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
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.
# CONTENTS

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUMMARY</td>
</tr>
<tr>
<td>INTRODUCTION</td>
</tr>
<tr>
<td>THE FORMULATION</td>
</tr>
<tr>
<td>THE PROGRAM</td>
</tr>
<tr>
<td>General Discussion</td>
</tr>
<tr>
<td>Method of Solution</td>
</tr>
<tr>
<td>Preparation of Input Data</td>
</tr>
<tr>
<td>Postprocessor Description</td>
</tr>
<tr>
<td>CONCLUDING REMARKS</td>
</tr>
<tr>
<td>APPENDIX A - ROTOR THICKNESS NOISE PROGRAM</td>
</tr>
<tr>
<td>Program RTN</td>
</tr>
<tr>
<td>Subroutine NEWYH</td>
</tr>
<tr>
<td>Subroutine CALC VH</td>
</tr>
<tr>
<td>Subroutine RMOTIME</td>
</tr>
<tr>
<td>Function FOF ETA1</td>
</tr>
<tr>
<td>Subroutine OBSPOS</td>
</tr>
<tr>
<td>Subroutine CALCYST</td>
</tr>
<tr>
<td>Subroutine CALCSP</td>
</tr>
<tr>
<td>Function PFUN CN</td>
</tr>
<tr>
<td>Subroutine LE ETA</td>
</tr>
<tr>
<td>Subroutine TEST</td>
</tr>
<tr>
<td>APPENDIX B - ROTOR THICKNESS NOISE POSTPROCESSOR PROGRAM</td>
</tr>
<tr>
<td>Program RTNPP</td>
</tr>
<tr>
<td>Subroutine PLOTPT</td>
</tr>
<tr>
<td>Subroutine HELP</td>
</tr>
<tr>
<td>Subroutine CSDS</td>
</tr>
<tr>
<td>Subroutine SPLDER</td>
</tr>
<tr>
<td>Subroutine PSEUDO</td>
</tr>
<tr>
<td>Subroutine INFOPLT</td>
</tr>
<tr>
<td>Subroutine CALPLT</td>
</tr>
<tr>
<td>REFERENCES</td>
</tr>
<tr>
<td>TABLE I</td>
</tr>
<tr>
<td>FIGURES</td>
</tr>
</tbody>
</table>
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 \( \tau \) 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)

\[
\frac{1}{c^2} \frac{\partial^2 p}{\partial \tau^2} - \nabla^2 p = \frac{\partial}{\partial \tau} \left[ \rho \nu_n |\nabla f| \delta(f) \right]
\]

where \( c \) is the speed of sound in the undisturbed medium, \( \delta(f) \) is the Dirac delta function, and \( \rho \) 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 \( \tilde{n}'- \) and \( \tilde{n}\)-frames in reference 5 are replaced by the \( \tilde{n}'- \) and \( \tilde{Y}_H\)-frames, respectively. Consider one single blade and a nonrotating coordinate system: the \( \tilde{Y}_H\)-frame is fixed to the center of rotation (fig. 1), and a rotating system is the \( \tilde{n}'\)-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:

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

where

- \( p(\vec{X},t) \) acoustic pressure
- \( \rho \) density of undisturbed medium
- \( 2 \)
\[ \mathbf{X}, t \] \hspace{1cm} \text{observer position and time}

\[ \tau_1, \tau_2 \] \hspace{1cm} \text{source times when the sphere}\ g = \tau - t + \frac{1}{c} \mathbf{X} - \mathbf{Y} = 0 \text{ enters and leaves the blade, respectively (fixed), } |\mathbf{X} - \mathbf{Y}| = r

\[ \mathcal{Y} \] \hspace{1cm} \text{source location } \mathcal{Y}-\text{frame}

\[(V_{H1}, V_{H2})\] \hspace{1cm} \text{vehicle velocity components along } Y_1 \text{ and } Y_2 \text{ axes of } \mathcal{Y}-\text{frame}

\[ \Omega \] \hspace{1cm} \text{angular velocity of rotor}

\[(\hat{r}_1, \hat{r}_2, \hat{r}_3)\] \hspace{1cm} \text{unit vector along radiation direction in } \hat{r}-\text{frame}

\[ \Gamma \] \hspace{1cm} \text{curve of intersection of sphere } g = 0 \text{ and mean surface of the blade}

D.P. \hspace{1cm} \text{disk plane}

\[ T_1 = \frac{\partial T}{\partial \eta_1}, \quad T_2 = \frac{\partial T}{\partial \eta_2} \]

\[ \tilde{V}_1 = -V_{H1} + \eta_2 \Omega \]

\[ \tilde{V}_2 = -V_{H2} - \eta_1 \Omega \]

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 \( \eta_2 \). 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 \([\Phi(\vec{X},t),t]\) where \(t\) is the observer time and \(\Phi(\vec{X},t)\) is defined by equation (3).

\[
\Phi(\vec{X},t) = \frac{\rho_0 c}{2\pi} \int_\Gamma \int_{\tau_1}^{\tau_2} \frac{\vec{V}_r F'(\eta_1)}{r \left[ 1 - \hat{\tau}_3^2 + F'2(\eta_1)(1 - \hat{\tau}_1^2) \right]^{1/2}} d\Gamma d\tau
\]  

(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 \((V_H)\) 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 \(Y_1 Y_2\)-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 \((X_0)\). 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 (PHIO). 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.
IREGION = 1 The point lies between the circles defined by the outer and inner radii of the blades, but is not on a blade.
IREGION = 3 The point lies inside the circle defined by the inner (hub) radius of the blades.
IREGION = 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 (DELTANU). 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°.

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

\[ F(E)/CH = T\left(C_1E + C_2E^2 + C_3E^3 + C_4E^4 + C_5E^5\right) \]

where \( E = \eta_1/CH \), \( \eta_1 \) 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 \( \eta_1 \). 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 \( \Phi(\mathbf{x},t) \). A useful formula to select DELTET is

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

where \( \psi \) 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

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

**ETAMAX**

The maximum \( \eta_1 \) 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:

\[
JFLG = 0 \quad \text{Hovering helicopter} \\
= 1 \quad \text{Helicopter in forward flight}
\]

**KFLG**

Airfoil section type indicator:

\[
KFLG = 0 \quad \text{Sharp leading edge} \\
= 1 \quad \text{Blunt leading edge}
\]

**MORDATA**

Additional data indicator:

\[
MORDATA = .T. \quad \text{Process another data set} \\
= .F. \quad \text{Terminate}
\]

**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 (\( \phi, t \)) 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 \( \phi \) 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

A

Read R0T0R data

Initialize COMMON variables

B

No

Helicopter moving

Yes

Determine helicopter location

Compute position of observer in helicopter reference frame

Compute observer time

Observer time decreasing

F

NEWYH

OBSRNS

RM0TIME

C
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

CALCYST termination flag=2

Yes

No

Correct Simpson sum

Output result

F

Increase time

E

More data

Yes

No

END

A

13
APPENDIX A
Listing of Program RTN

PROGRAM RTN(OUTPUT=131B, INPUT=131B, TAPE12=OUTPUT, TAPE5=INPUT)

C*************************************************************************
C
C Rotor Thickness Noise
C*************************************************************************
C
C
C
C PURPOSE:
PROGRAM RTN determines the acoustic pressure generated by a rotating thick body, such as an idealized rigid rotor, with some specified motion relative to an observer at some initial distance from the body. The present program includes routines developed solely for rotor systems with rectangular blades and uniform airfoil section across the span of the blades.

C
C METHOD OF SOLUTION:

THE FORM OF THE SOLUTION UTILIZED BY THIS PROGRAM INCLUDES A DOUBLE INTEGRAL WHERE ONE INTEGRATION IS OVER THE INTERSECTION OF THE CONTRACTING SPHERE \( C^* (\tau - \tau_0) + R = 0 \) (centered at observer) AND THE BODY, AND THE OTHER INTEGRATION IS OVER SOURCE TIME, \( \tau_0 \). THE FIRST OF THESE INTEGRALS IS EVALUATED USING A TRAPEZOIDAL INTEGRATION, THE SECOND INTEGRAL IS PERFORMED WITH SIMPSONS RULE AND A TRAPEZOIDAL END POINT CORRECTION WHERE REQUIRED.


TO DECREASE EXECUTION TIME, TWO SWEEP INCREMENTS ARE USED BY THE PROGRAM. FOR POINTS ON THE ARC WHICH ARE NOT ON A BLADE A LARGE ANGULAR INCREMENT IS USED TO SPACE ACROSS THESE REGIONS. SMALLER ANGULAR INCREMENTS ARE AUTOMATICALLY USED BY THE PROGRAM WHEN A BLADE IS ENCOUNTERED. FOR BLUNT EDGES A SMALLER INCREMENT MAY BE GENERATED FOR POINTS BETWEEN THE LEADING EDGE AND A MAXIMUM VALUE OF THE DISTANCE FROM THE LEADING EDGE. FOR EACH POINT ON A BLADE THE VALUE OF THE INTEGRAND AND DIFFERENTIAL INCREMENT ARE COMPUTED. THE LINE

***************
APPENDIX A

INTEGRAL (TRAPEZOIDAL) IS COMPUTED DYNAMICALLY BY CUMULATIVE SUMMING, WHEN A POINT ON THE ARC IS REACHED WHICH IS OUTSIDE THE CIRCLE DEFINED BY THE ROTOR SYSTEM, THE VALUE OF THE RESULTANT INTEGRAL IS MULTIPLIED BY THE APPROPRIATE PHYSICAL CONSTANT.

THE VALUE OF SOURCE TIME IS INCREASED SLIGHTLY. THE CENTER POSITION OF THE ROTOR SYSTEM RELATIVE TO THE OBSERVER IS RECOMPUTED. A NEW SPHERE IS CONSTRUCTED AND THIS LINE INTEGRAL IS CALCULATED. THIS PROCESS IS CONTINUED UNTIL THE PROJECTION OF THE SPHERE HAS COMPLETELY PASSED THROUGH THE ROTOR SYSTEM. THE FINAL VALUE OF SOURCE TIME REPRESENTS THE UPPER LIMIT ON THE TIME INTEGRAL. SINCE THE SOURCE TIME DIFFERENTIAL INCREMENT IS CONSTANT, A SIMPSON RULE INTEGRATION IS POSSIBLE. THIS INTEGRAL IS PERFORMED DYNAMICALLY AS A CUMULATIVE SUM. AS LINE INTEGRALS ARE CALCULATED THEY ARE MULTIPLIED BY TWO (FOUR) IF THEY ARE EVEN (ODD) TERMS AND ADDED TO THE SUM. IF THE LAST TERM IS EVEN, A TRAPEZOIDAL CORRECTION IS MADE. THE VALUE OF TAU IS INCREASED 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 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 DEGREES

ONCE ONE OF THE TERMINATION CRITERIA IS SATISFIED, THE PROGRAM TERMINATES UNLESS AN ADDITIONAL DATA SET IS SPECIFIED. NUMERICAL DIFFERENTIATION OF THE OUTPUT, AS WELL AS PLOTTING THE RESULTS AND FORMATTING THE RESULTS FOR SUBSEQUENT ANALYSIS IS PERFORMED IN A COMPANION POST-PROCESSOR.

PARAMETERS:

X0 THE LOCATION OF THE OBSERVER RELATIVE TO (0,0,0), THE INITIAL POSITION OF THE ROTOR SYSTEM AT TIME TAU = 0. (METERS)
NBLADES THE NUMBER OF BLADES IN THE ROTOR SYSTEM. THE PROGRAM ALLOWS A MAXIMUM OF 8 BLADES.
RADIUS OF THE BLADES. (METERS)
CH BLADE CHORD. (METERS)
OMEGA ANGULAR VELOCITY OF THE ROTOR SYSTEM. (REVOLUTIONS PER MINUTE)
RO HUB RADIUS. (METERS)
THKRAT BLADE THICKNESS RATIO.
APPENDIX A

C* DLTET ANGULAR DISPLACEMENT OF THE ROTOR SYSTEM BETWEEN
C* CONSECUTIVE POSITIONS OF THE CONTRACTING SPHERE.  * RTN 134
C* THIS VARIABLE IS USED WITH THE ROTOR SYSTEM ANGULAR
C* VELOCITY TO COMPUTE THE SOURCE TIME DIFFERENTIAL
C* INCREMENT, DELTAU. (DEGREES)  * RTN 135
C*  * RTN 136
C* N FACTOR TO CONTROL SPACING ALONG ARC BETWEEN BLADES
C* (I.E., TO SKIP FROM ONE BLADE TO THE NEXT).  * RTN 140
C* APPROXIMATELY N POINTS ARE USED TO TRAVERSE A
C* DISTANCE ALONG THE ARC EQUAL TO THE BLADE CHORD.  * RTN 141
C*  * RTN 142
C* NS FACTOR TO CONTROL SPACING ALONG ARC ON A BLADE,
C* APPROXIMATELY NS POINTS ARE USED TO TRAVERSE A
C* DISTANCE ALONG THE ARC EQUAL TO THE BLADE CHORD.
C* THIS VARIABLE DETERMINES THE LINE INTEGRAL
C* DIFFERENTIAL INCREMENTS.  * RTN 143
C*  * RTN 144
C* KFLG AIRFOIL SECTION TYPE INDICATOR.  * RTN 145
C* KFLG = 0: SHARP LEADING EDGE
C* = 1: BLUNT LEADING EDGE  * RTN 146
C*  * RTN 147
C* ETAMAX MAXIMUM ETA VALUE FOR WHICH INCREASED RESOLUTION
C* IS DESIRED NEAR THE LEADING EDGE FOR BLUNT LEADING
C* EDGES.  * RTN 148
C*  * RTN 149
C* BLNTN FACTOR TO GENERATE FINE SPACING NEAR THE LEADING
C* EDGE FOR BLUNT LEADING EDGES. FOR ETA VALUES
C* BETWEEN ZERO AND ETAMAX, THE SPACING FACTOR NS IS
C* MULTIPLIED BY BLNTN.  * RTN 150
C*  * RTN 151
C* JFLG ROTOR SYSTEM TRANSLATIONAL MOTION INDICATOR.  * RTN 152
C* JFLG = 0: HOVERING HELICOPTER.  * RTN 153
C* = 1: HELICOPTER IN FORWARD FLIGHT.  * RTN 154
C*  * RTN 155
C* COFFS COEFFICIENTS IN THE FUNCTION DESCRIBING THE AIRFOIL
C* SECTION. A MINUS SIGN MUST APPEAR EXPLICITLY FOR
C* NEGATIVE TERMS.  * RTN 156
C*  * RTN 157
C* TAUINT THE INITIAL VALUE OF SOURCE TIME (TAU). (SECONDS)  * RTN 158
C*  * RTN 159
C* PEROM THE NUMBER OF PERIODS OF THE ROTOR SYSTEM TO BE
C* PROCESSED. THIS VARIABLE IS USED TO ESTABLISH
C* THE MAXIMUM TIME TERMINATION CRITERION.  * RTN 160
C*  * RTN 161
C* DTAM TIME_INCREMENT BETWEEN DATA POINTS. THE SOURCE
C* TIME_DIFFERENTIAL_INCREMENT DERIVED FROM DLTET
C* IS MULTIPLIED BY DTAM TO DETERMINE THIS_INCREMENT.  * RTN 162
C*  * RTN 163
C* SNOMSD SPEED OF SOUND. (METERS/SEC)  * RTN 164
C*  * RTN 165
C* MORA A LOGICAL VARIABLE TO SPECIFY IF AN ADDITIONAL
C* DATA SET IS TO BE PROCESSED.  * RTN 166
C*  * RTN 167
C* REQUIRED ROUTINES
C* SUBROUTINE NEWWH
C* SUBROUTINE CALCVH (SEE ADDITIONAL DETAILS)
C* SUBROUTINE OJSPOS
C* SUBROUTINE RMOTIME
C* SUBROUTINE CALYST
C* SUBROUTINE CALCSP
C* FUNCTION FOFETAI (SEE ADDITIONAL DETAILS)

16
SUBROUTINE TEST (SEE ADDITIONAL DETAILS)
SUBROUTINE LEEETA

FOR A MOVING ROTOR SYSTEM (JFLG = 1), A SUBROUTINE MUST BE PROVIDED TO DESCRIBE THE MOTION.

SUBROUTINE CALCVH(VH,TIME)

SUBROUTINE CALCVH DEFINES THE VECTOR VH(3) WHICH MUST CONTAIN THE THREE COMPONENTS OF THE ROTOR SYSTEM TRANSLATIONAL VELOCITY AT T = TIME. THESE COMPONENTS ARE TO BE SPECIFIED IN THE ROTOR SYSTEM REFERENCE FRAME. THE THREE AXES IN THIS COORDINATE SYSTEM ARE DEFINED AS FOLLOWS.

THE Y1 AXIS IS PARALLEL TO THE INITIAL ORIENTATION OF BLADE NUMBER ONE WITH THE DIRECTION FROM THE BLADE TIP TO THE CENTER OF THE ROTOR SYSTEM TAKEN AS POSITIVE.

THE Y2 AXIS IS PERPENDICULAR TO Y1 PASSING THROUGH THE CENTER OF THE ROTOR SYSTEM AND LYING IN THE ROTOR SYSTEM PLANE. THE DIRECTION TO THE LEFT OF AN OBSERVER FACING TOWARD POSITIVE Y1 IS TAKEN AS POSITIVE.

THE Y3 AXIS IS PERPENDICULAR TO THE PLANE OF THE ROTOR SYSTEM WITH THE POSITIVE DIRECTION DETERMINED BY THE USUAL CONVENTION FOR A RIGHT-HANDED COORDINATE SYSTEM.

THE PRESENT VERSION OF FUNCTION FOFETA1 IMPOSES A LIMITATION TO UNIFORM AIRFOIL SECTION ACROSS THE SPAN OF THE BLADES. FOR NONUNIFORM AIRFOIL SECTION THIS FUNCTION MUST BE MODIFIED.

THE PRESENT VERSION OF SUBROUTINE TEST IMPOSES A LIMITATION TO ROTOR SYSTEMS WITH RECTANGULAR BLADES. FOR OTHER BLADE SHAPES THIS SUBROUTINE MUST BE MODIFIED.

SOURCE: RTN WAS PROGRAMMED AT LANGLEY RESEARCH CENTER BY COMPUTER SCIENCES CORPORATION BASED UPON A DESIGN BY F. FAFASSAT.

LANGUAGE: FORTRAN

DATE RELEASED: MAY 1, 1974

LATEST REVISION: DECEMBER 16, 1975

17
APPENDIX A

LOGICAL MORDATA

DIMENSION BETA(8), COEFFS(5), ETA(2,8), YH(3),
X(3), X(3), Y(3),
YEND(2), YH(3), YSTART(2),
VH(3)

COMMON ALFA, BETA, BLADES, BLNTN, CBETA,
MACH, COEFFS, DELTAU, ETA, ETA2,
IC, FAC, FN, FPS, FETA1,
IFLAG, IREGION, J2, JFLG, KFLG,
NBLADES, OMEGAR, OTIME, PI, R,
PM, PMIN, S, SBETA, SNOSPO,
SRCOST, SWPCST, SWPTAU, TAU, THKRAT,
TINTH, VH, Y(3), YSTART(2),
VH, YSTART, DELTA1, DELF

NAMELIST/ROTOR/XO,N8LADES,R,CH,OMEGA,P.
THKRAT,OELTET,N,NS,KFLG,
1 ETAMAX,BLNTN,JFLG,COEFFS,TAUINT,PERDM,OTAUM,SNOSPO,MORDATA

READ NAMELIST DATA

10 READ(5,ROTOR)
WRITE(12,ROTOR)

INITIALIZE VARIABLES

PI = 3.14159265358979
OMEGAR = 2.*OMEGA*PI/6C.
TESTIM = 0.
YH(1) = 0.
YH(2) = 0.
YH(3) = 0.
Y(1) = 0.
Y(2) = 0.
Y(3) = 0.
VH(1) = 0.
VH(2) = 0.
VH(3) = 0.
ALFA = ASIN(C.5*CH/R)
PMIN = R
RSQR=R*R
POSG=RO*RO
TAU = TAUINT
IF(KFLG.EQ.0) BLNTN = 1.
NPTS=0
PKEP1 = 0.
PKEP2 = 0.
SCRIPTP = 0.
SCRIPTP = 0.
BLADES = FLOAT(NBLADES)

COMPUTE SOURCE TIME DIFFERENTIAL INCREMENT AND MAXIMUM TIME

DELTAU = DELTET/(6.*OMEGA)
PERIOD = 6C./ (OMEGA*8LADES)
TAUMAX = TAUINT + PERDM*PERIOD

S. N. 2C BEGINS A NEW DATA POINT

20 IF(JFLG.NE.0) CALL NEWYH(TAU)

DETERMINE RELATIVE LOCATIONS OF ROTOR SYSTEM AND OBSERVER

CALL OBSPOS

CALL RMOTIME

DTIM = OTIME - TESTIM
APPENDIX A

IF OBSERVER TIME IS DECREASING SKIP TO NEXT DATA POINT

IF (DTIM.LE.0.) GO TO 40
TESTIM = OTIME
QPSANGL = ASIN(X(3)/SQRT(AX(1)**2 + X(2)**2 + X(3)**2))

IF ANY TERMINATION CRITERION IS SATISFIED SKIP TO THE CHECK FOR ADDITIONAL DATA

IF (QPSANGL.GT. (PI/4.) GO TO 60
IF (RM.LT.RMINI) GO TO 6J
IF (TAU.GT.TAUMAX) GO TO 6C

INITIALIZE SWEEP VARIABLES

SWPTAU = TAU
TIMTHRU = TAU
IEVEN = -1

S. N. 1C BEGINS A NEW POSITION OF THE SPHERE

3C TIMTHRP = TIMTHRU + 1.
IEVEN = -IEVEN
FN = N
FNS = NS
CALL CALCYST(RSQ,PHIO)
PHI = PHIC

AND TO CUMULATIVE SIMPSON SUM

PKEP2 = PKEP1
PKEP1 = SCRIP1
PDELT = 2.*SCRIP1
IF (IEVEN.GT.0) PDELT = 2.*PDELT
IF (TIMTHRU.LT.2.5) PDELT = SCRIP1
SCRIPTP = SCRIPTP + PDELT

IF IFLAG=4, WE ARE IN THE ROTOR SYSTEM

IF (IFLAG.EQ.4) GO TO 50

IF IFLAG=2, A TERMINATION CRITERION WAS DETECTED IN CALCYST

IF (IFLAG.EQ.2) GO TO 60

OTHERWISE, WE HAVE COMPLETED THIS DATA POINT, CORRECT THE SIMPSON INTEGRATION AND OUTPUT THE RESULTS

PDELT = -PKEP1
IF (IEVEN.GT.0) PDELT = (PKEP2 - 5.*PKEP1)/2.
SCRIPTP = SCRIPTP + PDELT
SCRIPTP = SCRIPTP/3.
NPTS = NPTS + 1
P = SCRIPTP*415./(2.*PI)
WRITE (12,70) NPTS, TAU, OTIME, P
SCRIP1 = 0.
PKEP1 = 0.
PKEP2 = 0.
SCRIPTP = 0.

INCREASE THE TIME AND BEGIN A NEW DATA POINT

TAU = TAU + OTAUM*DELTAS
GO TO 20

WE ARE IN THE ROTOR SYSTEM, SWEEP THE ARC

RTN 338
RTN 339
RTN 340
RTN 341
RTN 342
RTN 343
RTN 344
RTN 345
RTN 346
RTN 347
RTN 348
RTN 349
RTN 350
RTN 351
RTN 352
RTN 353
RTN 354
RTN 355
RTN 356
RTN 357
RTN 358
RTN 359
RTN 360
RTN 361
RTN 362
RTN 363
RTN 364
RTN 365
RTN 366
RTN 367
RTN 368
RTN 369
RTN 370
RTN 371
RTN 372
RTN 373
RTN 374
RTN 375
RTN 376
RTN 377
RTN 378
RTN 379
RTN 380
RTN 381
RTN 382
RTN 383
RTN 384
RTN 385
RTN 386
RTN 387
RTN 388
RTN 389
RTN 390
RTN 391
RTN 392
RTN 393
RTN 394
RTN 395
RTN 396
RTN 397
RTN 398
RTN 399
RTN 400
RTN 401
RTN 402
RTN 403
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
C* SUBROUTINE NEWYH IS CALLED BY PROGRAM RTN AND SUBROUTINE CALCYST. THIS ROUTINE RETURNS YH(3), THE THREE
C* COORDINATES OF THE CENTER OF THE ROTOR SYSTEM RELATIVE TO ITS INITIAL LOCATION (0,0,0). SUBROUTINE NEWYH IS
C* CALLED ONLY FOR MOVING (TRANSLATIONAL) ROTOR SYSTEMS TO DETERMINE THE POSITION AT THE TIME SPECIFIED BY THE
C* VARIABLE TIME (M/SEC). THE ROTOR SYSTEM VELOCITY (POSSIBLY TIME-DEPENDENT) IS OBTAINED FROM SUBROUTINE CALCVH.
C*
DIMENSION ETA(2,8), X(3), YENO(2), COMMON VΗ(3),
1 CALL CALCVH(TIME)
2 DO 10 I=1,3
3 YH(I) = VH(I)*TIME
4 CONTINUE
5 RETURN
6 END

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 Y1-axis.

Flow Chart of Subroutine CALCVH

Define helicopter velocity components

Return
APPENDIX A

Listing of Subroutine CALCVH

Subroutine CALCVH

SUBROUTINE CALCVH(TIME)

SUBROUTINE CALCVH IS CALLED BY SUBROUTINE NEWYH. CALCVH

RETURNS VH(3), THE THREE COMPONENTS OF THE VELOCITY

MAY BE USED TO SPECIFY A TIME-DEPENDENT VELOCITY.

DIMENSION

VH(3) = 61.7

RETURN

END

Called by

SUBROUTINE NEWYH

Three components

of the velocity

used to specify a time-dependent velocity.

CALCVH

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

1. Initialize blade counter to 1, TDST to zero
2. Compute square of range to each blade tip
3. Either value greater than TDST
   - No
   - Yes: Set TDST to the maximum of these values
4. Increment blade counter
5. All blades processed
   - No
   - Yes: Compute maximum range and observer time
6. Return
APPENDIX A

Listing of Subroutine RMOTIME

SUBROUTINE RMOTIME

C***********************************************************************
C* SUBROUTINE RMOTIME IS CALLED BY PROGRAM RTN. THIS ROUTINE
C* DETERMINES THE RANGE TO THE MOST DISTANT TIP IN THE POTOR
C* SYSTEM, WHICH IS USED AS THE MAXIMUM SWEPTING DISTANCE
C* FOR A GIVEN TAU. THE OBSERVERS TIME (CORRESPONDING TO TAU)
C* IS ALSO RETURNED.
C***********************************************************************

C DIMENSION NFTA(8), COEFFS(5), ETA(2,8),
1 X(3), X(3),
2 YEND(2), YH(3), YSTART(2),
3 V(3)

COMMON ALFA, BETA, NBLADES, BLTN, CBETA,
1 CH, COEFFS, DELTAU, ETA, ETA2,
2 ETRNAX, FOC, FN, FNS, FETA1,
3 IFLAG, IREGION, J2, JFLG, KFLG,
4 NLADENS, OMEGAR, OTIME, PI, R,
5 RM, RMN, S, SBETA, SNDSPD,
6 SRCOST, SWPOST, SWPTAU, TAU, THKRAT,
7 TIMTHRU, VH, X, XG, YEND,
8 YH, YSTART, DELETA, DELF

DIST(A,B,C,D,E) = (A - D*COS(E))**2 + (B - D*SIN(E))**2 + C*C

OD 10 I=1,NBLADES

OD TIP1 = OMEGAR*TAU

OD 10 TIP1 = TIP1 + 2.*ALFA

OD DI = DIST(XT1),X(2),X(3),R,TIP1

OD IF(D1.GE.TOST) TOST = DI

OD IF(D1.GE.TOST) TOST = DI

OD CONTINUE

OD RM = SORT(TOST)

OD TIME = TAU + RM/SNDSPD

OD RETURN

OD END

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 FOFETA1

[Diagram showing the flow chart of Function FOFETA1]

Return
APPENDIX A

Listing of Function FOFETAl

```plaintext
FUNCTION FOFETAl(XETA)

C** FUNCTION FOFETAl COMPUTES THE BLADE THICKNESS FOR SPECIFIED
C** DISTANCE ALONG THE BLADE CHORD
C*
C** DIMENSION BETA(8), COEFFS(5), ETA(2,8),
C 1 X(3), X0(3),
C 2 YEND(2), YH(3), YSTART(2),
C 3 VH(3)
C COMMON ALFA, BETA, BLADES, BLTN, CBETA,
1 CH, COEFFS, DELTAU, ETA, ETA2,
2 E TMAX, FAC, FN, FNS, FETA1,
3 IFLAG, IREGION, J2, JFLG, KFLG,
4 NOLADES, OMEGAR, OTIME, PI, R,
5 RM, RMIN, S, SRETA, SNOSPD,
6 SRCOST, SWPOST, SWPTAU, TAU, THKRAT,
7 TIMTHRU, VH, X, X0, YEND,
8 YH, YSTART, DELTA1, DELF
9 E = XETA/CH
10 A1 = COEFFS(1)*E + COEFFS(4)
11 A1 = A1*E + COEFFS(3)
12 A1 = (A1*E + COEFFS(2))*E
13 FOFETAl = (A + A1)*THKRAT
14 RETURN
END
```

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

```
Compute position of observer in reference frame

Return
```

25
APPENDIX A

Listing of Subroutine OBSPOS

```plaintext
SUBROUTINE OBSPOS

C* SUBROUTINE OBSPOS IS CALLED BY PROGRAM RTN AND BY SUBROUTINE
C* CALCYST. THIS ROUTINE RETURNS X(2), THE THREE
C* COORDINATES OF THE OBSERVER IN THE CURRENT ROTOR SYSTEM
C* REFERENCE FRAME.

DIMENSION BF(TAI9I, 1, 2, 7), COM10N(1, 2, 3, It, 5, 6, 7, f, X(3I), X(1)=XO, X(2)=XO, X(3)=XG
RFTURN

END
```

Subroutine CALCYST

Subroutine CALCYST is called by RTN. A second entry point, CALCY, is used by subroutine CALCSNP. This subprogram calculates the points of intersection of the projection of the sphere on the rotor system plane with the circle(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

ENTRY
CALCYST

Set
termination
flag to 2

Contract
the
sphere

No

Helicopter
moving

Yes

Get
helicopter
coordinates
for this
sweep time

NEWYH

Get position
of observer
in helicopter
reference
frame

ØBSPoS

Satisfy
termination
criteria

No

A

Yes

Return
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, PHIC)

**CALYST** 2
**CALYST** 3
**CALYST** 4
**CALYST** 5
**CALYST** 6
**CALYST** 7
**CALYST** 8
**CALYST** 9
**CALYST** 10
**CALYST** 11
**CALYST** 12
**CALYST** 13
**CALYST** 14
**CALYST** 15
**CALYST** 16
**CALYST** 17
**CALYST** 18
**CALYST** 19
**CALYST** 20
**CALYST** 21
**CALYST** 22
**CALYST** 23
**CALYST** 24
**CALYST** 25
**CALYST** 26
**CALYST** 27
**CALYST** 28
**CALYST** 29
**CALYST** 30
**CALYST** 31
**CALYST** 32
**CALYST** 33
**CALYST** 34
**CALYST** 35
**CALYST** 36
**CALYST** 37
**CALYST** 38
**CALYST** 39
**CALYST** 40
**CALYST** 41
**CALYST** 42
**CALYST** 43
**CALYST** 44
**CALYST** 45
**CALYST** 46
**CALYST** 47
**CALYST** 48
**CALYST** 49
**CALYST** 50
**CALYST** 51
**CALYST** 52
**CALYST** 53
**CALYST** 54
**CALYST** 55
**CALYST** 56
**CALYST** 57
**CALYST** 58
**CALYST** 59
**CALYST** 60
**CALYST** 61
**CALYST** 62
**CALYST** 63
**CALYST** 64
**CALYST** 65
**CALYST** 66
**CALYST** 67
**CALYST** 68
**CALYST** 69
**CALYST** 70
**CALYST** 71
**CALYST** 72
**CALYST** 73
**CALYST** 74
**CALYST** 75

**C**
**CALYST** 2)
**CALYST** 3)
**CALYST** 4)
**CALYST** 5)
**CALYST** 6)
**CALYST** 7)
**CALYST** 8)
**CALYST** 9)
**CALYST** 10)
**CALYST** 11)
**CALYST** 12)
**CALYST** 13)
**CALYST** 14)
**CALYST** 15)
**CALYST** 16)
**CALYST** 17)
**CALYST** 18)
**CALYST** 19)
**CALYST** 20)
**CALYST** 21)
**CALYST** 22)
**CALYST** 23)
**CALYST** 24)
**CALYST** 25)
**CALYST** 26)
**CALYST** 27)
**CALYST** 28)
**CALYST** 29)
**CALYST** 30)
**CALYST** 31)
**CALYST** 32)
**CALYST** 33)
**CALYST** 34)
**CALYST** 35)
**CALYST** 36)
**CALYST** 37)
**CALYST** 38)
**CALYST** 39)
**CALYST** 40)
**CALYST** 41)
**CALYST** 42)
**CALYST** 43)
**CALYST** 44)
**CALYST** 45)
**CALYST** 46)
**CALYST** 47)
**CALYST** 48)
**CALYST** 49)
**CALYST** 50)
**CALYST** 51)
**CALYST** 52)
**CALYST** 53)
**CALYST** 54)
**CALYST** 55)
**CALYST** 56)
**CALYST** 57)
**CALYST** 58)
**CALYST** 59)
**CALYST** 60)
**CALYST** 61)
**CALYST** 62)
**CALYST** 63)
**CALYST** 64)
**CALYST** 65)
**CALYST** 66)
**CALYST** 67)
**CALYST** 68)
**CALYST** 69)
**CALYST** 70)
**CALYST** 71)
**CALYST** 72)
**CALYST** 73)
**CALYST** 74)
**CALYST** 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

- Set JREGION = 0
- Determine region point is in IREGION
- If IREGION = 1, go to C
- Set Δ REG = IREGION - JREGION
- If ΔREG = 0, go to C
- If ΔREG = 6, go to D
- If ΔREG = 5, go to I
- Otherwise, go to B
APPENDIX A

If \( \Delta \text{REG} = 3 \) then Yes, go to J

If \( \Delta \text{REG} = -1 \) then Yes, go to P

Select large sweep angle

Get coordinates of new point

Set \( J\text{REGION} = I\text{REGION} \)

Determine which region point is in \( I\text{REGION} \)

TEST

A

Not on blade
APPENDIX A

Use intersection point from CALCYST to begin sweep

Entering the blade from outside the rotor circle

Select small sweep angle

Entering the blade from another region

Blunt edge

Decrease sweep angle

Determine coordinates of new point

Near leading edge

Yes

No

Set JREGION \(=\) IREGION

G
APPENDIX A

Determine which region point is in IREGI0N

Set
\[ \Delta R E G = I R E G I 0 N - J R E G I 0 N \]

\[ \Delta R E G = 0 \]

Yes \rightarrow H

No

\[ \Delta R E G = -5 \]

Yes \rightarrow K

No

\[ \Delta R E G = -6 \]

Yes \rightarrow L

No \rightarrow N
APPENDIX A

H

Increment point counter

Determine blade thickness

Compute integrand

Accumulate trapezoidal sum

\[ \text{INC} \]

Yes

No

Complete trapezoidal integration

\[ \text{IREGION} = 0 \]

Yes

On a blade

F

Reached other edge of blade

PFUNCN

F\(\Phi\)ETAL

A

P
APPENDIX A

I

Yes

\[ \eta_1 > \frac{1}{2} \text{ chord} \]

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 \( \eta_1 \) and \( \eta_2 \)
APPENDIX A

Get point of intersection with leading edge

Get point of intersection with trailing edge

计算 $\eta_1$

$\eta_1 > \frac{1}{2}$ chord

Yes

No

Get points of intersection of sphere and rotor circle

[Leaving rotor system]

Leaving blade, but still in rotor system

$\eta_1 < 0$

Yes

No

Compute $\eta_2$

H

K

LEETA

TEETA

CALCY
APPENDIX A

1. Get intersection of sphere with hub radius
2. Entering hub region from blade
3. CALCY
4. M
5. Passed out of rotor system
6. P
7. Compute resultant integral
8. Return
APPENDIX A

Listing of Subroutine CALCSP

SUBROUTINE CALCSP (RSQ, POSQ, PHI, Y, SCRIPTP)

SUBROUTINE CALCSP is called by program RTN. This routine controls the sweeping (counterclockwise) across the arc defined by a fixed position of the contracting sphere. Two sweeping increments are used: DELPHI2 for sweeping across a rotor blade and OELPHI1 for the space between blades, as determined by subroutine TEST. The line integral representing the contribution of each blade is a trapezoidal integration, points on the edge of any blade are obtained from subroutines LEFTA and TEETA.

DIMENSION BETA(8), COEFFS(5), ETA(2,8), X(3), Y(3), YEND(2), YH(3), YLST(3), YSTARK2), VH(3)
COMMON ALFA, BETA, BLADES, BLTN, CRETA, E
1 CH, COEFS, DELTAU, ETA, ETA2, CALCS
2 ETAMAX, FAC, FN, FNS, FETA1, CALCS
3 IFLAG, IREGION, J2, JFLG, KFLG, CALCS
4 NBLADES, OMEGAR, O TIME, PI, R, CALCS
5 SRCOST, SWPDST, SWPTAU, TAU, THKRAT, CALCS
6 TINTHRU, VH, X, XO, YEND, CALCS
7 Y, YSTART, DELETA1, DELF

C INITIALIZE VARIABLES FOR THIS SHEEP

DELPHI1 = CH/(FN*SRCOST)
DELPHI2 = CH/(FNS*SRCOST)
DELETAl = 0.
DELF = 0.
YLIST(3) = c.
REGION = 0
SUMSP = 0.
OGLST = 0.
SCRIPTP = c.
J = 1
J2 = 1

C DETERMINE WHICH REGION WE ARE IN

CALL TEST(YSTART, RSQ, POSQ, SWPTAU)
IF(REGION.EQ.1) GO TO 20
10 IDELREG = (REGION - JREGION)
IF(DELREG.EQ.0) GO TO 20
IF(DELREG.EQ.6) GO TO 30
IF(DELREG.EQ.5) GO TO 90
IF(DELREG.EQ.3) GO TO 130
IF(DELREG.EQ.-11) GO TO 80

C NOT ON A BLADE. SELECT LARGE ANGLE AND SEARCH THE ARC

C DELPHI = DELPHI1
PHI = PHI + FAC*DELPHI
Y(1) = X(1) + SRCOST*COS(PHI)
Y(2) = X(2) + SRCOST*SIN(PHI)
REGION = IREGION
CALL TEST(Y, RSQ, POSQ, SWPTAU)
GO TO 19

C ENTERING REGION 6 FROM REGION 5. USE THE INTERSECTION DATA

30 Y(1) = YSTART(1)
Y(2) = YSTART(2)
APPENDIX A

ENTERING REGION 6 FROM ANOTHER REGION. SELECT SMALL ANGLE AND PROCESS THE BLADE.

IF ETA1(J1, J2) < ETAMAX) DELPHI = DELPHI / BLNTN
FETA1 = FOFETA1(ETA(1, J2))
50 JREGION = IREGION

INCREASE THE SWEEP ANGLE AND TEST THE NEXT POINT

PHI' = PHI + FAC * DELPHI
FETA11 = ETA(1, J2) - ETA11
DELETA2 = ETA(2, J2) - ETA22
SAVEF = DELF
YLST(1) = Y(1)
YLST(2) = Y(2)
Y(1) = Y(1) + SRCST * COS(PHI)
Y(2) = Y(2) + SRCST * SIN(PHI)
CALL TEST(Y, RSO, ROSQ, SWPTAU)
IDELREG = IREGION - JREGION
IF (IDELREG = 0) GO TO 6C
IF (IDELREG = -5) GO TO 1C
IF (IDELREG = -6) GO TO 170
GO TO IflC

ON A BLADE

60 J = J + 1
DELETA1 = ETA(1, J2) - ETA11
DELETA2 = ETA(2, J2) - ETA22
FETA1 = FOFETA1(ETA(1, J2))
DEL = FETA1 - FETA11
DELGAM = SQRT(DELF**2 + DELETA1**2 + DELETA2**2)
IF (DELETA1 < 0) DELGAM = - DELGAM
SP = PFUNCN(ETA22, DELF, DELETA1, YLST)
SUMSP = SUMSP + SP * DGLST
DGLST = DELGAM
IF (IDELREG = 0) GO TO 50

JUST LEFT REGION 6. PERFORM THE TRAPEZOIDAL INTEGRATION

SP = PFUNCN(ETA(2, J2), DELF, DELETA1, Y)
SUMSP = SUMSP + SP * DELGAM
IF (IDELREG = 0) GO TO 80
DGLST = 0
DELETA1 = 0
DELF = 0
J = 1
J2 = 1
GO TO 20

PASSED OUT OF THE ROTOR SYSTEM FOR THIS ARC

SCRIPTP = 0.5 * DELTAU * SUMSP / SWPOST
RETURN

ENTERING REGION 6 FROM REGION 1

IF (ETA1(J1, J2) < 0.5 * CH / 2) GO TO 100

THROUGH THE LEADING EDGE

CALL LEETA(ETA(2, J2), Y)
FETA1(J2) = Y
GO TO 110

THROUGH THE TRAILING EDGE
APPENDIX A

100 CALL TEETA(ETA(2,J2),Y)
ETA(1,J2) = CH
110 ETA(2,J2) = ETA2

SET ANGLE AND RETURN TO BLADE LOGIC

120 PHI = ACOS((J.9999999999*Y(1) - X(1))/SRCOST)
IF(Y(2).LT.X(2)) PHI = 2.*PI - PHI
GO TO 46

ENTERING REGION 6 FROM REGION 3. START FROM INTERSECTION WITH INNER RADIUS

130 CALL CALCY(RSQ,PHX)
Y(1) = YEND(1)
Y(2) = YEND(2)
ETA(1,J2) = CH/2. - Y(1)*CBETA + Y(2)*SBETA
IF(ETA(1,J2).LT.X) GO TO 90
ETA(2,J2) = -Y(1)*CBETA - Y(2)*SBETA
GO TO 120

ENTERING REGION 1 FROM REGION 6 THROUGH LEADING EDGE

140 IF(ETA(1,J2).GE.CH/2.) GO TO 150

CALL LEETA(ETA(2,J2),Y)
ETA(1,J2) = CH
GO TO 16C

THROUGH TAILING EDGE

150 CALL TEETA(ETA(2,J2),Y)
ETA(1,J2) = CH
15C ETA(2,J2) = ETA2
GO TO 50

LEAVING THE ROTOR SYSTEM FOR THIS ARC. GET INTERSECTION WITH OUTER RADIUS

170 CALL CALCY(RSQ,PHX)
Y(1) = YEND(1)
Y(2) = YEND(2)
GO TO 190

ENTERING REGION 3 FROM REGION 6. GET INTERSECTION WITH INNER RADIUS

180 CALL CALCY(RSQ,PHX)
Y(1) = YSTART(1)
Y(2) = YSTART(2)
190 ETA(1,J2) = CH/2. - Y(1)*SBETA + Y(2)*CBETA
IF(ETA(1,J2).LT.X) GO TO 146
ETA(2,J2) = -Y(1)*CBETA - Y(2)*SBETA
GO TO 5C
END
APPENDIX A

Function PFUNCN

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

Flow Chart of Function PFUNCN

```
Compute integrand

Return
```

Listing of Function PFUNCN

```fortran
function pfuncn(xeta, delf, deltalpha, y)
common eta2(8), x(3), yh(3), ystart(2), blad, delta(8), deltalpha, eta(2,8), alfa, beta, blades, bltn, cbeta,.
1 x(3), xh(3),
2 yend(2), yh(3), ystart(2),
3 vh(3)

dimension beta(5), coeff(5), eta2(2,8),
1 x(3), xh(3),
2 yend(2), yh(3), ystart(2),
3 vh(3)

calculate line integral

pfuncn = 0.

s = (((x(1) - y(1)) * cbeta + (y(2) - x(2)) * deltalpha) * delf
1 + (x(3) - y(3)) * deltalpha) / sqrt(s)
q = delf**2 + deltalpha**2 - s**2
if (q < 1.e-10) return
pfuncn = delf * omegea * xeta - yh(2) * cbeta + yh(1) * sbeta / sqrt(q)
return
end
```

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

ENTRY
LEETA

Set sign for leading edge

ENTRY
TEETA

Set sign for trailing edge

Compute intersection of sphere and blade edge

Return
APPENDIX A

Listing of Subroutine LEETA

**SUBROUTINE LEETA(XETA,Y)**

```fortran
SUBROUTINE LEETA IS CALLED BY SUBROUTINE CALCSP. THIS ROUTINE RETURNS THE COORDINATES (IN THE Y REFERENCE FRAME) OF THE INTERSECTION OF A FIXED SWEEPING ARC AND THE LEADING EDGE OF A BLADE. A SECOND ENTRY POINT, TEETA, DOES THE SAME FOR THE TRAILING EDGE. THE SECOND ETA COMPONENT IS ALSO RETURNED.

DIMENSION BETA(S), COEFFS(I5), ETA(2,8), X(3), XC(3), Y(3), YEND(2), YH(3), YSTART(2), VH(3)

COMMON ALFA, BETA, BLADES, BLNTN, CBETA, ETA, ETA2, ETA1, ETA2, ETA3, ETA4, ETA5, ETA6, ETA7, ETA8, ETA9, ETA10, ETA11, ETA12, ETA13, ETA14, ETA15, ETA16, ETA17, ETA18, ETA19, ETA20, ETA21, ETA22, ETA23, ETA24, ETA25, ETA26, ETA27, ETA28, ETA29, ETA30, ETA31, ETA32, ETA33, ETA34, ETA35, ETA36, ETA37, ETA38, ETA39, ETA40, ETA41

A = -0.5*CH
GO TO 1C
ENTRY TEETA
A = 0.5*CH
10 R = -A
XBAR1 = X(1) + A*S8ETA
XBAR2 = X(2) + D*CBETA
XBAR = X(1)*CBETA + X(2)*S8ETA
XTILDE = XBAR1*XBAR1 + XBAR2*XBAR2 - SRCOST*SRCOST
DISC1 = SQRT(YBAR*YBAR - XTILDE)
ETA2 = -XBAR + DISC1
IF(ABS(XETA - ETA2).GE.CH/FN) ETA2 = ETA2 - 2.*DISC1
Y(1) = B*S8ETA - ETA2*CRETA
Y(2) = A*CRETA - ETA2*S8ETA
RETURN
END
```

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

1. Set region indicator to zero
2. Point outside outer radius
   - Yes
   - Set region indicator to 3
   - Point inside hub radius
     - Yes
     - Return
     - No
8. Set region indicator to 6
9. Compute $\beta$ and $\eta_2$ for this blade
10. Increment blade counter
11. All blades processed
    - No
    - A
    - Yes
    - Return
12. Set region indicator to 0
13. Compute $\eta_1$ for this blade
14. Return
APPENDIX A

Listing of Subroutine TEST

**SUBROUTINE TEST (Y, RSQ, ROSQ, TIME)**

**C**

**SUBROUTINE TEST IS CALLED BY SUBROUTINE CALCSP, THIS**

**ROUTINE RETURNS THE REGION INDEX FOR A GIVEN TIME AND**

**A GIVEN PAIR OF COORDINATES IN THE Y REFERENCE FRAME.**

**THE REGION INDEX HAS FOUR POSSIBLE VALUES.**

**IREGION = 0, THE POINT LIES OUTSIDE THE CIRCLE**

**DEFINED BY THE BLADES OUTER RADIUS**

**IREGION = 1, THE POINT LIES BETWEEN THE CIRCLES**

**DEFINED BY THE BLADES OUTER AND**

**INNER RADII, BUT IS NOT ON A BLADE**

**IREGION = 2, THE POINT LIES INSIDE THE CIRCLE**

**DEFINED BY THE BLADES INNER RADIUS**

**IREGION = 3, THE POINT LIES ON A BLADE.**

**WHEN THE REGION INDEX IS 6, TEST RETURNS THE BLADE NUMBER,**

**J2, THE ANGLE MADE BY THAT BLADE, THE SIN AND COSINE OF**

**THIS ANGLE, SBETA AND CBETA, AND THE ETA COORDINATES OF**

**THE POINT.**

**DIMENSION BETA(8), COEFFS(5), ETA(2, 8),**

**1 X(3), X0(3), Y(3),**

**2 YEND(2), YH(3), YSTART(2),**

**3 VH(3),**

**COMMON ALFA, BETA, BLADES, BLNTH, CBETA,**

**1 CH, COEFFS, DELTAU, ETA, ETA2,**

**2 ETA1, F, FN, FNS, FETA1,**

**3 INF, J, J2, JFLG, JGLF, KFLG,**

**4 NLADES, OMEGAR, OTIME, PI, P,**

**5 R, RM, RMN, S, SBETA, SNOSPO,**

**6 SRCOST, SWPOST, SWPTAU, TAU, THKRAT,**

**7 TIMTHRU, VH, X, XO, YEND,**

**8 YH, YSTART, DELETA1, DELF,**

**RYSQ = Y(1)**2 + Y(2)**2

**IREGION = 0**

**IF(RY SQ.GT.RSQ) GO TO 3**

**IREGION = 16**

**WHEN THE REGION INDEX IS 6, TEST RETURNS THE BLADE NUMBER,**

**J2, THE ANGLE MADE BY THAT BLADE, THE SIN AND COSINE OF**

**THIS ANGLE, SBETA AND CBETA, AND THE ETA COORDINATES OF**

**THE POINT.**

**END**
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 $\Phi(\bar{X},t)$ 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

A

Read ROTOR data

Read ordered pairs

Apply trigonometric smoothing

Compute difference between raw and smooth data

Plot raw data

Plot smoothed data

B

PLOTPT

PLOTPT

CSDS
APPENDIX B

Plot raw data with smoothed data

Interpolate and differentiate smoothed data

Plot acoustic-pressure time history

A

Yes
More data

No
END
APPENDIX B

Listing of Program RTNPP

```plaintext
PROGRAM RTNPP(INPUT=10IB,OUTPUT=110B,TAPE5=INPUT,TAPE6=OUTPUT)
DIMENSION SCRIPTP(350),OTIME(350),F(500),T(500),DERI(500),
1 CER2(500),COEF(350),MK(350),DF(350)
DIMENSION XO(3),COEFFS(350),LOGICAL MORDATA
NAMELIST/ROTOR/XC,NBLADES,R,CH,OMEGA,RO,THKRAT,DELTEI,N,NS,KFLG,
1 ETAMAX,BLNTN,JFLG,COEFFS.TAUINT,PERDUM,OTAUM,SNDSPO,MORDATA
1 FORMAT(1HG,*,IERR = I5)
2 FORMAT(I5,3E25.15)
M = 0
INIT = C
30 CONTINUE
M = M + 1
30 CONTINUE
READ(5,2)
OTIME(M) = (OTIME(M) - OTIME(1)) / 499.
DO 40 IJ=1,500
T(IJ) = OTIME(IJ) + FLOAT(IJ-1)*DT
40 CONTINUE
S = FLOAT(M) / SORT(FLOAT(2*M))
PI = 3.141592654
NM1 = M - 1
DO 50 I=1,M
DERI(I) = 0.
WK(I) = 0.
DF(I) = SCRIPTP(I) - SCRIPTP(I-1) - FLOAT(I-1)*(SCRIPTP(M) - SCRIPTP(1))/S
1 (I))/FLOAT(NM1)
50 CONTINUE
WK(J) = 2.*WK(J)/FLOAT(NM1)
70 CONTINUE
DERI(I) = DERI(I) + WK(I)*DF(I)*SIN(FLOAT(J*(I-1))*PI/FLOAT(NM1))
80 CONTINUE
COEF(I,1) = DERI(I) + SCRIPTP(I) + FLOAT(I-1)*(SCRIPTP(M) - SCRIPTP(1))/S
1 - SCRIPTP(I))/FLOAT(NM1)
90 CONTINUE
COEF(I,1) = SCRIPTP(I)
COEF(M,1) = SCRIPTP(M)
DO 100 I=1,M
DF(I) = 1.5*ABS(COEF(I,1) - SCRIPTP(I))
100 CONTINUE
```

CALL CSOS(MNPTS,M,OTIME,SCRIPTP,DF,S,IM,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
  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
OTINE(M+1) = XMN  RTNPP 78
OTINE(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,IM,WK,1 IERR)  RTNPP 87
WRITE(6,1) IERR  RTNPP 88
DO 120 I=1,MM
  DER1(I) = 1000.*DER1(I)  RTNPP 90
120 CONTINUE  RTNPP 91
ML = 2  RTNPP 92
IF(MORMDATA) ML = 1  RTNPP 94
CALL PLOTPT(T,DER1,MM,ML,3)  RTNPP 95
M = 0  RTNPP 96
IF(MORMDATA) GO TO 13
STOP
END

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

No

First plot

Yes

Create plot vector file

PSEUDO

Plot number > 3

Yes

No

Determine MIN and MAX for this plot

Set size parameters

Adjust MIN and MAX

HELP

A
APPENDIX B

A

Branch to proper plotting call

Plot raw data

Plot smoothed data

Plot acoustic pressure

Plot raw data

Plot smoothed data

No Last plot

Yes

Terminate plot vector file

Return

INFOPLT

CALPLT
APPENDIX B

Listing of Subroutine PLOTPT

```
SUBROUTINE PLOTPT(X,Y,N,ML,J)
DIMENSION X(I),Y(I)
IF(ML.EQ.0) CALL PSEUDO
IF(J.GT.3) GO TO 15
XMIN = 1.E10
YMIN = 1.E10
XMAX =-1.E10
YMAX =-1.E10
DO 10 I=1,N
  IF(X(I).LT. XMIN) XMIN = X(I)
  IF(Y(I),LT.YMIN) YMIN = Y(I)
  IF(X(I).GT.XMAX) XMAX = X(I)
  IF(Y(I).GT.YMAX) YMAX = Y(I)
10 CONTINUE
GO TO 16
15 CONTINUE
XMIN = X(N*1)
XMAX = X(N+1)
YMIN = Y(N+1)
YMAX = Y(N+2)
16 CONTINUE
SH = 8.
SV = 5.
ISH = 8
ISV = 5
CALL HELP(XMIN,XMAX,ISH,XMIN,XMAX)
CALL HELP(YMIN,YMAX,ISV,YMIN,YMAX)
GO TO (2C,36,60,60,70),J
20 CALL INFOPLT(1,N,X,1,Y,1,XMIN,XMAX,YMIN,YMAX,1.,13,13)
1 13.13HBORDER TIME,16,16HSCRIPTP(1PAW DATA),0,SH,SV,1.,1.)
GO TO 50
30 CALL INFOPLT(1,N,X,1,Y,1,XMIN,XMAX,YMIN,YMAX,1.,13,13)
1 13.13HBORDER TIME,16,16HSCRIPTP(SPLINE),0,SH,SV,1.,1.)
GO TO 50
40 CALL INFOPLT(1,N,X,1,Y,1,XMIN,XMAX,YMIN,YMAX,1.,13,13)
1 28.28HBORDER TIME (MILLISECONDS),17,17HPRESSURE (NM**2),0.
2 SH,SV,1.,1.)
GO TO 50
50 CALL INFOPLT(0,N,X,1,Y,1,XMIN,XMAX,YMIN,YMAX,
1 13,13HBORDER TIME,7,7HSCRIPTP,0,SH,SV,1.,1.)
GO TO 50
70 CALL INFOPLT(1,N,X,1,Y,1,XMIN,XMAX,YMIN,YMAX,
1 13,13HBORDER TIME,7,7HSCRIPTP,0,SH,SV,1.,1.)
50 IF(ML.EQ.2) CALL CALPLT(0.,0..999)
RETURN
END
```

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

Shift MIN and MAX to common power of 10

Compute difference between MAX and MIN

Even multiple of plot size

Yes \[ \text{Return} \]

No

Select new MIN and MAX

Return

Listing of Subroutine HELP

```
SUBROUTINE HELP(X1,X2,n,XMIN,XMAX)
INTEGER 0
P1 = ALOG10 (ABS(X1))
P2 = ALOG10 (ABS(X2))
P = AMAX1(P1,P2)
IF(P.LT.0.) P = P - 1.
IP = P - 2.
X1P = X1/10.**IP
X2P = X2/10.**IP
IX1P = X1P - 1.
IX2P = X2P + 1.
IDIF = IX2P - IX1P
N = IOIF/(IP+1)
JDIF = N*IP
IF( (JDIF.EQ.JDIF) GO TO 10
N = N + 1
KDIF = N*IP
LOIF = (KDIF - IDIF)/2
IX1P = IX1P - LOIF
IX2P = IX1P + KDIF
10 CONTINUE
XMIN = FLOAT(IX1P)*10.**IP
XMAX = FLOAT(IX2P)*10.**IP
RETURN
END
```
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 \( \leq \) 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 \( (IX - (2*IX)**.5) \leq S \leq (IX + (2*IX)**.5) \) 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 \( X(I) \leq X1 \leq X(I + 1) \) is given by:
APPENDIX B

\[ F(X_1) = (COEF(I,4) \cdot H + COEF(I,3)) \cdot H + COEF(I,2)) \cdot H + COEF(I,1) \]

where \( H = X_1 - X(I) \)

WK A one-dimensional work area array dimensioned at least \((7 \cdot IX + 9)\) 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 \((X_i, F_i)\), \(i = 1, 2, \ldots, IX\) are given, where \(F_i\) is the functional value at point \(X_i\). A set of weights \(DF_1, DF_2, \ldots, DF_{IX}\) and a nonnegative smoothing parameter \(S\) are also given.

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

i) \( \sum_{i=1}^{IX} \left( \frac{G(X_i) - F_i}{DF_i} \right)^2 \leq S \)

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

Subroutine SPLDER

Language: FORTRAN

Purpose: To perform a cubic spline approximation, interpolation, and differentiation.

SPLDER computes \( F = f_1(X) \), \( DER1 = f'_1(X) \), and \( DER2 = f''_1(X) \) 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.
APPENDIX B

N  An input integer specifying the number of values in the independent variable array. \( N \geq MNPTS \).

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. \( M \geq MMAX \).

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.
APPENDIX B

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 $X_{IW} \leq X_0 \leq X_{IW+1}$. 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 $3(N_xN_CVS) + 8N$. 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 III 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 satisfy $X_1 \leq T_j \leq X_n$.

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.

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,NYMCM,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.
APPENDIX B

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.
APPENDIX B

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.
APPENDIX B

= 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.
APPENDIX B

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


   Vol. I.- General Information.
   Vol. II.- Subprogram Library.
   Vol. III.- Manufacturers Manuals.
   Vol. IV.- Special Capabilities.
### TABLE I.- ROTOR THICKNESS NOISE PROGRAM INPUT VARIABLES

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

*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.
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.
"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