N70-34200

NASA TECHNICAL MEMORANDUM

NASA TM X-2061

NASA TM X-2061

FORTRAN PROGRAM FOR CALCULATING AXIAL TURBOMACHINERY BLADE COORDINATES

by Theodore Katsanis

Lewis Research Center
Cleveland, Ohio 44135

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • AUGUST 1970

1. Report No. NASA TM X-2061	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle FORTRAN PROGRAM FOR CALCULATING AXIAL TURBOMACHINERY BLADE COORDINATES		5. Report Date August 1970
		6. Performing Organization Code
7. Author(s) Theodore Katsanis		8. Performing Organization Report No. E-5662
9. Performing Organization Name and Address Lewis Research Center National Aeronautics and Space Administration Cleveland, Ohio 44135		10. Work Unit No. 720-03
		11. Contract or Grant No.
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D. C. 20546		13. Type of Report and Period Covered Technical Memorandum
		14. Sponsoring Agency Code
15. Supplementary Notes		

16. Abstract

A FORTRAN IV computer program has been written to calculate blade coordinates with respect to the true blade chord. The required input is the axial blade chord, blade stagger, leading- and trailing-edge radii, angles of tangency on leading- and trailing-edge radii, and a few intermediate spline points. This input is identical to the geometrical input required for blade-to-blade aerodynamic analysis programs previously published by NASA (TN D-5427, TM X-1764, and TN D-5044).

17. Key Words (Suggested by Author(s)) Axial turbines; Blade coordinates	18. Distribution Statement Unclassified - unlimited

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	24	$3.00

*For sale by the Clearinghouse for Federal Scientific and Technical Information
Springfield, Virginia 22151

FORTRAN PROGRAM FOR CALCULATING AXIAL TURBOMACHINERY BLADE COORDINATES

by Theodore Katsanis

Lewis Research Center

SUMMARY

A FORTRAN IV computer program has been written to calculate blade coordinates with respect to the true blade chord. The required input is the axial blade chord, blade stagger, leading- and trailing-edge radii, angles of tangency on leading- and trailing-edge radii, and a few intermediate spline points. This input is identical to the geometrical input required for blade-to-blade aerodynamic analysis programs previously published by NASA (TN D-5427, TM X-1764, and TN D-5044).

INTRODUCTION

There are several NASA computer programs for calculating velocities on a blade-to-blade surface between blades (refs. 1 to 3). These programs are easy to use because the blades can be described very simply. The required geometrical input consists of the axial blade chord and stagger, leading- and trailing-edge radii, angles of tangency on the leading- and trailing-edge radii, and a few intermediate points which are fitted with a spline curve. This required geometrical input results in a precisely defined blade surface. After a satisfactory blade surface velocity distribution is obtained, it is often desired to calculate a large number of offset coordinates with respect to the true blade chord. The true blade chord is tangent to the lower surface of the blade. Since the blade shape is specified by mathematical equations, these coordinates may be calculated in a straightforward manner. However, this is a tedious and time consuming hand calculation. It is the purpose of the program TFORM to perform these calculations.

The FORTRAN IV program TFORM is presented herein with a complete description of the input required and the output obtained. The input and output for an example case are also given. The geometrical input is just a part of that required for the programs TSONIC, TURBLE, or TANDEM (refs. 1 to 3).

SYMBOLS

r	radius from axis of rotation
w	linear coordinate in tangential direction
w_b	w-coordinate of blade surface
w_0	w-coordinate of (x, y) origin
w_1	w-coordinate of $(x_i, 0)$
x	coordinate tangent to blade lower surface
x_i	x-coordinate at i^{th} increment from blade leading edge
y	coordinate normal to x-axis
y_i	y-coordinate at i^{th} increment from blade leading edge
$y_{l,i}$	y for lower blade surface
$y_{u,i}$	y for upper blade surface
z	axial distance from blade leading edge
z_b	z-coordinate of blade surface
z_0	z-coordinate of (x, y) origin
z_1	z-coordinate of $(x_i, 0)$
θ	angular coordinate about axis of rotation, radians
φ	blade angle from axial direction, deg

TRANSFORMATION PROCEDURE

The basic transformation consists of a rotation and translation. The input coordinates are given as (z, θ) coordinates where z is the axial direction and θ is the angular coordinate in radians about the axis of rotation. The linear coordinate in the θ-direction is equal to $r\theta = w$. The origin in the w-z plane is at the leading edge of the blade, as shown in figure 1. The entire curve for each surface is specified mathematically by the leading- and trailing-edge radii and by a spline curve in between. The output coordinates are given as (x, y) coordinates with the x-axis tangent to the blade lower surface and the y-axis tangent to the blade leading edge, as shown in figure 2.

The first step in the program is to determine the angle φ, the true chord, and (z_0, w_0), which are the (z,w) coordinates of the (x,y) origin. These constants specify the amount of translation and rotation and are calculated by equations (A1) to (A18) in the appendix.

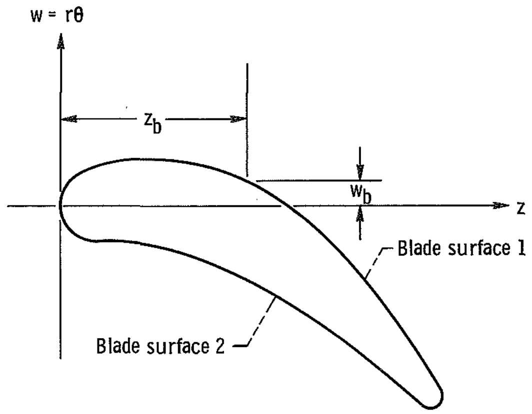

Figure 1. - Typical blade geometry.

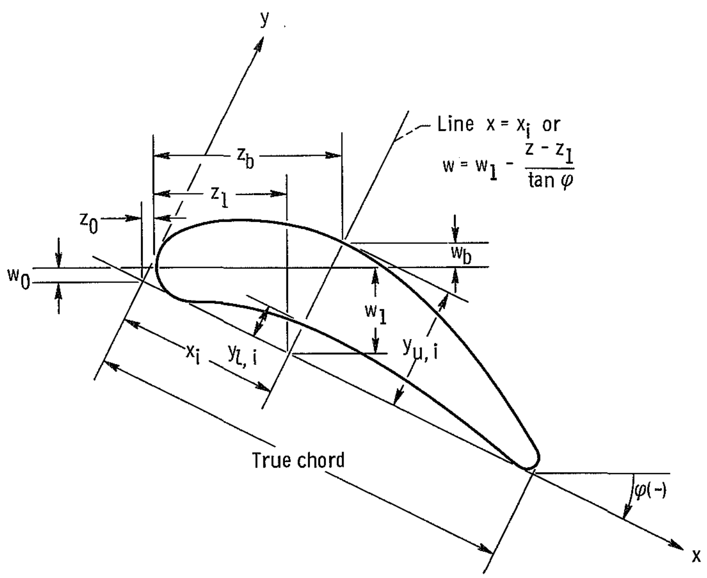

Figure 2. - Transformed coordinates and transformation constants.

The next step after the translation and rotation constants have been calculated is to calculate the y-coordinates for each blade surface corresponding to each increment in the x-direction. This is done by finding the intersection of the line $x = x_i$ and the curve $w = w(z)$ (see fig. 2). In w-z coordinates, the line $x = x_i$ is

$$w = w_1 - \frac{z - z_i}{\tan \varphi} \tag{1}$$

where

$$\left.\begin{aligned} z_1 &= z_0 + x_i \cos \varphi \\ w_1 &= w_0 + x_i \sin \varphi \end{aligned}\right\} \tag{2}$$

The blade surfaces are described mathematically by piecewise functions; that is, the leading- and trailing-edge segments are given by the equation of a circle and the rest of the blade by a spline curve which is a piecewise cubic polynomial (ref. 4). We can denote this by

$$w = w_b(z) \tag{3}$$

For any z then, w is determined as indicated by equation (3). Equations (1) and (3) can be solved simultaneously to determine (z_b, w_b) where the line intersects the blade. The numerical procedure for solving equations (1) and (3) is described in the appendix. Then y_i is calculated by

$$y_i = \sqrt{(z_1 - z_b)^2 + (w_1 - w_b)^2} \tag{4}$$

DESCRIPTION OF INPUT AND OUTPUT

The computer program requires as input a geometrical description in (z, θ) coordinates of the two blade surfaces, the radius r, a scale factor if desired, and the desired x-increment for the output coordinates. Output from the program includes x- and y-coordinates for the upper and lower surfaces (see fig. 2).

Input

Figure 3 shows the input variables as they are punched on the data cards. The first input card is for a title, which will serve for problem identification. The remaining cards are for input variables. All variables are real (decimal point must be punched) in a 10-column field. It should be noted that the input corresponds very closely to the blade geometry input for the NASA blade-to-blade analysis programs of references 1 to 3. Further explanation of the input variables is given in the Instructions for Preparing Input section.

1 10	11 20	21 30	31 40	41 50	51 60	61 70	71 80
TITLE							
CHORD	STGR	RMI	SCALE	DELX			
RI1	RO1	BETI1	BETO1	SPLNO1			
MSP1 ARRAY							
THSP1 ARRAY							
RI2	RO2	BETI2	BETO2	SPLNO2			
MSP2 ARRAY							
THSP2 ARRAY							

Figure 3. - Input form.

The input variables are as follows:

CHORD — Overall length of blade in the z-direction, see fig. 4

STGR — Angular θ-coordinate for center of trailing-edge circle of blade with respect to center of leading-edge circle, radians, see fig. 4

RMI — Radius of blade section from the axis of rotation (If RMI = 1, then all θ-coordinates are the actual linear dimension w.)

SCALE — Ratio of output dimensions to input dimensions (For example, if input is in feet and output is desired in inches, SCALE = 12 should be used.)

DELX — Spacing of output coordinates in the x-direction, see fig. 5 (DELX should be chosen to be at least CHORD*SCALE/100. DELX must be given in the output units; i.e., if input is in feet and output is in inches (SCALE = 12), then DELX is in inches.)

RI1, RI2 — Leading-edge radii of the two blade surfaces, see fig. 4

RO1, RO2 — Trailing-edge radii of the two blade surfaces, see fig. 4

BETI1, BETI2 — Angles (with respect to z-direction) at tangent points of leading-edge radii with the two blade surfaces, deg, see fig. 4 (These must be true angles in degrees.)

BETO1, BETO2 — Angles (with respect to z-direction) at tangent points of trailing-edge radii with the two blade surfaces

SPLNO1, SPLNO2 — Number of blade spline points given for each surface as input, maximum of 50 (These include the first and last points (dummies) that are tangent to the leading- and trailing-edge radii (fig. 4).)

MSP1, MSP2 — Arrays of z-coordinates of spline points on the two blade surfaces, measured from blade leading edges, see fig. 4 (The first and last points in each of these arrays must be left blank, since these values are calculated by the program. If the last point is on a new card, a blank card must be used.)

THSP1, THSP2 — Arrays of θ-coordinates of spline points corresponding to MSP1 and MSP2, radians, see fig. 4 (Blanks must be used in positions corresponding to those in MSP1 and MSP2.)

Instructions for Preparing Input

Units of measurement. - Two units are used: one for linear measurements and one for angles. Any unit may be used for linear measurement. If a different unit is desired for output, this may be accomplished by the use of a scale factor in SCALE. If SCALE = 1, the output units are the same as the input units. The angular measurement θ must be given in radians. However, if RMI = 1 is specified, the θ-coordinate can be given as a true linear measurement.

Blade geometry. - The upper and lower surfaces of the blade are each defined by specifying three things: leading- and trailing-edge radii, angles at which these radii are tangent to the blade surfaces, and z- and θ-coordinates of several points along each surface. These angles and coordinates are used to define a cubic spline curve fit (ref. 4) to the surface. The standard sign convention is used for angles, as indicated in figure 4. The blade must be oriented with a concave lower surface.

A cubic spline curve is a piecewise cubic polynomial which expresses mathematically the shape taken by an idealized spline passing through the given points. Reference 4 describes a method for determining the equation of the spline curve. When this method is used, only a few points are required to specify most blade shapes accurately, usually no more than five or six, in addition to the two end points. As a guide, enough points should be specified so that a physical spline passing through these points would accurately follow the blade shape. This means that the spline points should be closer where there is large curvature and farther apart where there is small curvature.

The coordinates for either surface of the blade are given with respect to the leading edge, with the leading edge of the blade being defined as the furthest point upstream.

Format for input data. - All input variables are real numbers (punch decimal point) in a 10-column field.

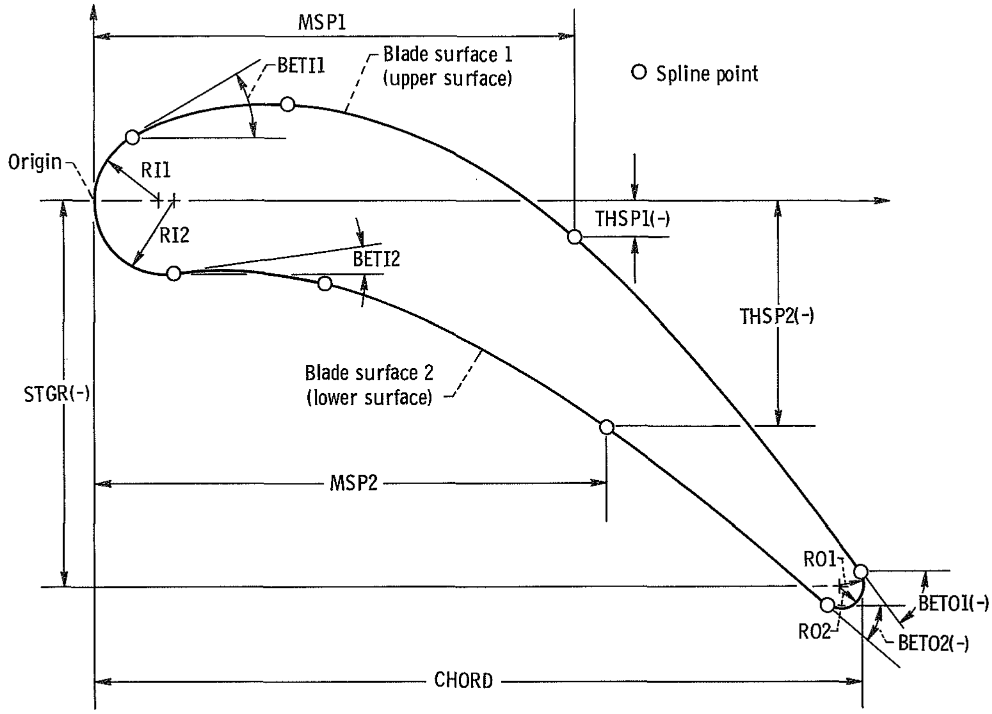

Figure 4. - Geometric input variables. Angles BETI1, BETI2, BETO1, and BETO2 must be given as true angle in degrees, not angle as measured in z-θ plane.

TABLE I. - INPUT FOR SAMPLE PROBLEM

2ND ROTOR HUB SECTION

CHORD	STGR	RMI	SCALE	DELX
0.7966700E-01	-0.3615000E-01	0.2593300	12.000000	0.5000000E-01

BLADE SURFACE 1 -- UPPER SURFACE

RI1	RO1	BETI1	BETO1	SPLNO1
0.3125000E-02	0.8330000E-03	41.000000	-46.300000	3.0000000

MSP1 ARRAY

-0	0.5883000E-01	-0

THSP1 ARRAY

-0	0.3454000E-01	-0

BLADE SURFACE 2 -- LOWER SURFACE

RI2	RO2	BETI2	BETO2	SPLNO2
0.3125000E-02	0.8330000E-03	29.000000	-35.500000	3.0000000

MSP2 ARRAY

-0	0.5883000E-01	-0

THSP2 ARRAY

-0	0.4820000E-02	-0

TABLE II. - OUTPUT FOR SAMPLE PROBLEM

BLADE DATA AT INPUT SPLINE POINTS

BLADE SURFACE 1

Z	THETA	DERIVATIVE	2ND DERIV.
0.10748E-02	0.90945E-02	3.35205	-101.424
0.58830E-01	0.34540E-01	-2.45348	-99.6146
0.79436E-01	-0.33931E-01	-4.03517	-53.9014

BLADE SURFACE 2

Z	THETA	DERIVATIVE	2ND DERIV.
0.46400E-02	-0.10539E-01	2.13747	-66.5432
0.58830E-01	0.48200E-02	-1.62164	-72.1948
0.78350E-01	-0.38765E-01	-2.75052	-43.4682

NO. OF POINTS = 21 PHI = -5.3371 DEGREES

X	Y LOWER	Y UPPER
0	0.37500E-01	0.37500E-01
0.50000E-01	0.21445E-02	0.10269
0.10000	0.33325E-01	0.14606
0.15000	0.61872E-01	0.18273
0.20000	0.86266E-01	0.21294
0.25000	0.10657	0.23691
0.30000	0.12285	0.25485
0.35000	0.13516	0.26696
0.40000	0.14356	0.27343
0.45000	0.14812	0.27443
0.50000	0.14888	0.27014
0.55000	0.14591	0.26071
0.60000	0.13926	0.24630
0.65000	0.12899	0.22705
0.70000	0.11513	0.20310
0.75000	0.97808E-01	0.17468
0.80000	0.77298E-01	0.14224
0.85000	0.53936E-01	0.10626
0.90000	0.28038E-01	0.67177E-01
0.95000	0.32655E-03	0.25395E-01
0.96253	0.99960E-02	0.99960E-02

Output

Sample output is given in table II for the example blade given in table I. The first output gives additional computed blade data at the input spline points. This includes the z- and θ-coordinates at the points where the spline curves are tangent to the leading- and trailing-edge radii. Also, the first and second derivatives are given at each spline point. Of particular interest are the second derivatives. Any error in blade geometry input will

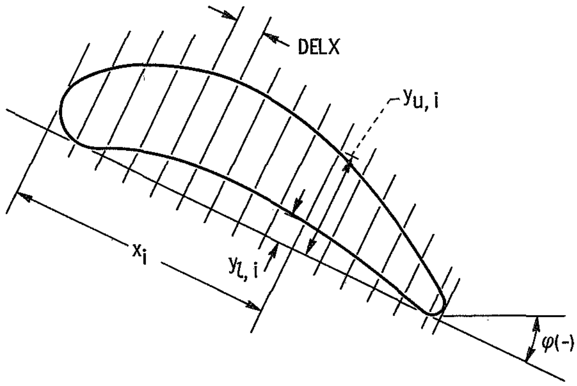

Figure 5. - Output coordinates.

usually result in wild values for some of these second derivatives.

The next output gives the transformed blade coordinates. The first line of output gives the number of x-coordinates and the orientation angle φ, as shown in figure 5. This is followed by a tabulation of the x- and y-coordinates for the upper and lower surfaces (see fig. 5).

Error Conditions

The error message is given first for each error condition:

(1) BETI2 MUST BE GREATER THAN PHI AND BETO2 MUST BE LESS THAN PHI TO HAVE X AXIS TANGENT TO LOWER BLADE SURFACE

It is assumed in the program that the x-axis is tangent to the leading- and trailing-edge radii. If either BETI2 is less than φ or BETO2 is greater than φ this tangent line will not actually be tangent to the lower blade surface, and part of the lower surface will be below the x-axis. Normal calculations will still be made, but there will be negative values for Y LOWER.

(2) PART OF BLADE HAS NEGATIVE X VALUES

This message is printed if part of the blade would extend to the left of y-axis. This can happen if BETI1 is greater than $\varphi + 90^{\circ}$ or if BETI2 is less than $\varphi - 90^{\circ}$. No further calculations are made and the program will proceed to the next case.

(3) LOWER BLADE SURFACE IS NOT ENTIRELY CONCAVE

This message is printed if some part of the blade lies below the x-axis. Normal calculations will still be made, including negative values for Y UPPER or Y LOWER.

(4) Z COORDINATE IS NOT WITHIN BLADE

This message is printed by subroutine BLCD if the z-coordinate given this subroutine as input is not within the bounds of blade surface. The value of z and the blade surface number are also printed when this happens. This message should only occur if there is an error in the input data.

(5) ROOT HAS FAILED TO OBTAIN A VALID ROOT

This message is printed by subroutine ROOT if a root cannot be located, or if the accuracy of the root is not satisfactory. The user should thoroughly check the input data.

PROGRAM PROCEDURE

The main program is TFORM. There are 4 subroutines: FUNCT, ROOT, BLCD, and SPLN22. The calling relation of all the subroutines is shown in figure 6.

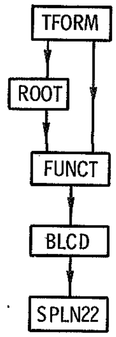

Figure 6. - Calling relation of subroutines.

TFORM reads and prints out all the input data. Then the transformation constants φ, z_0, and w_0 are calculated as described in the appendix. Next the x and y arrays are calculated. The method for calculating y for a given x value is described in the appendix. The root finding procedure required by this method is accomplished by subroutine ROOT.

Subroutine FUNCT calculates f(z) in equation (A19) for either the upper or lower surface.

Subroutines ROOT, BLCD, and SPLN22 are the same as described in references 1 to 3. Subroutine ROOT was changed in reference 1 from the coding used in references 2 and 3. This was to adopt the more foolproof method of locating roots by the bisection method. Subroutine BLCD calculates the θ blade coordinates when given a z-coordinate. Subroutine SPLN22 calculates the spline curve for the blade surfaces.

FORTRAN Variables in TFORM and FUNCT

A	a, fig. 7
B	b, fig. 7
BETI	array, BETI1 or BETI2, see input
BETO	array, BETO1 or BETO2, see input
C	c, fig. 7
CHORD	see input
CNVX	if CNVX ≥ 1, either the lower or the upper surface has negative y-coordinates
CPHI	cos φ
D	d, fig. 7
DELX	see input
E	e, fig. 8
F	f, figs. 8 and 9
FZ	f(z), eq. (A19)
G	g, figs. 8 and 9
H	h, figs. 8 and 9
I	temporary index
INDEX	used as both a switch and subscript in calculating y blade coordinates
ISURF	index indicating blade surface number
J	index for DO loop
MSP	input arrays MSP1 or MSP2
NOPT	number of points in x and y output arrays
NSP	number of spline points
NSPI	array of number of spline points
P	p, figs. 8 and 9
PHI	φ, fig. 7
PHICC	φ_{c-c}, fig. 7
PHICOR	φ_{corr}, fig. 7
PHIDEG	φ, deg

PI π

RI array, RI1 or RI2, see input

RMI see input

RO array, RO1 or RO2, see input

SCALE see input

SPHI $\sin \varphi$

SPLNO either SPLNO1 or SPLNO2, see input

SRW integer code variable that causes either ROOT (if SRW = 21) or SPLN22 (if SRW = 18) to write out data useful for debugging

STGR see input

TCHORD true chord, fig. 2

THETA θ

THSP input arrays THSP1 or THSP2

TOLERW permissible tolerance in value of w for a given value of x, TOLERW $= \text{CHORD} \times 10^{-4}$

TPHI $\tan \varphi$

W0 w_0, fig. 2

WB w_b, fig. 2

W1 w_1, fig. 2

X array of output values of x

Y array of output values of y_l and y_u

Z0 z_0, fig. 2

Z1 z_1, fig. 2

ZB z_b, fig. 2

ZERO zero value variable

ZL, ZT if Z1 is less than ZL or greater than ZT, the blade surface in the x, y-coordinates is the opposite of the one in the w, z-coordinates

PROGRAM LISTING

```
      COMMON SRW,INIT(2),TPHI,WB,Z1,W1,CHORD,STGR,RMI,RI(2),RO(2),
     1   BETI(2),BETO(2),NSPI(2),MSP(50,2),THSP(50,2)
      DIMENSION X(101),Y(101,2)
      REAL MSP
      EXTERNAL FUNCT1,FUNCT2
    1 CONTINUE
      INIT(1) = 0
      INIT(2) = 0
C
C  READ AND PRINT ALL INPUT DATA
C
      WRITE(6,1000)
      READ(5,1100)
      WRITE(6,1100)
      WRITE(6,1110)
      READ (5,1030) CHORD,STGR,RMI,SCALE,DELX
      WRITE(6,1040) CHORD,STGR,RMI,SCALE,DELX
      DO 10 J=1,2
      IF (J.EQ.1) WRITE(6,1120)
      IF (J.EQ.2) WRITE(6,1130)
      WRITE(6,1140) J,J,J,J,J
      READ (5,1030) RI(J),RO(J),BETI(J),BETO(J),SPLNO
      WRITE(6,1040) RI(J),RO(J),BETI(J),BETO(J),SPLNO
      NSPI(J)= SPLNO
      NSP = NSPI(J)
      WRITE(6,1150) J
      READ (5,1030) (MSP(I,J),I=1,NSP)
      WRITE(6,1040) (MSP(I,J),I=1,NSP)
      WRITE(6,1160) J
      READ (5,1030) (THSP(I,J),I=1,NSP)
   10 WRITE(6,1040) (THSP(I,J),I=1,NSP)
C
C   CALCULATE TRANSFORMATION CONSTANTS
C
      PI = 3.1415927
      CNVX = 0.
      TOLERW = CHORD/1.E4
      A = CHORD-RI(2)-RO(2)
      B = STGR*RMI
      C = SQRT(A*A+B*B)
      PHICC = ATAN(B/A)
      PHICOR = ARSIN((RI(2)-RO(2))/C)
      PHI = PHICC+PHICOR
      PHIDEG = PHI/PI*180.
      IF(BETI(2).LT.PHIDEG.OR.BETO(2).GT.PHIDEG) WRITE (6,1165)
      IF(BETI(1)-90..LE.PHIDEG.AND.90.+BETI(2).GE.PHIDEG) GO TO 15
      WRITE(6,1167)
      GO TO 1
   15 CONTINUE
      SPHI = SIN(PHI)
      CPHI = COS(PHI)
      TPHI = TAN(PHI)
      D = C*COS(PHICOR)
      E = (RI(1)-RI(2))*CPHI
      F = RI(2)*SPHI
```

```
      G = (RI(1)-E)*CPHI
      H = RI(2)*CPHI
      P = (RI(1)-E)*SPHI
      TCHORD = D-E+RI(1)+RO(2)
      IF(PHI.LE.0.) GO TO 20
      E = (RO(1)-RO(2))*CPHI
      G = RI(2)*CPHI
      P = F
      TCHORD = D-E+RI(2)+RO(1)
   20 Z0 = RI(2)+F-G
      W0 = -H-P
C
C    CALCULATE X AND Y ARRAYS
C
      X(1) = 0.
      Y(1,1) = RI(2)*SCALE
      IF(PHI.LT.0.) Y(1,1) = Y(1,1)+SPHI*(RI(2)-RI(1))*SCALE
      Y(1,2) = Y(1,1)
      I = 1
   35 I = I+1
      X(I) = X(I-1)+DELX
      Z1 = Z0+X(I)/SCALE*CPHI
      W1 = W0+X(I)/SCALE*SPHI
      ZERO = 0.
      ZL = -W1*TPHI
      ZT = CHORD-(W1-RMI*STGR)*TPHI
      DO 100 ISURF=1,2
      INDEX = ISURF
      A = 0.
      B = CHORD
      ZB = Z1
      IF(PHI.EQ.0.) GO TO (60,80),INDEX
      IF(PHI*(FLOAT(ISURF)-1.5).LE.0.) GO TO 40
C    PHI NEGATIVE AND UPPER SURFACE   OR
C    PHI POSITIVE AND LOWER SURFACE
      A = RI(INDEX)*(1.-CPHI)
      IF(Z1.LE.ZT) GO TO (50,70),INDEX
      INDEX = 3-INDEX
      A = CHORD-RO(INDEX)*(1.-CPHI)
      GO TO (50,70),INDEX
C    PHI NEGATIVE AND LOWER SURFACE   OR
C    PHI POSITIVE AND UPPER SURFACE
   40 B = CHORD-RO(INDEX)*(1.-CPHI)
      IF(Z1.GE.ZL) GO TO (50,70),INDEX
      INDEX = 3-INDEX
      B = RI(INDEX)*(1.-CPHI)
      GO TO (50,70),INDEX
   50 CALL ROOT(A,B,ZERO,FUNCT1,TOLERW,ZB)
   60 CALL FUNCT1(ZB,FZ)
      GO TO 90
   70 CALL ROOT(A,B,ZERO,FUNCT2,TOLERW,ZB)
   80 CALL FUNCT2(ZB,FZ)
   90 Y(I,ISURF) = SQRT((ZB-Z1)**2+(WB-W1)**2)*SCALE
      IF(WB.GE.W1) GO TO 100
      CNVX = CNVX+1.
      Y(I,ISURF) = -Y(I,ISURF)
  100 CONTINUE
      IF(X(I)+DELX.LT.TCHORD*SCALE.AND.I.LT.100) GO TO 35
      IF(CNVX.GT.0.) WRITE(6,1190)
```

```
      NOPT = I+1
      X(NOPT) = TCHORD*SCALE
      Y(NOPT,1) = RO(2)*SCALE
      IF(PHI.GT.0.) Y(NOPT,1) = Y(NOPT,1)+SPHI*(RO(1)-RO(2))*SCALE
      Y(NOPT,2) = Y(NOPT,1)
      PHI = PHI/PI*180.
      WRITE(6,1170) NOPT,PHI
      WRITE(6,1180) (X(I),Y(I,2),Y(I,1),I=1,NOPT)
      GO TO 1
 1000 FORMAT (1H1)
 1030 FORMAT (8F10.5)
 1040 FORMAT (1X,8G16.7)
 1100 FORMAT (80H
     1                          )
 1110 FORMAT (5X,5HCHORD,12X,4HSTGR,13X,3HRMI,12X,5HSCALE,12X,4HDELX)
 1120 FORMAT (39HL     BLADE SURFACE 1  --  UPPER SURFACE)
 1130 FORMAT (39HL     BLADE SURFACE 2  --  LOWER SURFACE)
 1140 FORMAT (7X,2HRI,I1,12X,2HRO,I1,12X,4HBETI,I1,11X,4HBETO,I1,11X,5HS
     1PLNO,I1)
 1150 FORMAT (7X,3HMSP,I1,2X,5HARRAY)
 1160 FORMAT (7X,4HTHSP,I1,2X,5HARRAY)
 1165 FORMAT (111HL BETI2 MUST BE GREATER THAN PHI, AND BETO2 MUST BE LE
     1SS THAN PHI TO HAVE X AXIS TANGENT TO LOWER BLADE SURFACE/1HL)
 1167 FORMAT (37HL PART OF BLADE HAS NEGATIVE X VALUES)
 1170 FORMAT (18H1  NO. OF POINTS =,I4,10X,5HPHI =,G12.4,8H DEGREES)
 1180 FORMAT (46HL     X                  Y LOWER             Y UPPER/
     1   (2X,3(G13.5,5X)))
 1190 FORMAT (45HL LOWER BLADE SURFACE IS NOT ENTIRELY CONCAVE)
      END
```

```
      SUBROUTINE FUNCT
      COMMON SRW,INIT(2),TPHI,WB,Z1,W1,CHORD,STGR,RMI,RI(2),RO(2),
     1   BETI(2),BETO(2),NSPI(2),MSP(50,2),THSP(50,2)
      ENTRY FUNCT1(Z,FZ)
      CALL BL1(Z,THETA)
      GO TO 10
      ENTRY FUNCT2(Z,FZ)
      CALL BL2(Z,THETA)
   10 WB = THETA*RMI
      IF(TPHI.NE.0.) FZ = WB-W1+(Z-Z1)/TPHI
      RETURN
      END
```

```
      SUBROUTINE BLCD
C
C  BLCD CALCULATES BLADE THETA COORDINATE AS A FUNCTION OF M (=Z FOR AXIAL)
C
      COMMON SRW,INIT(2),TPHI,WB,Z1,W1,CHORD,STGR,RMI,RI(2),RO(2),
     1   BETI(2),BETO(2),NSPI(2),MSP(50,2),THSP(50,2)
      DIMENSION EM(50,2),AAA(50)
```

```
      INTEGER SRW,SURF
      ENTRY BL1(M,THETA)
      REAL M,MSP,MSPMM,MMMSP
      SURF= 1
      SIGN= 1.
      GO TO 10
      ENTRY BL2(M,THETA)
      SURF= 2
      SIGN=-1.
   10 CONTINUE
      NSP= NSPI(SURF)
      IF (INIT(SURF).EQ.13) GO TO 30
      INIT(SURF)= 13
C
C   INITIAL CALCULATION OF FIRST AND LAST SPLINE POINTS ON BLADE
C
      AA = BETI(SURF)/57.295779
      AA = SIN(AA)
      MSP(1,SURF) =  RI(SURF)*(1.-SIGN*AA)
      BB = SQRT(1.-AA**2)
      THSP(1,SURF) = SIGN*BB*RI(SURF)/RMI
      BETI(SURF) = AA/BB/RMI
      AA = BETO(SURF)/57.295779
      AA = SIN(AA)
      MSP(NSP,SURF) = CHORD-RO(SURF)*(1.+SIGN*AA)
      BB = SQRT(1.-AA**2)
      THSP(NSP,SURF) = STGR+SIGN*BB*RO(SURF)/RMI
      BETO(SURF) = AA/BB/RMI
      CALL SPLN22(MSP(1,SURF),THSP(1,SURF),BETI(SURF),BETO(SURF),NSP,
     1 AAA,EM(1,SURF))
      IF (SURF.EQ.1) WRITE(6,1000)
      WRITE(6,1010) SURF
      WRITE (6,1020) (MSP(IA,SURF),THSP(IA,SURF),AAA(IA),EM(IA,SURF),
     1    IA=1,NSP)
C
C   BLADE COORDINATE CALCULATION
C
   30 KK = 2
      IF (M.GT.MSP(1,SURF)) GO TO 50
C
C   AT LEADING EDGE RADIUS
C
      IF(M.LT.0.) GO TO 90
      THETA = SQRT(M*(2.*RI(SURF)-M))*SIGN
      IF (THETA.EQ.0.) GO TO 40
      RMM = RI(SURF)-M
      THETA = THETA/RMI
      RETURN
   40 THETA = 0.
      RETURN
C
C   ALONG SPLINE CURVE
C
   50 IF (M.LE.MSP(KK,SURF)) GO TO 60
      IF (KK.GE.NSP) GO TO 70
      KK = KK+1
      GO TO 50
   60 S= MSP(KK,SURF)-MSP(KK-1,SURF)
      EMKM1= EM(KK-1,SURF)
```

```
      EMK= EM(KK,SURF)
      MSPMM= MSP(KK,SURF)-M
      MMMSP= M-MSP(KK-1,SURF)
      THK= THSP(KK,SURF)/S
      THKM1= THSP(KK-1,SURF)/S
      THETA= EMKM1*MSPMM**3/6./S + EMK*MMMSP**3/6./S + (THK-EMK*S/6.)*
     1 MMMSP + (THKM1-EMKM1*S/6.)*MSPMM
      RETURN
C
C  AT TRAILING EDGE RADIUS
C
   70 CMM = CHORD-M
      IF(CMM.LT.-CHORD/1.E5) GO TO 90
      CMM= AMAX1(0.,CMM)
      THETA= SQRT(CMM*(2.*RO(SURF)-CMM))*SIGN
      IF (THETA.EQ.0.) GO TO 80
      RMM= RO(SURF)-CMM
      THETA = STGR+THETA/RMI
      RETURN
   80 THETA = STGR
      RETURN
C
C  ERROR RETURN
C
   90 WRITE(6,1030) M,SURF
      THETA = 0.
      RETURN
 1000 FORMAT (1H1,13X,33HBLADE DATA AT INPUT SPLINE POINTS)
 1010 FORMAT(1HL,17X,16HBLADE     SURFACE,I4)
 1020 FORMAT (7X ,1HZ,10X,5HTHETA,10X,10HDERIVATIVE,5X,10H2ND DERIV. /
     1  (4G15.5) )
 1030 FORMAT (33H Z COORDINATE IS NOT WITHIN BLADE/4H Z =,G14.6,10X,
     1   6HSURF =,G14.6)
      END
```

```
      SUBROUTINE ROOT(A,B,Y,FUNCT,TOLERY,X)
C
C  ROOT FINDS A ROOT FOR (FUNCT MINUS Y) IN THE INTERVAL (A,B)
C
      COMMON SRW
      INTEGER SRW
      IF (SRW.EQ.21) WRITE(6,1000) A,B,Y,TOLERY
      X1 = A
      CALL FUNCT(X1,FX1)
      IF(SRW.EQ.21) WRITE(6,1010) X1,FX1
      X2 = B
   10 DO 30 I=1,20
      X = (X1+X2)/2.
      CALL FUNCT(X,FX)
      IF(SRW.EQ.21) WRITE(6,1010) X,FX
      IF((FX1-Y)*(FX-Y).GT.0.) GO TO 20
      X2 = X
      GO TO 30
   20 X1 = X
      FX1 = FX
```

```
   30 CONTINUE
      IF(ABS(Y-FX).LT.TOLERY) RETURN
      WRITE(6,1020) A,B,Y,X,FX
      RETURN
 1000 FORMAT (32H1INPUT ARGUMENTS FOR ROOT -- A =G13.5,3X,3HB =,G13.5,
     1   3X,3HY =,G13.5,3X,8HTOLERY =,G13.5/16X,1HX,17X,2HFX)
 1010 FORMAT(8X,G16.5,G18.5)
 1020 FORMAT(37HLROOT HAS FAILED TO OBTAIN VALID ROOT/4H A =,G14.6,
     1   10X,3HB =,G14.6,10X,3HY =,G14.6,3HX =,G14.6,4HFX =,G14.6)
      END
```

```
      SUBROUTINE SPLN22 (X,Y,Y1P,YNP,N,SLOPE,EM)
C
C  SPLN22 CALCULATES FIRST AND SECOND DERIVATIVES AT SPLINE POINTS
C  END CONDITION - DERIVATIVES SPECIFIED AT END POINTS
C
      COMMON SRW
      DIMENSION X(N),Y(N),EM(N),SLOPE(N)
      DIMENSION SB(100),G(100)
      INTEGER SRW
      SB(1) = .5
      F = (Y(2)-Y(1))/(X(2)-X(1))-Y1P
      G(1) = F*3./(X(2)-X(1))
      NO=N-1
      IF(NO.LT.2) GO TO 20
      DO 10 I=2,NO
      A = (X(I)-X(I-1))/6.
      C = (X(I+1)-X(I))/6.
      W = 2.*(A+C)-A*SB(I-1)
      SB(I) = C/W
      F = (Y(I+1)-Y(I))/(X(I+1)-X(I))-(Y(I)-Y(I-1))/(X(I)-X(I-1))
   10 G(I) = (F-A*G(I-1))/W
   20 F = YNP-(Y(N)-Y(N-1))/(X(N)-X(N-1))
      W = (X(N)-X(N-1))/6.*(2.-SB(N-1))
      EM(N) = (F-(X(N)-X(N-1))*G(N-1)/6.)/W
      DO 30 I=2,N
      K = N+1-I
   30 EM(K) = G(K)-SB(K)*EM(K+1)
      SLOPE(1) = (X(1)-X(2))/6.*(2.*EM(1)+EM(2))+(Y(2)-Y(1))/(X(2)-X(1))
      DO 40 I=2,N
   40 SLOPE(I) = (X(I)-X(I-1))/6.*(2.*EM(I)+EM(I-1))+(Y(I)-Y(I-1))/
     1   (X(I)-X(I-1))
      IF(SRW.EQ.18) WRITE (6,1000) N,(X(I),Y(I),SLOPE(I),EM(I),I=1,N)
      RETURN
 1000 FORMAT (2X,15HNO. OF POINTS =,I3/10X,1HX,19X,1HY,19X,5HSLOPE,15X,
     12HEM/(4G20.8))
      END
```

Lewis Research Center,

National Aeronautics and Space Administration,

Cleveland, Ohio, May 20, 1970,

720-03.

APPENDIX - EQUATIONS FOR CALCULATING TRANSLATION AND ROTATION CONSTANTS

The following equations can be obtained by referring to figure 7:

$$a = \text{CHORD} - r_i(2) - r_0(2) \tag{A1}$$

$$c = \sqrt{a^2 + b^2} \tag{A2}$$

$$\tan \varphi_{c-c} = \frac{b}{a} \tag{A3}$$

$$\sin \varphi_{corr} = \frac{r_i(2) - r_0(2)}{c} \tag{A4}$$

$$\varphi = \varphi_{c-c} + \varphi_{corr} \tag{A5}$$

The angle φ is the desired angle between the x-axis and the z-axis (fig. 2). The amount of translation (z_0 and w_0) is obtained next. There are two sets of equations, depending on whether φ is positive or negative.

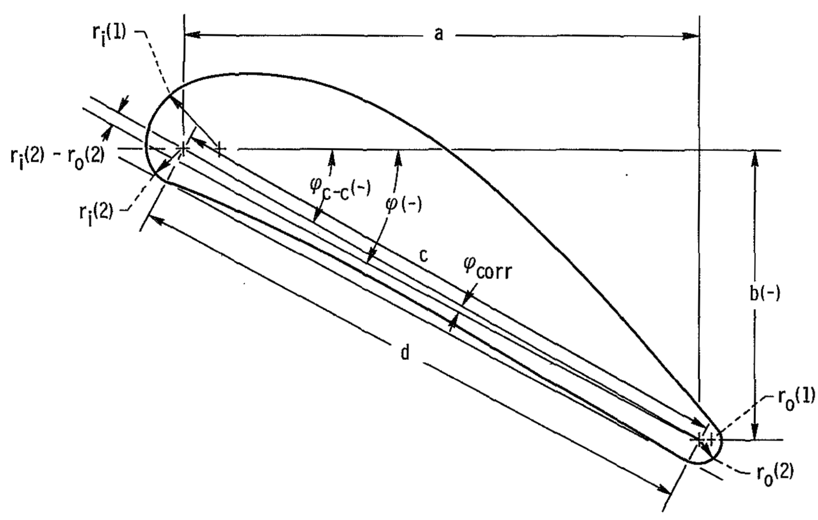

Figure 7. - Quantities required to compute φ.

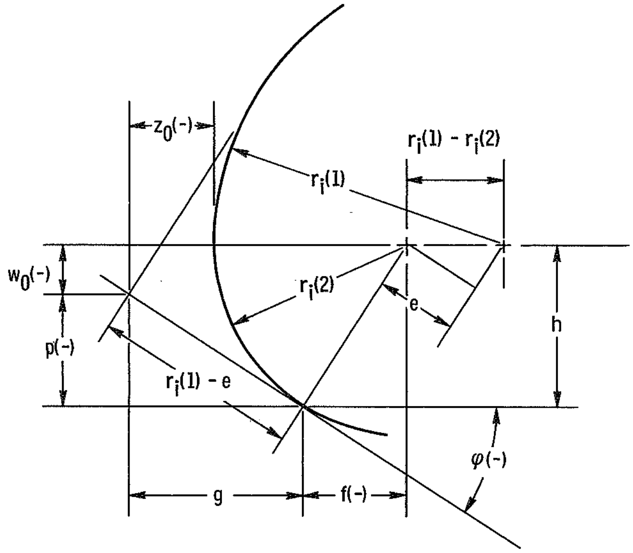

Figure 8. - Quantities required to compute true chord for negative φ.

When φ is negative or zero, the following equations hold (see figs. 7 and 8):

$$d = c \cos \varphi_{corr} \tag{A6}$$

$$e = \left[r_i(1) - r_i(2)\right] \cos \varphi \tag{A7}$$

$$f = r_i(2) \sin \varphi \tag{A8}$$

$$g = \left[r_i(1) - e\right] \cos \varphi \tag{A9}$$

$$h = r_i(2) \cos \varphi \tag{A10}$$

$$p = \left[r_i(1) - e\right] \sin \varphi \tag{A11}$$

$$z_0 = r_i(2) + f - g \tag{A12}$$

$$w_0 = -h - p \tag{A13}$$

$$\text{True chord} = d - e + r_i(1) + r_0(2) \tag{A14}$$

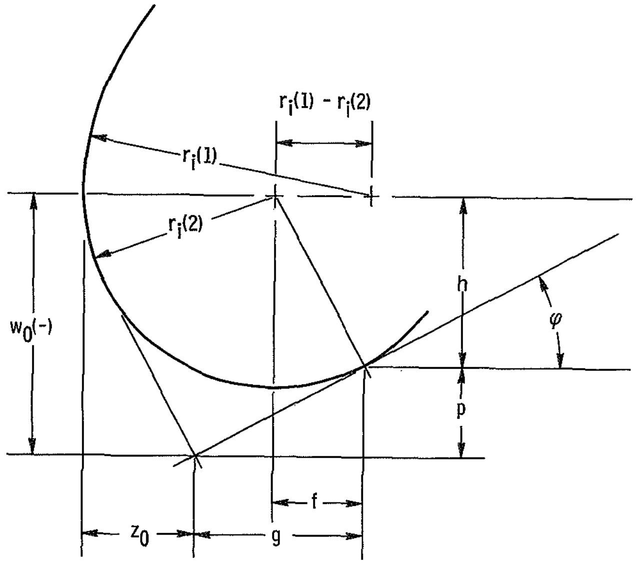

Figure 9. - Quantities required to compute true chord for positive φ (e is at trailing edge).

On the other hand, when φ is positive, the equations for e, g, p, and the true chord change as follows (see fig. 9):

$$e = \left[r_0(1) - r_0(2)\right] \cos \varphi \tag{A15}$$

$$g = r_i(2) \cos \varphi \tag{A16}$$

$$p = r_i(2) \sin \varphi \tag{A17}$$

$$\text{True chord} = d - e + r_i(2) + r_0(1) \tag{A18}$$

This completes the determination of the transformation constants.

CALCULATION OF y-COORDINATE OF BLADE SURFACE

The problem here is to find the y-coordinate for a given blade surface corresponding to a given x-coordinate. This can be done by the simultaneous solution of equations (1) and (3). If w is eliminated, z can be obtained by finding a root for the function

$$f(z) = w_b(z) - w_1 + \frac{z - z_1}{\tan \varphi} \tag{A19}$$

There are two problems that arise. First, the function $w_b(z)$ must be the correct blade surface. This is not straightforward, since y_u could be on the lower surface in the w-z plane near the leading edge. The second problem is that for certain values of z and w near the leading or trailing edge there may be two solutions to equation (A19). Both of these problems are overcome by restricting the interval for z for which the solution is found. After the proper interval for z has been determined, the proper surface can be ascertained so that equation (A19) must have a unique root.

With the proper interval for z and the correct blade surface the unique root for equation (A19) is found by the bisection method. That is, the interval is bisected to determine z_n, then $f(z_n)$ is calculated to determine whether the root is in the right or the left interval. This gives a reduced interval. The procedure is repeated until the root has been located within the desired accuracy.

REFERENCES

1. Katsanis, Theodore: FORTRAN Program for Calculating Transonic Velocities on a Blade-to-Blade Stream Surface of a Turbomachine. NASA TN D-5427, 1969.

2. Katsanis, Theodore; and McNally, William D.: Revised FORTRAN Program for Calculating Velocities and Streamlines on a Blade-to-Blade Stream Surface of a Turbomachine. NASA TM X-1764, 1969.

3. Katsanis, Theodore; and McNally, William D.: FORTRAN Program for Calculating Velocities and Streamlines on a Blade-to-Blade Stream Surface of a Tandem Blade Turbomachine. NASA TN D-5044, 1969.

4. Walsh, J. L.; Ahlberg, J. H.; and Nilson, E. N.: Best Approximation Properties of the Spline Fit. J. Math. Mech., vol. 11, no. 2, 1962, pp. 225-234.

"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.

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 conference proceedings, 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 Notes, and Technology Surveys.

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

SCIENTIFIC AND TECHNICAL INFORMATION DIVISION

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Washington, D.C. 20546