDIF = (KDIF - IDIF)/2                                      HELP     19
      IX1P = IX1P - LDIF                                          HELP     20
      IX2P = IX1P + KDIF                                          HELP     21
   10 CONTINUE                                                    HELP     22
      XMIN = FLOAT(IX1P)*10.**IP                                  HELP     23
      XMAX = FLOAT(IX2P)*10.**IP                                  HELP     24
      RETURN                                                      HELP     25
      END                                                         HELP     26
```

APPENDIX B

Subroutine CSDS

Language: FORTRAN

Purpose: To fit a smooth cubic spline curve to a set of data points representing a univariate function. The functional values are allowed to vary by specified amounts in smoothing the curve. Data points may be unequally spaced.

Use: CALL CSDS (MAX,IX,X,F,DF,S,IPT,COEF,WK,IERR)

MAX — An input integer specifying the maximum number of data points for the independent variable, as given in the dimension statement of the calling program.

IX — An input integer specifying the actual number of data points for the independent variable. IX ≦ MAX.

X — A one-dimensional input array dimensioned at least IX in the calling program. Upon entry to CSDS, X(I) must contain the I-th value of the independent variable.

F — A one-dimensional input array dimensioned at least IX in the calling program. Upon entry to CSDS, F(I) must contain the I-th value of the function.

DF — A one-dimensional input array dimensioned at least IX in the calling program. Upon entry to CSDS, DF(I) must contain an estimate of the standard deviation of F(I). The value DF(I) determines the amount of variation permitted in the I-th functional value in smoothing the data. If the standard deviations of the functional values cannot be estimated, specifying DF = 1 is suggested.

S — A nonnegative input parameter which controls the extent of smoothing. A value in the range (IX - (2*IX)**.5) ≦ S ≦ (IX + (2*IX)**.5) is suggested. Increasing values of S permit increasing amounts of smoothing of the functional values. If S = 0 is specified, an unsmoothed cubic spline curve passing through the data points is computed.

IPT — An input initialization parameter. The user must specify IPT = -1 whenever a new X-array is input. The routine will then check to insure that the X-array is in strictly increasing order.

COEF — A two-dimensional output array which must be dimensioned (MAX,4) in the calling program. Upon return, COEF(I,J) contains the J-th coefficient of the spline for the interval beginning at point X(I). The functional value of the spline at abscissa X1, where X(I) ≦ X1 ≦ X(I + 1) is given by:

F(X1) = ((COEF(I,4)*H + COEF(I,3))*H + COEF(I,2))*H + COEF(I,1)

where H = X1 - X(I)

WK A one-dimensional work area array dimensioned at least (7*IX + 9) in the calling program.

IERR An output integer error parameter.

= 0 Normal return.

= J The J-th element of the X-array is not in strictly increasing order. No calculation performed.

= -1 There are less than four values in the X-array. No calculation performed.

Upon return, this parameter should be tested in the calling program.

Method: A set of IX data points (X_i,F_i), i = 1,2,. . .IX are given, where F_i is the functional value at point X_i. A set of weights DF_1,DF_2. . .DF_{IX} and a nonnegative smoothing parameter S are also given.

The routine computes a cubic spline G(X) having the following properties:

i) $$\sum_{i=1}^{IX} \left\{ \left[G(X_i) - F_i \right] \Big/ DF_i \right\}^2 \leq S$$

ii) $$\int_{X_1}^{X_{IX}} \left[G''(X) \right]^2 dx = \text{Minimum of all splines satisfying property (i)}$$

Subroutine SPLDER

Language: FORTRAN

Purpose: To perform a cubic spline approximation, interpolation, and differentiation. SPLDER computes $F = f_i(X)$, $DER1 = f_i'(X)$, and $DER2 = f_i''(X)$ for any number of different dependent variable arrays associated with the independent variable array.

Use: CALL SPLDER (MNPTS,N,NCVS,MMAX,M,X,Y,T, F, DER1,DER2,IW,WK, IERR)

MNPTS An input integer specifying the maximum number of values in the independent variable array as stated in the dimension statement of the calling program.

N — An input integer specifying the number of values in the independent variable array. $N \leq MNPTS$.

NCVS — An input integer specifying the number of dependent variable tables associated with the independent variable.

NMAX — An input integer specifying the maximum number of values at which interpolation is desired as stated in the dimension statement of the calling program.

M — An input integer specifying the number of values at which interpolation is desired on this entry into SPLDER. $M \leq MMAX$.

X — A one-dimensional input array containing the independent variables. The array X should be dimensioned by at least N in the calling program, and the value must be monotonic.

Y — A two-dimensional input array containing the dependent variables. The array Y is dimensioned with variable dimension in the subroutine; therefore, Y must be dimensioned in the calling program with first dimension MNPTS and second dimension at least NCVS.

T — A one-dimensional input array containing the values of the independent variable for which values of the dependent variable and the first and second derivatives are desired. The array T must be dimensioned by at least M in the calling program.

F — A two-dimensional output array in which SPLDER stores the values of the function at the M values of the independent variable. The array F is dimensioned with variable dimension in the subroutine; therefore F must be dimensioned in the calling program with first dimension MMAX and second dimension at least NCVS.

DER1 — A two-dimensional output array in which SPLDER stores the values of the derivative of the function at the M values of the independent variable. The array DER1 is dimensioned by variable dimension in the subroutine; therefore, DER1 must be dimensioned in the calling program with first dimension MMAX and second dimension at least NCVS.

DER2 — A two-dimensional output array in which SPLDER stores the values of the second derivative of the function at the M values of the independent variable. The array DER2 is dimensioned by variable dimension in the subroutine; therefore, DER2 must be dimensioned in the calling program with first dimension MMAX and second dimension at least NCVS.

IW — An input-output integer.

INPUT: IW is the initialization integer. Whenever a new X- or Y-array is input, IW must be set to -1. This condition will cause the independent variable array to be tested to determine if it is increasing. Also, certain values pertaining to the X- and Y-arrays will be computed. These values will not change unless either the X- or Y-arrays are replaced.

OUTPUT: IW is an index pointer indicating that $X_{IW} \leqq X_O \leqq X_{IW+1}$. On the next call to SPLDER, the previous IW is used to begin the search for the interval containing the interpolation point.

WK An array used by SPLDER as a work area. WK must be dimensioned at least 3(NxNCVS) + 8N. This array should not be used elsewhere in the program.

IERR An output integer error code.

= 0 Normal return.

= 1 The independent variable array is not increasing. A message will be printed, "INDEPENDENT VARIABLE ARRAY NOT INCREASING IN SPLDER AT POSITION IIII X=XXXX,XXXX."

= 2 A value in the T-array is not within the limits of the X-array. There are no provisions for extrapolation; therefore, every T_j must satify $X_1 \leqq T_j \leqq X_n$.

Upon return to the calling program, the parameter IERR should be tested.

Method: The method used in SPLDER is that of the reference. The reference gives the derivative of a matrix equation relating the second derivative of a univariate spline function at the given values of the independent variable to the values of the function at these values of the independent variable. Values of the second derivative are assumed to be zero at the end points. The matrix equation is tridiagonal and is solved by the Thomas algorithm which is equivalent to Gaussian elimination without pivoting. Expressions are derived for the first and second derivatives of the spline function at any point in an interval in terms of the values of the spline function and its second derivative at the end points of the interval.

References: Greville, T. N. E., "Spline Functions, Interpolation and Numerical Quadrature," Mathematical Methods for Digital Computers, Vol. II, pp. 156-168, John Wiley and Sons, 1967.

APPENDIX B

Subroutine PSEUDO

Language: COMPASS

Purpose: To create and write an appropriately named Plot Vector File. Through linkages set up by an initial call to PSEUDO, all subsequent graphics data generated by the user will be routed through one of the PSEUDO entry points and written on the Plot Vector File.

Use: CALL PSEUDO

Subroutine INFOPLT

Language: FORTRAN

Purpose: To provide a one-call method of preparing plotting displays automatically.

Use: CALL INFOPLT(IEC,N,XDATA,KX,YDATA,KY,XMIN,XMAX,YMIN,YMAX, PCTPTS,NXMC,XM,NYMC,YM,ISYM,SX,SY,XOFF,YOFF), where

IEC The code for terminating the frame.

0 (1) Used to initialize a frame and plot first curve of multiple curves per frame. This value of IEC leaves frame incomplete and expects additional curves.

OR

(2) Used to plot second to nth curves on the frame. This value of IEC guarantees that all curves will be plotted in the same frame, although the scales for second to nth curve could be different depending on other options selected. If the scales are different from labeled scales, the rescaled values of the origin and the scale increment for the call are printed out in the right vertical margin of the frame.

1 (1) Used to plot one curve per frame. This value of IEC completes the frame.

OR

(2) Used to plot the last curve for the frame. This value of IEC guarantees that all curves will be plotted on the same frame, although the scales for the last curve could be different, depending on the other options selected. If the scales are different from label scales, the rescaled values of origin and scale increment for the call are printed out in the right vertical margin of the frame.

OR

(3) Used to complete frame only. Generates no plotting. Special call with two parameters only.

CALL INFOPLT(1,0)

-1 Used to plot a curve with scales from previous calls on the same frame. This value of IEC leaves frame incomplete. If the curve will not fit on the plot, a new frame is set up for the remaining curves.

Example:

Same Plot (Scales could change)

```
CALL INFOPLT(0,. . .
CALL INFOPLT(0,. . .
CALL INFOPLT(1,. . .
```

Same Plot if Possible (If scales change, create new frame)

```
CALL INFOPLT(0,. . .
CALL INFOPLT(-1,. . .
CALL INFOPLT(-1,. . .
CALL INFOPLT(1,0)
```

N The number of points to be plotted.

XDATA The name of the array containing the floating-point values of X to be plotted. If all data within the array are to be plotted, the parameter would be expressed as XDATA. If it is desired to plot only a portion of the array, the desired beginning location is specified; for example, XDATA(4) would begin the plotting at the fourth element of the array.

KX The interleave factor which specifies the sequence in which X-data are stored.

= 1 Indicates that values are stored sequentially.

= 2 Indicates that values are stored in every other location in the array, etc.

YDATA The name of the array containing the floating-point values of Y to be plotted.

KY The interleave factor which specifies the sequence in which Y-data are stored.

XMIN The minimum value for X.

XMAX The maximum value for X.

YMIN The minimum value for Y.

YMAX The maximum value for Y.

The routine checks for the first call only to determine if either (XMAX - XMIN) or (YMAX - YMIN) is equal to zero. When either is zero, the routine will scan the X- and/or Y-array to determine the limits. For multiple curves per display, the limits must be specified on the first call to include all curves since the limits from the first call will be used for all curves.

If any data falls outside the limits, it will be eliminated; but a count will be kept of all points dropped.

At the completion of a particular curve, if the percentage of points dropped exceeds the value of PCTPTS, the current curve data will be automatically rescaled and replotted. For multiple curves, the first curves would not be contained on the rescaled plot.

If values are given for XMIN, XMAX, YMIN, and YMAX different than on the first call and IEC = 0, the scales could be different on the second to nth call. If the scales are different, the new origin and scale increment will be printed out. The XMIN, XMAX, YMIN, and YMAX values are ignored on the second to nth call if IEC = -1.

PCTPTS The percentage of points in any one curve that may be off scale without automatic replotting. The value of PCTPTS is a floating-point number ranging from 0.0 to 1.

= 0.0 No points are allowed off scale. The data will be rescaled if any points are off scale.

= 0.5 50 percent of the points are allowed off scale before the curve will be rescaled. Any percentage may be expressed.

= 1.0 No rescaling will be done.

NXMC The number of characters for the X-label including embedded blanks. The number of characters is calculated in two ways depending on the option selected in XM.

(1) XM is written in form nHxxx. . ..

NXMC = n

(2) XM is the beginning storage location containing alphanumeric information.

NXMC = Number of words in array multiplied by 10. Each alphanumeric word contains 10 characters including blanks.

The sign of NXMC is used to control generation of the X axis.

> 0 An axis will be drawn and annotated.

= 0 No axis will be drawn; therefore, no annotation.

< 0 One-inch grid lines will be drawn in addition to the axis.

XM The label for horizontal annotation. This label may be expressed in two ways:

(1) The string of alphanumeric characters for the label may be written in the form:

nHxxx. . ..

(the same way an alpha message is written using FORTRAN format statements).

(2) The beginning storage location of an array containing alphanumeric information may be used.

NYMC The number of characters for the Y-label. (See explanation under NXMC.)

YM The label for vertical annotation. (See explanation under XM.)

ISYM An integer code specifying the symbol or mode to be used in plotting the data values.

0 Draws a line (no symbols)

SX The length of the X-axis in floating-point inches. The default SX is 10 inches.

SY The length of Y-axis in floating-point inches. The default SY is 10 inches.

XOFF The offset for Y-axis from frame origin. A nonzero XOFF allows room for the Y-axis annotation. The default XOFF is 0.75 inches. If preprinted grid paper is used, the user may specify XOFF = 1.0 or change offset with postprocessor option.

YOFF The offset for the X-axis from the frame origin. A nonzero YOFF allows room for the X-axis annotation. The default YOFF is 0.5 inches. If preprinted grid paper is used, the user may specify YOFF = 1.0 or change offset with postprocessor option.

Subroutine CALPLT

Language: FORTRAN

Purpose: To move the plotter pen to a new location with pen up or down.

Use: CALL CALPLT(X,Y,IPEN), where

X,Y Are the floating-point values for pen movement.

IPEN = 2 Pen down

= 3 Pen up

Negative IPEN will assign X = 0, Y = 0 as the location of the pen after moving the X,Y (create a new reference point).

= 999 Writes a terminating block address of 999 to terminate the Plot Vector File and all further processing is skipped.

CALL CALPLT(0.0,0.0,999)

REFERENCES

1. Deming, A. F.: Noise From Propellers With Symmetrical Sections at Zero Blade Angle. II. NACA TN 679, 1938.

2. Arnoldi, R. A : Propeller Noise Caused by Blade Thickness. Rep. R-0896-1, Res. Dep., United Aircraft Corp., Jan. 1956.

3. Billing, H., ed.: Modern Aeronautical Acoustics. Rep. & Transl. No. 960, British M. M.A.P. Volkenrode, July 1, 1947.

4. Farassat, F.: Some Research on Helicopter Rotor Noise Thickness and Rotational Noise. Second Interagency Symposium on University Research in Transportation Noise, Vol. I (Raleigh, N.C.), June 1974, pp. 363-370.

5. Farassat, F.: Theory of Noise Generation From Moving Bodies With an Application to Helicopter Rotors. NASA TR R-451, 1975.

6. Abbott, Ira H.; and Von Doenhoff, Albert E.: Theory of Wing Sections. McGraw-Hill Book Co., Inc., 1949.

7. Computer Programing Manual. Langley Research Center, NASA, 1975.
   Vol. I.- General Information.
   Vol. II.- Subprogram Library.
   Vol. III.- Manufacturers Manuals.
   Vol. IV.- Special Capabilities.

TABLE I.- ROTOR THICKNESS NOISE PROGRAM INPUT VARIABLES

Variable	Dimension	Type	Units
BLNTN	*	Real	**
CH	*	Real	meters
COEFFS	*	Real	**
DELTET	*	Real	degrees
DTAUM	*	Real	**
ETAMAX	*	Real	meters
JFLG	*	Integer	**
KFLG	*	Integer	**
MORDATA	*	Logical	**
N	*	Integer	**
NBLADES	*	Integer	**
NS	*	Integer	**
OMEGA	*	Real	rev/min
PERDM	*	Real	**
R	*	Real	meters
RO	*	Real	meters
SNDSPD	*	Real	meters/sec
TAUINT	*	Real	seconds
THKRAT	*	Real	**
X0	3	Real	meters

*Single variable, not an array.

**Not applicable, unitless.

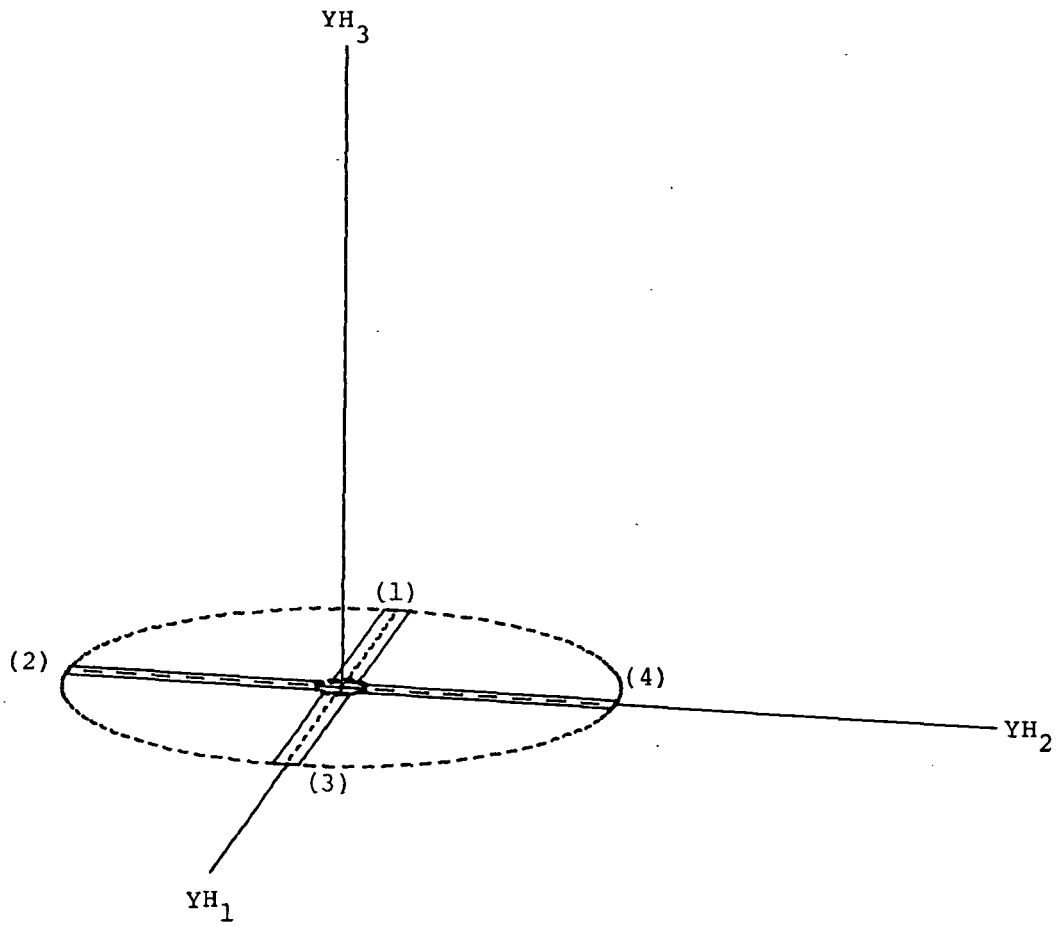


Figure 1.- Definition of YH reference frame based upon orientation of rotor system at time TAU = 0. (Numbers in parentheses are blade identification numbers.) Reference frames YH and Y are identical at TAU = 0.

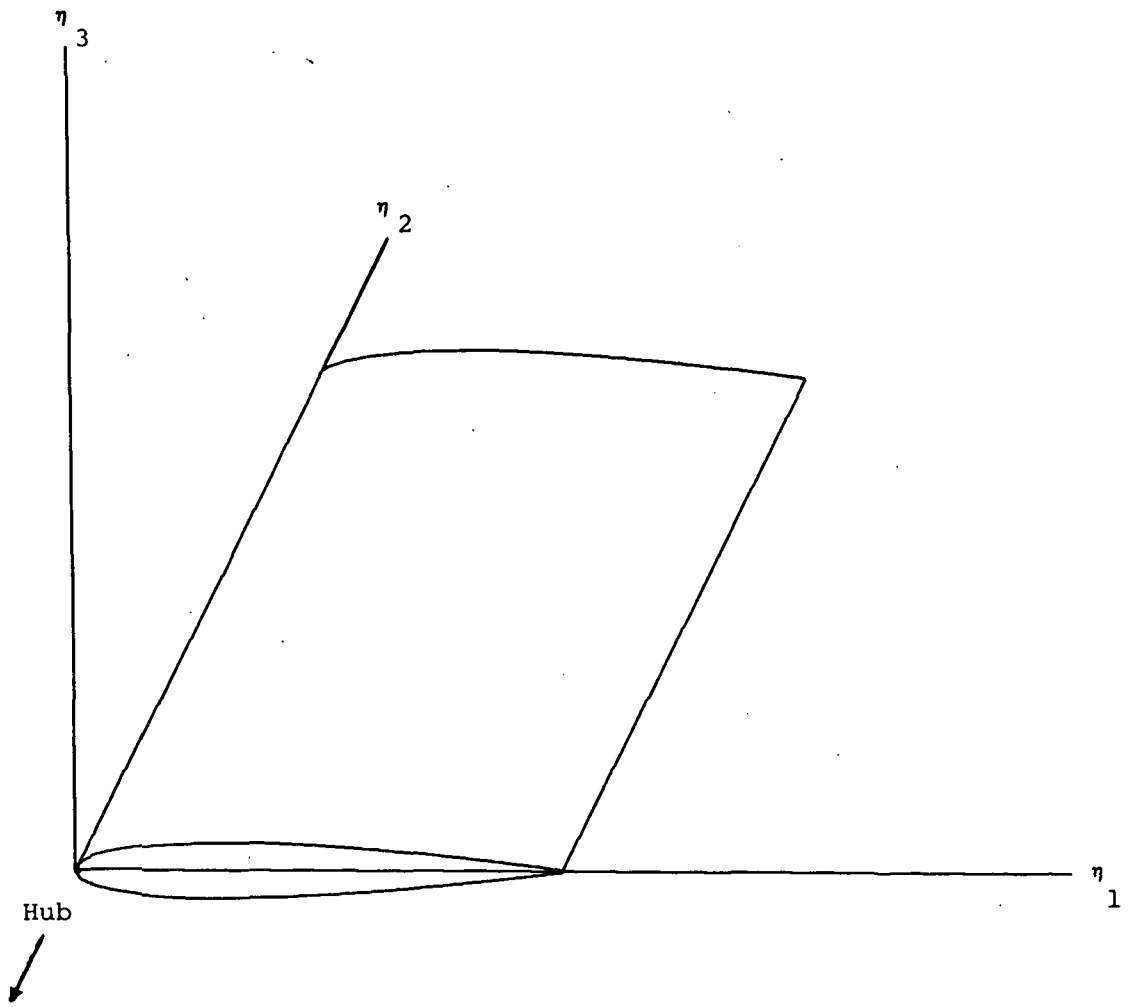


Figure 2.- Definition of blade coordinate system.

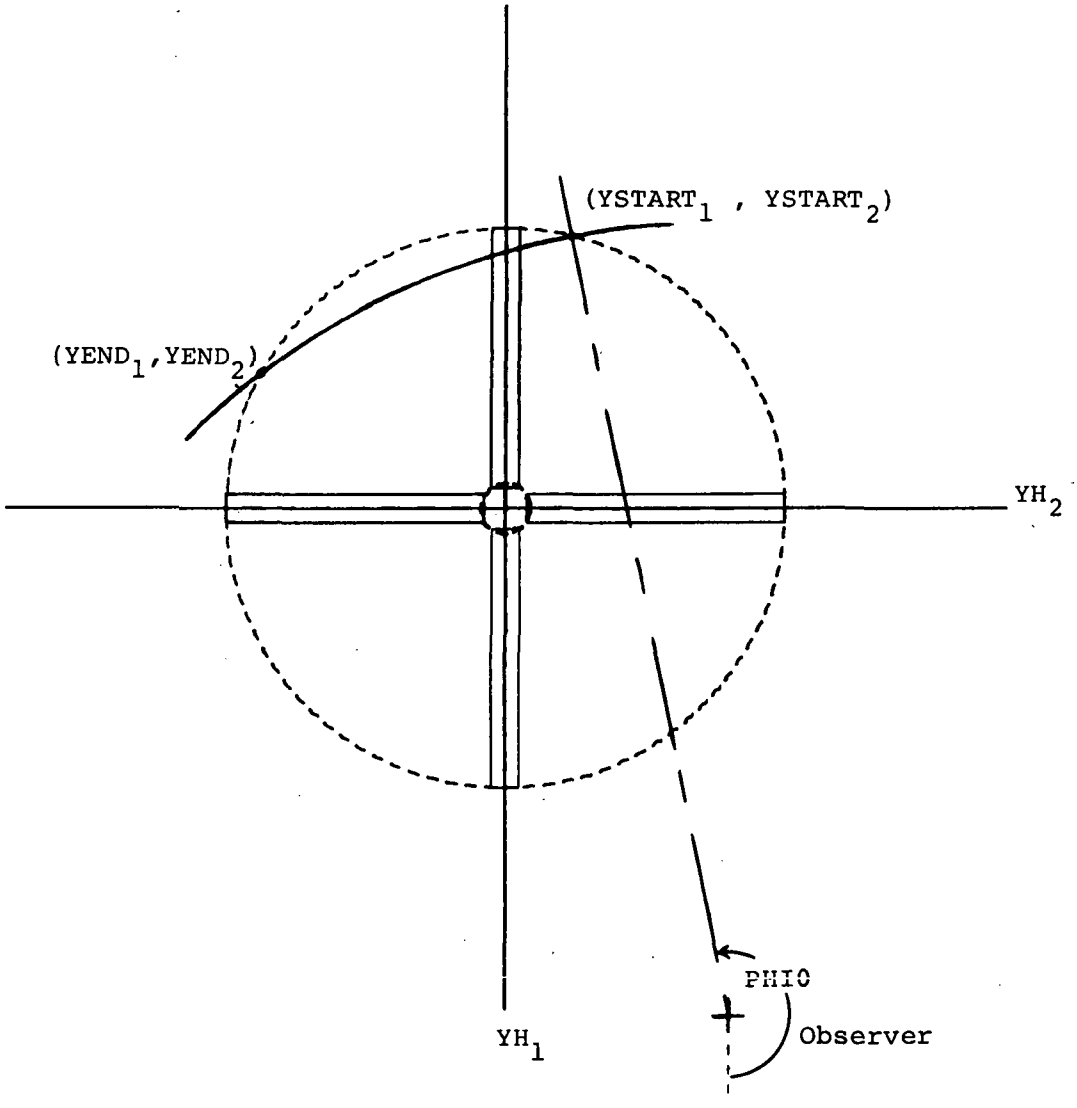


Figure 3.- Definition of arc used to sweep fixed position of rotor system.

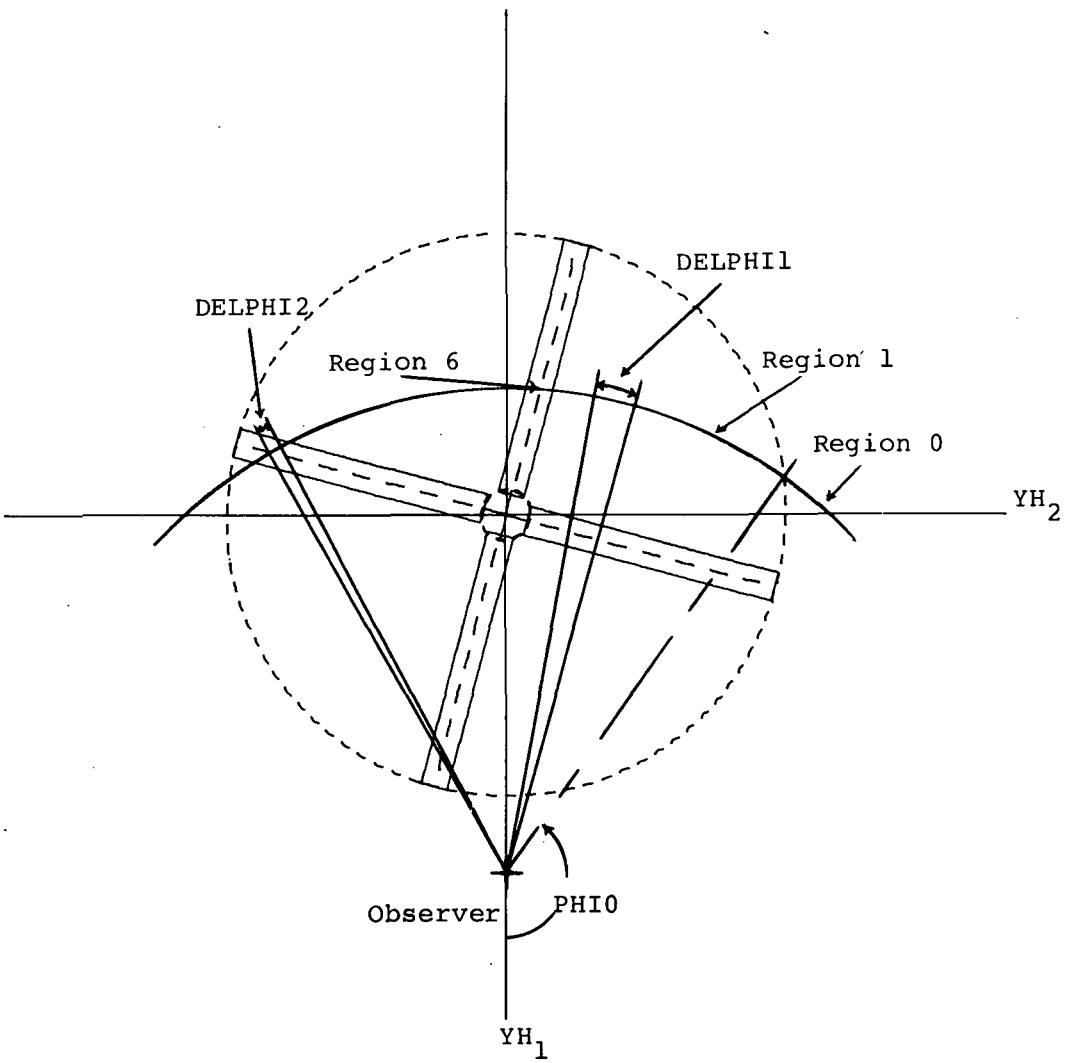


Figure 4.- Definition of sweep regions.

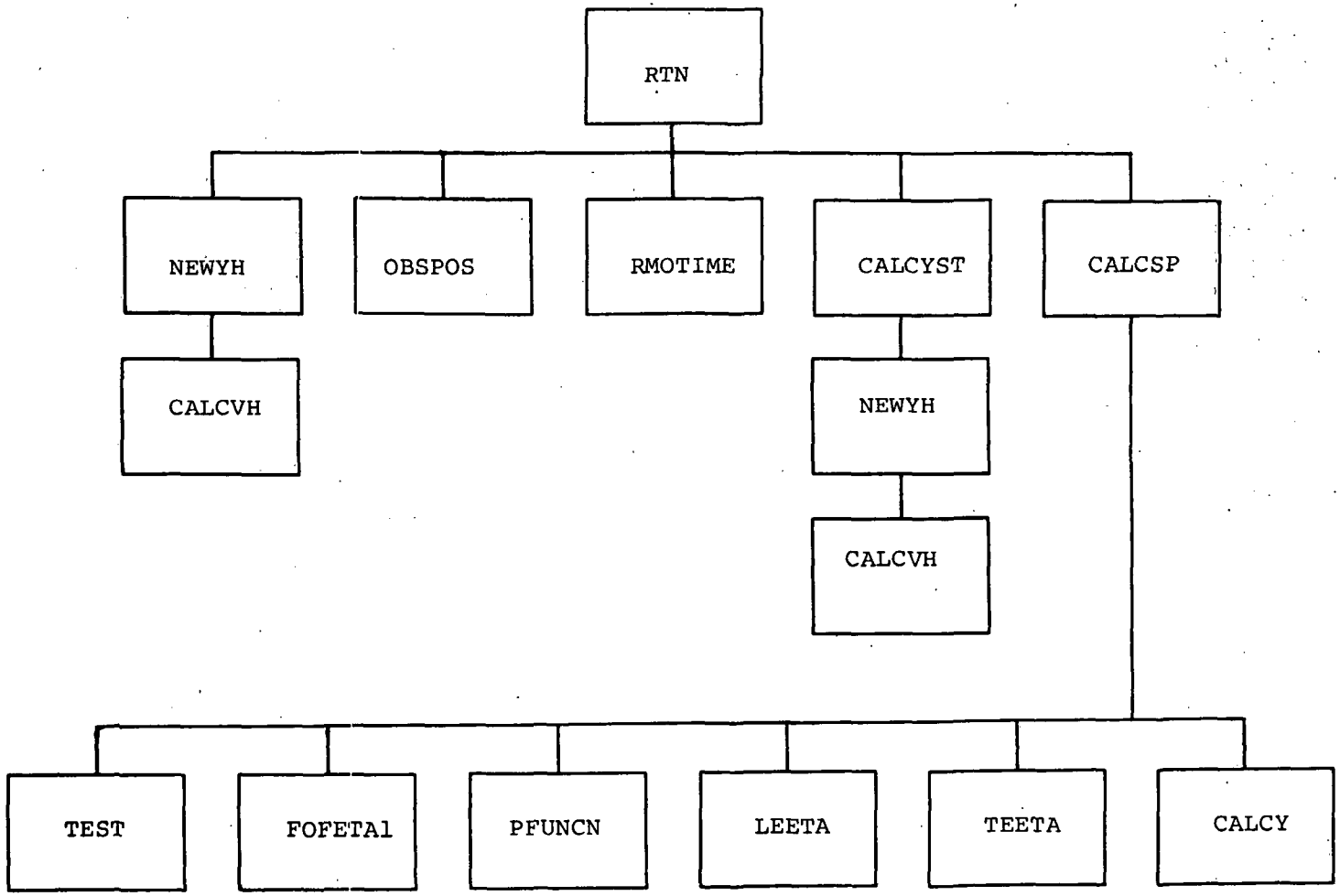


Figure 5.- Rotor Thickness Noise (RTN) subprogram hierarchy.

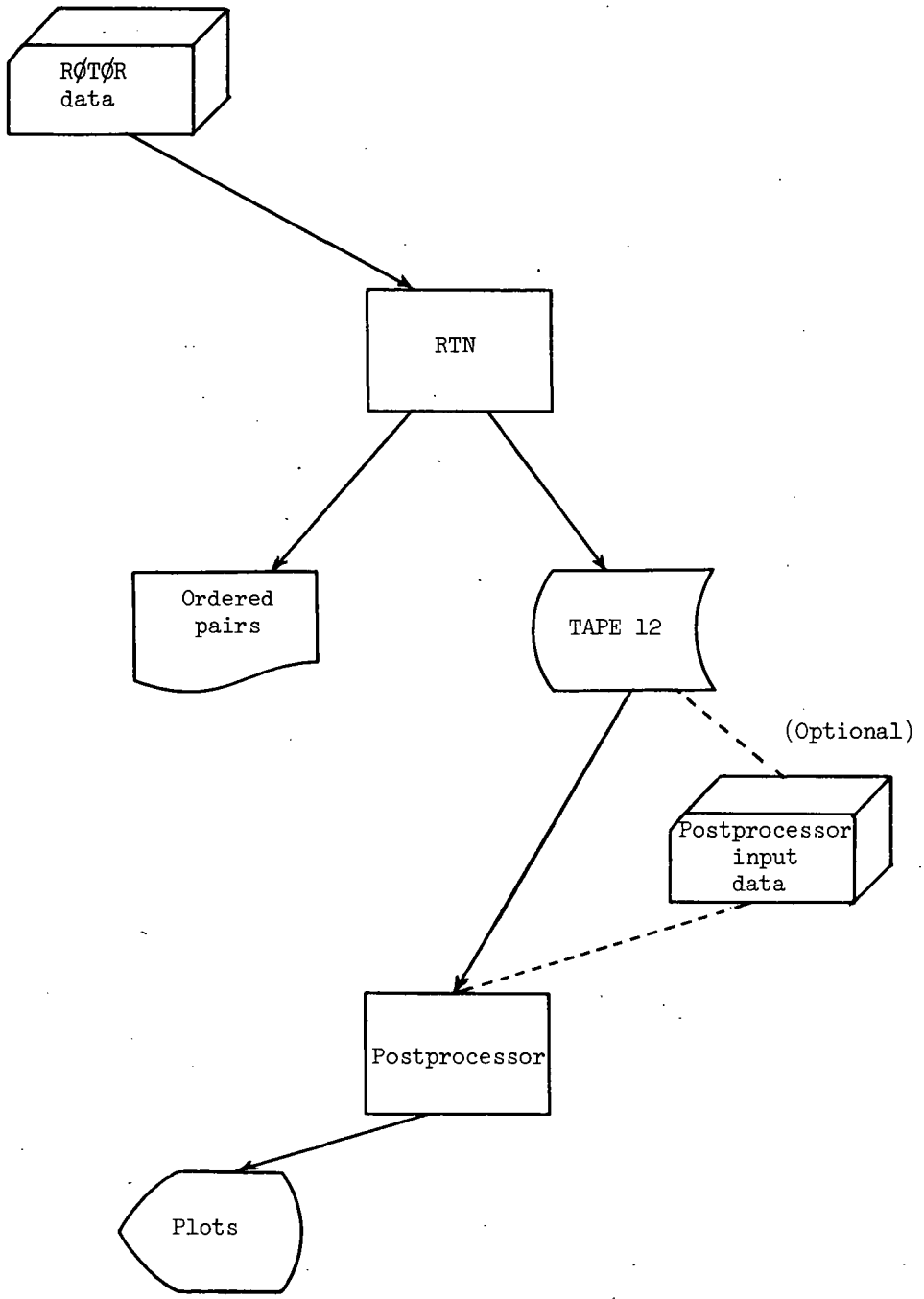


Figure 6.- RTN Postprocessor flow.

```
$ROTOR
XO       = .25E+03, 0.0, 0.0,
NBLADES  = 2,
R        = .671E+01,
CH       = .7E+00,
OMEGA    = .48E+03,
RO       = .61E+00,
THKRAT   = .8E-01,
DELTET   = .4E+00,
N        = 6,
NS       = 20,
KFLG     = 0,
ETAMAX   = 0.0,
BLNTN    = .1E+01,
JFLG     = 0,
COEFFS   = 0.0, .2E+01, -.2E+01, 0.0, 0.0,
TAUINT   = 0.0,
PERDM    = .106E+01,
DTAUM    = .2E+01,
SNDSPD   = .34E+03,
MORDATA  = F,
$END
```

1	0.	.755003247706419E+00	.368274186289126E-02
2	.277777777777778E-03	.755293152072209E+00	.376488349357262E-02
3	.555555555555556E-03	.755579312092344E+00	.412774739956806E-02
4	.833333333333332E-03	.755861726257422E+00	.423309736144570E-02
5	.111111111111110E-02	.756140393732565E+00	.462362007791356E-02
6	.138888888888888E-02	.756415314357586E+00	.465687047375710E-02
7	.166666666666665E-02	.756686488646952E+00	.500900726077658E-02
8	.194444444444442E-02	.756953917789762E+00	.504101714574137E-02
9	.222222222222219E-02	.757217603649526E+00	.544945062874086E-02
10	.249999999999996E-02	.757477548764001E+00	.534413020731811E-02
11	.277777777777773E-02	.757733756344756E+00	.565431454752224E-02
12	.305555555555549E-02	.757986233276764E+00	.559454617899044E-02
13	.333333333333326E-02	.758234975117880E+00	.600260255484447E-02
14	.361111111111102E-02	.758479996098231E+00	.602330162628115E-02
15	.388888888888879E-02	.758721299119451E+00	.652646018526923E-02
16	.416666666666654E-02	.758958890753917E+00	.661387664236740E-02
17	.444444444444431E-02	.759192778243825E+00	.678877483189216E-02
18	.472222222222207E-02	.759422969500225E+00	.726628270774723E-02
19	.499999999999984E-02	.759649473101881E+00	.712892068665186E-02
20	.527777777777760E-02	.759872298294145E+00	.762862808628620E-02
21	.555555555555537E-02	.760091454987670E+00	.785473094677686E-02
22	.583333333333313E-02	.760306953757027E+00	.813629285486817E-02
23	.611111111110990E-02	.760518805839279E+00	.874972874050439E-02
24	.638888888888867E-02	.760727023132397E+00	.855715233570625E-02
25	.666666666666643E-02	.760931618193645E+00	.935317605149660E-02
26	.694444444444420E-02	.761132604237847E+00	.961866129864003E-02
27	.722222222222196E-02	.761329995135512E+00	.975679084175196E-02

Figure 7.- Example of RTN output.

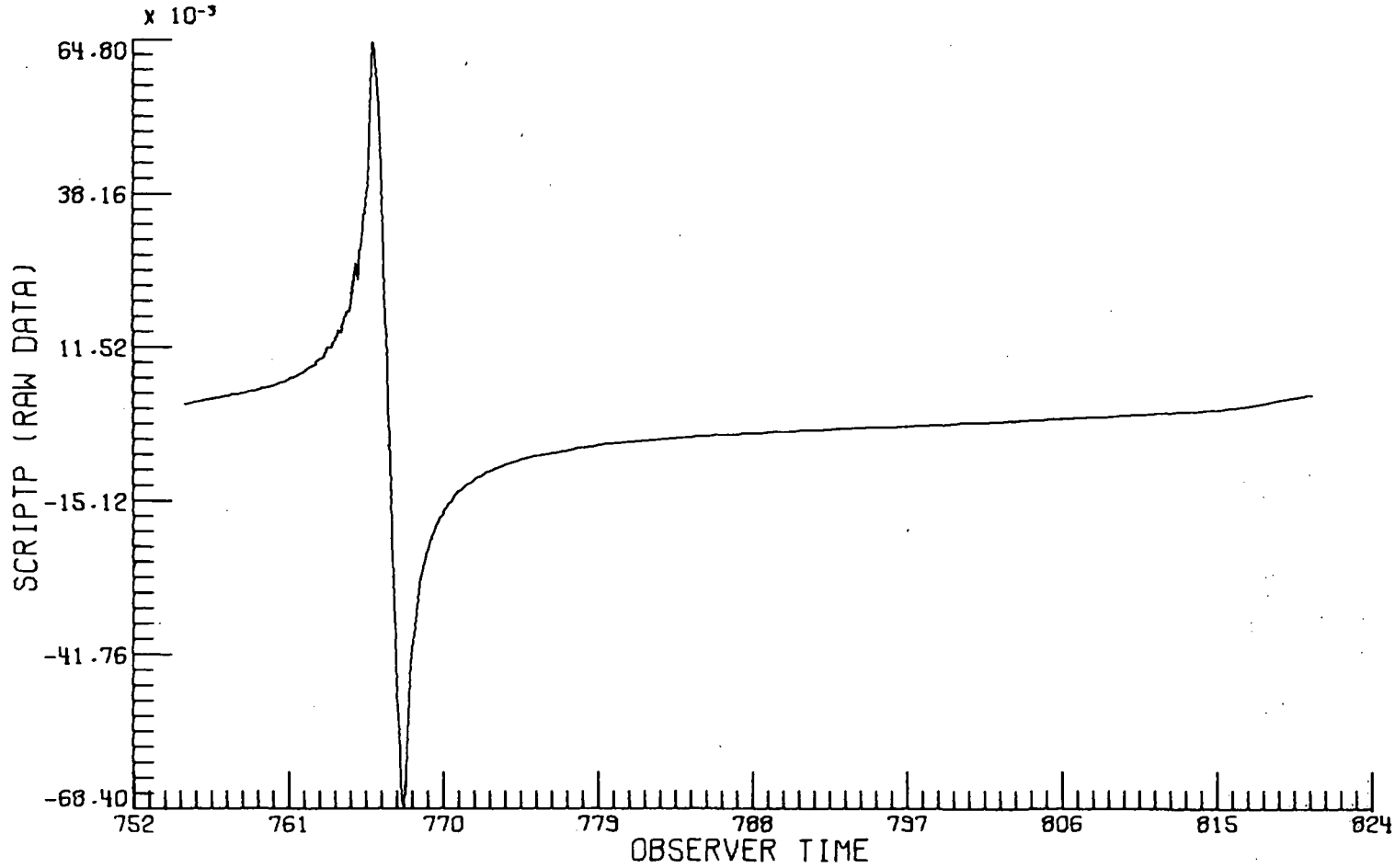


Figure 8.- Postprocessor graphic output of Rotor Thickness Noise output.

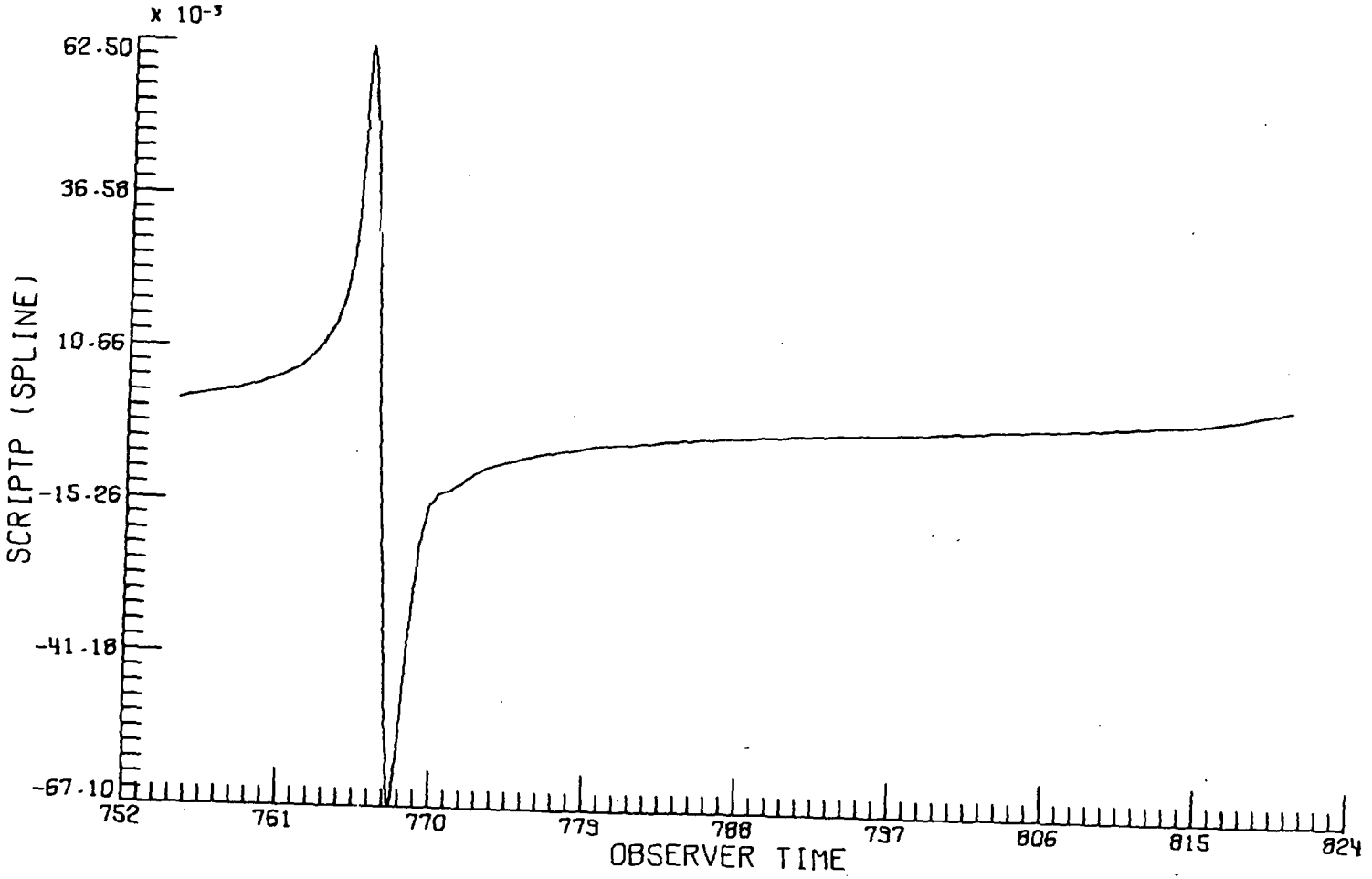


Figure 9.- Smoothed RTN output data using spline technique.

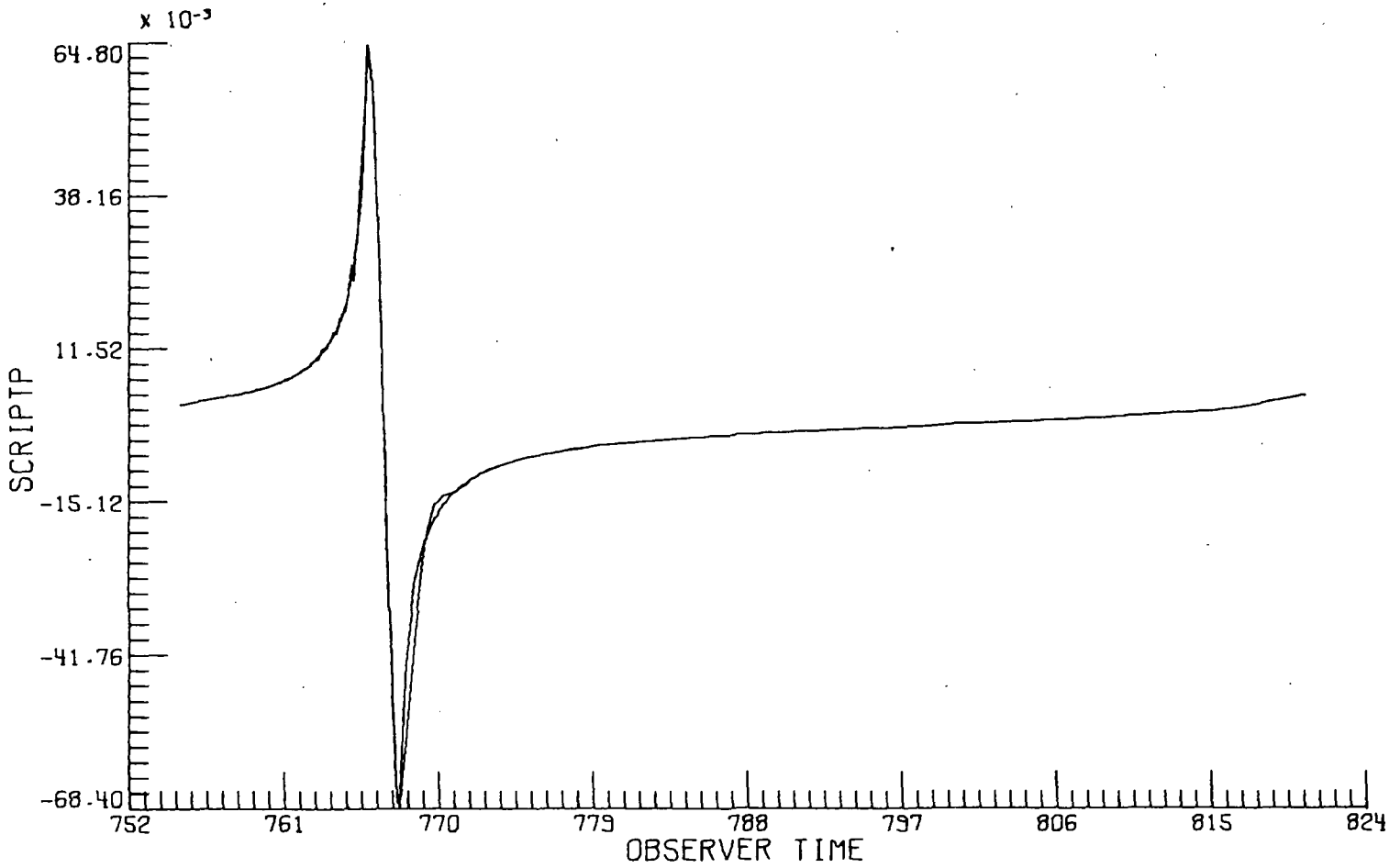


Figure 10.- Comparison of spline smoothed data with raw data.

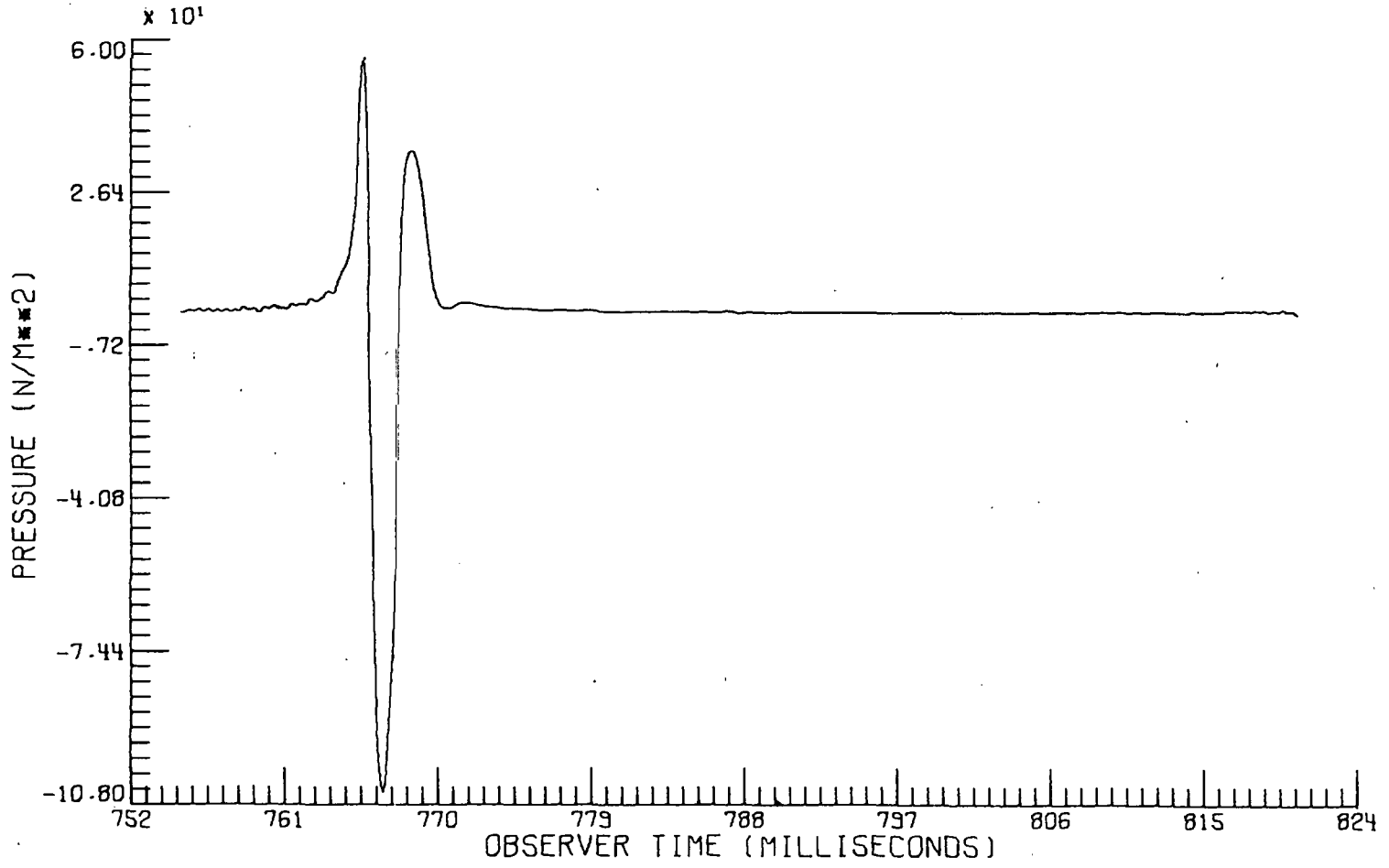


Figure 11.- Acoustic pressure caused by blade thickness for one period of blade system of example in figure 7.

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
WASHINGTON, D.C. 20546

OFFICIAL BUSINESS
PENALTY FOR PRIVATE USE $300

SPECIAL FOURTH-CLASS RATE BOOK

POSTAGE AND FEES PAID
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
451

U.S.MAIL

POSTMASTER: If Undeliverable (Section 158 Postal Manual) Do Not Return

"The aeronautical and space activities of the United States shall be conducted so as to contribute . . . to the expansion of human knowledge of phenomena in the atmosphere and space. The Administration shall provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof."

—NATIONAL AERONAUTICS AND SPACE ACT OF 1958

NASA SCIENTIFIC AND TECHNICAL PUBLICATIONS

TECHNICAL REPORTS: Scientific and technical information considered important, complete, and a lasting contribution to existing knowledge.

TECHNICAL NOTES: Information less broad in scope but nevertheless of importance as a contribution to existing knowledge.

TECHNICAL MEMORANDUMS: Information receiving limited distribution because of preliminary data, security classification, or other reasons. Also includes conference proceedings with either limited or unlimited distribution.

CONTRACTOR REPORTS: Scientific and technical information generated under a NASA contract or grant and considered an important contribution to existing knowledge.

TECHNICAL TRANSLATIONS: Information published in a foreign language considered to merit NASA distribution in English.

SPECIAL PUBLICATIONS: Information derived from or of value to NASA activities. Publications include final reports of major projects, monographs, data compilations, handbooks, sourcebooks, and special bibliographies.

TECHNOLOGY UTILIZATION PUBLICATIONS: Information on technology used by NASA that may be of particular interest in commercial and other non-aerospace applications. Publications include Tech Briefs, Technology Utilization Reports and Technology Surveys.

Details on the availability of these publications may be obtained from:

SCIENTIFIC AND TECHNICAL INFORMATION OFFICE

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Washington, D.C. 20546