NASA Contractor Report 4506

Coupled 2-Dimensional Cascade Theory for Noise and Unsteady Aerodynamics of Blade Row Interaction in Turbofans

Volume 2–Documentation for Computer Code CUP2D

Donald B. Hanson
United Technologies Corporation
Pratt and Whitney Division
East Hartford, Connecticut

Prepared for the
Lewis Research Center
Under Contract NAS3–25952
January 1994

National Aeronautics and
Space Administration

Office of Management

Scientific and Technical
Information Program

1994

TABLE OF CONTENTS

Sections **Page**

Summary

A 2D linear aeroacoustic theory for rotor/stator interaction with unsteady coupling was derived and explored in Volume 1 of this report. Computer program CUP2D has been written in FORTRAN embodying the theoretical equations. This volume (Volume 2) describes the structure of the code, installation and running, preparation of the input file, and interpretation of the output. A sample case is provided with printouts of the input and output. The source code is included with comments linking it closely to the theoretical equations in Volume 1.

Section 1
Introduction

This volume provides documentation and user information for the coupled 2D linearized cascade code CUP2D. Theory for the code is derived and explored in Volume 1 of this report. Material herein discusses how to install and run the code, explains the input file and the printed output, outlines the code structure, and provides a listing of the source code.

CUP2D is written as strictly as possible in FORTRAN 77 and is self-contained so that no system subroutines are needed. One exception is that the DOUBLE COMPLEX variable type is used; this should be accepted by any modern compiler. The only other exception is that the subroutine TIMDAT calls a system-dependent time and date function. This has been found to work on Sun™ and Iris™ systems but can be deleted or modified by the user, if necessary. Figure 1 shows the hierarchy of subroutines with a brief description of the subroutine functions. More description can be found in the subroutine comments. To interpret the figure 1, note that each routine calls only those routines indented underneath. Thus, for example, READIN calls only ALFBET, RTCOEF, GTWAKE, and PRNTIN. Each routine is called only once with the exception of GETVS and DSWK, as shown in figure 1.

The entire code is supplied on disk in a single module (or file) called *cup2d.f* and can be compiled on a UNIX™ system by entering *f77 -o cup2d cup2d.f*. This generates an executable file which can be run by typing *cup2d*. The code then looks for the input file *cupin.dat*, which must be in the same directory as *cup2d*. Normally, the output is written to the screen only. However, if the user wishes the output written to a file, *cupout.dat* for example, the command *cup2d > cupout.dat* would be used.

Sections 2 and 3 give detailed descriptions of the input and output. The source listing in Section 4 is heavily commented with descriptions of subroutine function at the top and throughout each routine. Also, to help in linking the code to the theory, variable names were chosen to be as close to the names used in the theory derivation as possible. Finally, wherever appropriate in the code, equation numbers from Volume 1 are given next to the corresponding FORTRAN statements.

HIERARCHY OF SUBROUTINES

PROGRAM CUP2D	main program
READIN	read input file (UNIT 8) and compute some constants
ALFBET	compute arrays of alphas and betas
RTCOEF	compute arrays of reflection and transmission coefficients
GTWAKE	compute upwash vectors
PRNTIN	print input
TIMDAT	print time and date of execution
INFNS	compute elements of KMATRX
RCOEFS	rotor on stator effect
GETVS	compute Smith's v_1' , v_2' , v_3'
SCOEFS	stator on rotor effect
GETVS	compute Smith's v_1' , v_2' , v_3'
GENKRR	rotor on rotor effect
DSWK & WAVE .	Smith's routines for matrix elements
GENKSS	stator on stator effect
DSWK & WAVE .	Smith's routines for matrix elements
SOLVE	solve coupled system for loading on both blade rows
MATINV	invert the matrix KMATRX
LINPAC Routines	
LOADS	compute loads from $[\text{KMATRX}]^{-1} \, \dot{*}$ WASH = LOAD
OUTPUT	print out sound pressure and sound power by mode
GETPWL	compute modal sound power

Figure 1. This figure indicates all subroutine calls. Except where shown, each routine is called only once.

Section 2
Explanation of Input File

The sample case input for code CUP2D is supplied on the same disk with the source listing in the file called *cupin.dat* , which is shown in figure 2. To facilitate verification of input, all of the input numbers are printed with the normal output. Brief definitions of the input are included at the bottom of the file. Some further explanations are provided below.

Line 1 - The comment is provided for user convenience and is printed on the 4^{th} line of output.

Line 2 - In the theory, a 2D cascade is considered to be wrapped into a narrow annulus to simulate a fan and to permit a mixture of fan nomenclature and cascade nomenclature. In particular, this permits numbers of blades to appear directly. The radius to the annulus provides a common dimension for the 2 blade rows which is considered to be the effective radius of the fan. It is used for non-dimensionalization in the axial spacing of the blade rows. In simulating a fan, the effective radius could be taken as 85 percent of the tip radius, in which case the rotor rotational Mach numbers at that radius would be input below in line 6.

Line 3 - The number of panels N_p is fixed to be the same for both blade rows. The number of harmonics N_h is the number used for the coupling equations and in the printout of sound pressure and sound power. The code is delivered with dimensioning for a max N_h=5 and a max N_p= 51. Section 5 gives array dimension information if this needs to be changed.

Line 4 - See figure 2 of this volume.

Line 5 - Speed of sound is in feet per second. Density is in pounds mass per cubic foot.

Line 6 - The code can treat counter-rotation configurations. In this case, input a positive rotational Mach number for the front rotor and a negative number for the rear. If either row is a stator, set its rotational Mach number to zero.

Line 7 - Here the user specifies the axial locations where he wants the modal sound pressure to be evaluated for the final table in the printout. Distances are measured downstream from the front row leading edge and are normalized by rotor effective radius.

Line 8 and 9 - For line 8, input the value of INTYPE to be used by the the subroutine GTWAKE in evaluating the upwash at the two blade rows for excitation of the system.

INTYPE = 1 is used to apply the Silverstein wake formulas derived in appendix E of Volume 1 of this report. Here, the user only specifies the drag coefficient on line 9 and the code computes upwash at N_p control points along the chord for each of N_h harmonics. This option was used for figure 15 of Volume 1.

INTYPE = 2 is the same as 1 except that the harmonics above BPF are set to zero. This is useful for evaluation of the frequency scattering effect and was used for most of the figures in Volume 1.

INTYPE = 3 also applies the formulas from appendix E of Volume 1 for computing upwash along the stator chord. However, in Equations E-12 and E-14, the absolute values of the wake harmonics (F_n in those equations) are input directly by the user in line 9. Since the wake amplitudes do not decay using this input, this is equivalent to specifying excitation by a vorticity wave. INTYPE 3 was used to check the Kousen/Verdon results in Section 4 of Volume 1.

INTYPE = 4 is provided so that the user can excite the rotor and stator with an upwash distribution of his own choosing. Thus, the upwash vectors WREXT(n,i) and WSEXT(n,i) that would computed from wake formulas using INTYPE 1 are input directly for harmonic order n and control point i. These are entered as real numbers in tabular form starting on line 9 as shown below for a two harmonic case.

Real[WREXT(1,1)]	Imag[WREXT(1,1)]
.	.
.	.
Real[WREXT(1,N_p)]	Imag[WREXT(1,N_p)]
Real[WREXT(2,1)]	Imag[WREXT(2,1)]
.	.
.	.
Real[WREXT(2,N_p)]	Imag[WREXT(2,N_p)]
Real[WSEXT(1,1)]	Imag[WSEXT(1,1)]
.	.
.	.
Real[WSEXT(1,N_p)]	Imag[WSEXT(1,N_p)]
Real[WSEXT(2,1)]	Imag[WSEXT(2,1)]
.	.
.	.
Real[WSEXT(2,N_p)]	Imag[WSEXT(2,N_p)]

This mode of input could be used to simulate excitation of one blade row by the potential field of the other or could be used to simulate blade vibration effects.

File *cupin.dat*

```
' Sample case for Code CUP2D, B=38, V=72'
38   72   .7628   .9567   .556
30    3
.318  .360   .427   0.231
1037.7  1070.0  1070.3  0.0293   0.0328  0.0326
.75     0.0
-0.217    0.86
2
.02
```

```
The above is input for a sample case for code CUP2D

Line 1   Comment - up to 70 characters - in single quotes

Line 2   # Blades-upstream row, # Blades-downstream row,
         gap/chord-1, gap/chord-2,
         axial dist LE1 to LE2 normalized by fan effective radius

Line 3   Number of panels each row, Number of harmonics

Line 4   Axial Mach number - upstream, inter-row, downstream
         Tangential Mach number - inter-row

Line 5   Speed of sound upstream, inter-row, downstream
         Density upstream, inter-row, downstream

Line 6   Rotational Mach number-front blade row (0 for IGV)
         Rotational Mach number- rear blade row (0 for EGV, negative # for
         rotor)

Line 7   Axial locations for acoustic pressure printout normalized by fan
         effective radius, measured positive downstream from front blade leading
         edge.

Line 8   1 or 2 for input based on drag coefficient.  2 sets the wake harmonics
         above BPF to zero.  See code documentation for other options.

Line 9   Drag coefficient
```

Figure 2. Input data set for sample case.

Section 3
Explanation of Code Output

Output for the sample case is shown in figure 3. Most of the input is printed on the first page. Subscripts *1* and *2* refer to the upstream and downstream blade rows respectively. Also, subscripts *a, b,* and *c* refer to the regions upstream of the upstream row, between rows, and downstream of the downstream row.

For the axial spacing of the blade rows, the user inputs the distance from between leading edges of the rows in fan effective radii. The code then computes and prints the axial distance from the upstream trailing edge to the downstream leading edge normalized by the upstream chord. Input values printed in the output include:

Mxa, Mxb, and Mxc - axial Mach numbers
Ms - swirl Mach number (in Region b)
My1, My2 - blade row rotational Mach numbers
RHOa, RHOb, and RHOc - densities
Aa, Ab, Ac - speeds of sound

Relative Mach numbers of the two blade rows *Mrel1 and Mrel2* are computed from the velocity triangles in figure 5 of Volume 1. Smith's reduced frequencies are based on full chord.

Flow angles *Theta 1 and Theta 2* and *Swirl Angle* are computed from the input Mach numbers. Note that *Theta 1* is normally negative per figure 5 of Volume 1.

The long table entitled "EXTERNAL VELOCITY IMPOSED ON CASCADE" gives the upwash values used as external excitation of the system. These are listed by harmonic order *N* and control point along the chord, counted by *I*. The control points are at Smith's unevenly spaced locations given by $z_i = 0.5*\{1 - \cos[\pi(2I-1)/(2N_p)]\}$. Values in the table become the vectors *WREXT* and *WSEXT* used in the call to the SOLVE routine.

After the listing of the upwash vectors, the printout in figure 3 shows the items "Entering RCOEFS" , and so forth to show the user how near execution is to completion. The "condition number" indicates whether the KMATRX is close to being singular.

"VALUES OF LIFT COEFFICIENTS" are the ΔC_p values computed in the LOADS routine integrated over the chords of each blade row.

In the final table showing modal sound pressures and sound powers, the FREQ column gives the value of $\Omega_{nk} = nB_1M_{y1} - kB_2M_{y2}$ and the mode column gives $nB_1 - kB_2$, which is defined so that positive values correspond to co-rotating modes. (Co-rotation implies a mode rotating in the direction of positive rotation of the blade rows and in the direction of positive swirl.) The cutoff ratios on the right are printed to help with diagnosis. For example, with the first harmonic (N=1) the K=1 mode can be seen to be cut on between the rotor and stator (region b) and cut off in the upstream and downstream regions. (Cutoff ratios larger than 9.99 are printed as 9.99.) This is the "trapped mode" discussed at length in Volume 1. It produces pressure but no power in regions a and b. Note that the downstream powers for n = 2 and 3 (namely 56.1 dB and 62.6 dB) can be found in the top part of figure 14 in Volume 1 for the rotor rotational Mach number = 0.75.

```
Code CUP2D for coupled cascade aeroacoustics - Version 1.1
Developed for NASA-Lewis by Pratt & Whitney under Contract NAS3-25952 - Task 10
Theory documented in NASA CR-4506, Volume I.

COMMENT: Sample case for Code CUP2D, B=38, V=72
Time of execution: Mon Mar  1 12:56:37 1993

B1= 38    B2= 72
Gap/Chord(1) =  0.763,  Gap/Chord(2) =  0.957
(Rotor LE to Stator LE)/(Local Radius of Rotor)= 0.556 (input)
Axial Spacing Between Blade Rows/Rotor Chord 1.9951    (computed)
Drag Coefficient = 0.020

Number of panels=30, Number of harmonics= 3

    Mxa     Mxb     Mxc      Ms     My1     My2   Mrel1  Mrel2
   0.318  0.360  0.427  0.231  0.750  0.000  0.632  0.428

    RHOa     RHOb     RHOc      Aa       Ab       Ac
  0.02930 0.03280 0.03260  1037.7  1070.0  1070.3

Remainder of printout is computed from input above
Smiths reduced freqs @ BPF front row,rear row =  18.532   6.078
Theta1, Theta2 (in degrees) =  -55.253  32.687
Swirl Angle (in degrees) =   32.69
Ambient Pressure/Sea Level STD (uptream, downstream) =  0.331  0.391

         EXTERNAL VELOCITY IMPOSED ON CASCADE
    N    I    WREXT(real, imag)      WSEXT(real, imag)
    1    1     0.0000  0.0000          0.0186  0.0261
    1    2     0.0000  0.0000          0.0194  0.0254
    1    3     0.0000  0.0000          0.0210  0.0241
    1    4     0.0000  0.0000          0.0232  0.0218
    1    5     0.0000  0.0000          0.0257  0.0186
    1    6     0.0000  0.0000          0.0282  0.0142
    1    7     0.0000  0.0000          0.0302  0.0087
    1    8     0.0000  0.0000          0.0312  0.0021
    1    9     0.0000  0.0000          0.0306 -0.0053
    1   10     0.0000  0.0000          0.0280 -0.0128
    1   11     0.0000  0.0000          0.0233 -0.0198
    1   12     0.0000  0.0000          0.0166 -0.0254
    1   13     0.0000  0.0000          0.0082 -0.0289
    1   14     0.0000  0.0000         -0.0011 -0.0298
    1   15     0.0000  0.0000         -0.0102 -0.0277
    1   16     0.0000  0.0000         -0.0182 -0.0229
    1   17     0.0000  0.0000         -0.0242 -0.0160
    1   18     0.0000  0.0000         -0.0277 -0.0078
    1   19     0.0000  0.0000         -0.0285  0.0009
    1   20     0.0000  0.0000         -0.0269  0.0089
    1   21     0.0000  0.0000         -0.0233  0.0158
    1   22     0.0000  0.0000         -0.0184  0.0210
    1   23     0.0000  0.0000         -0.0128  0.0246
    1   24     0.0000  0.0000         -0.0073  0.0266
    1   25     0.0000  0.0000         -0.0022  0.0273
    1   26     0.0000  0.0000          0.0021  0.0272
    1   27     0.0000  0.0000          0.0056  0.0266
    1   28     0.0000  0.0000          0.0082  0.0259
    1   29     0.0000  0.0000          0.0098  0.0253
    1   30     0.0000  0.0000          0.0107  0.0249
```

Figure 3 (beginning). Output for sample case.

2	1	0.0000	0.0000	0.0000	0.0000
2	2	0.0000	0.0000	0.0000	0.0000
2	3	0.0000	0.0000	0.0000	0.0000
2	4	0.0000	0.0000	0.0000	0.0000
2	5	0.0000	0.0000	0.0000	0.0000
2	6	0.0000	0.0000	0.0000	0.0000
2	7	0.0000	0.0000	0.0000	0.0000
2	8	0.0000	0.0000	0.0000	0.0000
2	9	0.0000	0.0000	0.0000	0.0000
2	10	0.0000	0.0000	0.0000	0.0000
2	11	0.0000	0.0000	0.0000	0.0000
2	12	0.0000	0.0000	0.0000	0.0000
2	13	0.0000	0.0000	0.0000	0.0000
2	14	0.0000	0.0000	0.0000	0.0000
2	15	0.0000	0.0000	0.0000	0.0000
2	16	0.0000	0.0000	0.0000	0.0000
2	17	0.0000	0.0000	0.0000	0.0000
2	18	0.0000	0.0000	0.0000	0.0000
2	19	0.0000	0.0000	0.0000	0.0000
2	20	0.0000	0.0000	0.0000	0.0000
2	21	0.0000	0.0000	0.0000	0.0000
2	22	0.0000	0.0000	0.0000	0.0000
2	23	0.0000	0.0000	0.0000	0.0000
2	24	0.0000	0.0000	0.0000	0.0000
2	25	0.0000	0.0000	0.0000	0.0000
2	26	0.0000	0.0000	0.0000	0.0000
2	27	0.0000	0.0000	0.0000	0.0000
2	28	0.0000	0.0000	0.0000	0.0000
2	29	0.0000	0.0000	0.0000	0.0000
2	30	0.0000	0.0000	0.0000	0.0000
3	1	0.0000	0.0000	0.0000	0.0000
3	2	0.0000	0.0000	0.0000	0.0000
3	3	0.0000	0.0000	0.0000	0.0000
3	4	0.0000	0.0000	0.0000	0.0000
3	5	0.0000	0.0000	0.0000	0.0000
3	6	0.0000	0.0000	0.0000	0.0000
3	7	0.0000	0.0000	0.0000	0.0000
3	8	0.0000	0.0000	0.0000	0.0000
3	9	0.0000	0.0000	0.0000	0.0000
3	10	0.0000	0.0000	0.0000	0.0000
3	11	0.0000	0.0000	0.0000	0.0000
3	12	0.0000	0.0000	0.0000	0.0000
3	13	0.0000	0.0000	0.0000	0.0000
3	14	0.0000	0.0000	0.0000	0.0000
3	15	0.0000	0.0000	0.0000	0.0000
3	16	0.0000	0.0000	0.0000	0.0000
3	17	0.0000	0.0000	0.0000	0.0000
3	18	0.0000	0.0000	0.0000	0.0000
3	19	0.0000	0.0000	0.0000	0.0000
3	20	0.0000	0.0000	0.0000	0.0000
3	21	0.0000	0.0000	0.0000	0.0000
3	22	0.0000	0.0000	0.0000	0.0000
3	23	0.0000	0.0000	0.0000	0.0000
3	24	0.0000	0.0000	0.0000	0.0000
3	25	0.0000	0.0000	0.0000	0.0000
3	26	0.0000	0.0000	0.0000	0.0000
3	27	0.0000	0.0000	0.0000	0.0000
3	28	0.0000	0.0000	0.0000	0.0000
3	29	0.0000	0.0000	0.0000	0.0000
3	30	0.0000	0.0000	0.0000	0.0000

Figure 3 (continued). Output for sample case.

```
Entering RCOEFS
Entering SCOEFS
Entering GENKRR
Entering GENKSS
Entering MATINV
Condition number of KMATRX =      4.3478710669121D-05

        ***  VALUES OF LIFT COEFFICIENTS ***
 N         CLROTOR(N)                CLSTATOR(N)
         real       imag           real       imag
 1    -0.00130   -0.00070        0.00270   -0.01780
 2     0.00009   -0.00004       -0.00029    0.00064
 3     0.00001    0.00003       -0.00032   -0.00051


 Axial locations for sound pressure output in radii from rotor leading edge
For PRESup, Xa = -.217,  For PRESdn, Xc = 0.860

 Decibel Levels for Pressure Waves and Power Levels          |   Cutoff Ratios
     N     K  FREQ  nB1-kB2 PRESup  PRESdn   PWLup    PWLdn  |   A      B      C
     1    -1  28.50   110     0.0     0.0     0.0      0.0    0.28   0.03   0.29
     1     0  28.50    38     0.0     0.0     0.0      0.0    0.82   0.56   0.83
     1     1  28.50   -34    69.9    21.1     0.0      0.0    0.91   1.15   0.93
     1     2  28.50  -106     0.0     0.0     0.0      0.0    0.29   0.54   0.30
--------------     Total power for N= 1      0.0      0.0

     2    -3  57.00   292     0.0     0.0     0.0      0.0    0.21   0.04   0.22
     2    -2  57.00   220     0.0     0.0     0.0      0.0    0.28   0.03   0.29
     2    -1  57.00   148     0.0     0.0     0.0      0.0    0.42   0.17   0.43
     2     0  57.00    76     0.0     0.0     0.0      0.0    0.82   0.56   0.83
     2     1  57.00     4    98.7   108.9    41.0     56.1    9.99   9.99   9.99
     2     2  57.00   -68    31.6     0.0     0.0      0.0    0.91   1.15   0.93
     2     3  57.00  -140     0.0     0.0     0.0      0.0    0.44   0.68   0.45
--------------     Total power for N= 2     41.0     56.1

     3    -3  85.50   330     0.0     0.0     0.0      0.0    0.28   0.03   0.29
     3    -2  85.50   258     0.0     0.0     0.0      0.0    0.36   0.11   0.37
     3    -1  85.50   186     0.0     0.0     0.0      0.0    0.50   0.25   0.51
     3     0  85.50   114     0.0     0.0     0.0      0.0    0.82   0.56   0.83
     3     1  85.50    42   103.4   116.3    45.4     62.4    2.21   1.93   2.25
     3     2  85.50   -30    88.9   102.1    31.1     48.7    3.10   3.30   3.15
     3     3  85.50  -102     6.5     0.0     0.0      0.0    0.91   1.15   0.93
--------------     Total power for N= 3     45.6     62.6
```

Figure 3. (concluded) Output for sample case.

Section 4
Source Code Listing

The remainder of this volume gives the listing of the routines shown in figure 1. The routines can be categorized as follows. The first group compises new code as described in Volume 1. Then there are two routines taken verbatum from the Smith code: DSWK and WAVE. Finally, there is a series of routines for inverting the (double precision, real) matrix of influence coefficients. These were taken from LINPAC (see LINPAC User's Guide, SIAM, Philadelphia, 1979) and are not shown here since they are in the public domain and are commonly available. Of course, they are included on the disk with the rest of the source code. The routine MATINV is included, since this was written for the present purposes to call the LINPAC routines.

```
      PROGRAM CUP2D

c.. Calculates the unsteady loading and associated acoustic and vorticity waves
c.. on 2 mutually interacting blade rows in a 2 dimensional, linear, subsonic
c.. analysis.  Blade rows can be rotor/egv, igv/rotor, or rotor/rotor, depending
c.. input values of rotor rotational Mach numbers, MY1 and MY2.
c.. Simultaneous solution for flow tangency for 2 blade rows, NH harmonics, and
c.. NP panels via inversion of the matrix coupling equation KMATRX*LOAD=WASH.
c.. A disturbance upwash distribution at either,or both, blade rows is generated
c.. either from direct user input or from Silverstein's wake formulas.  The code
c.. code then finds the unsteady loading (LOAD) that produces an upwash (WASH)
c.. that just cancels the disturbance wash.  These loads are used to find the
c.. acoustic waves and the sound power.  Blade row self-effect is computed via
c.. subroutines from Smith Code.  Effect of each row on the other is computed
c.. via an extension of Smith's theory by D.B. Hanson. Reflections at inlet and
c.. exit interfaces are treated with reflection and transmission coefficients
c.. derived from continuity of mass & momentum based on an actuator disk model.
c.. Overall theory documented in NASA CR-4506,Volume I. Comments in this listing
c.. refer to equation numbers in the same Contractor Report. Coding by Hanson.

c.. This routine is the main program.

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION MXA,MXB,MXC,MY1,MY2,MS,LAM1,LAM2,
     >                                        KMATRX(1020,1020)
      DOUBLE COMPLEX WREXT(5,51), WSEXT(5,51), LR(5,51), LS(5,51),
     > ALF(9,-5:5,-5:5),
     > KRUP(5,-5:5,51),KRDN(5,-5:5,51),KSUP(5,-5:5,51),KSDN(5,-5:5,51),
     > R12(-5:5,-5:5), R21(-5:5,-5:5),R13(-5:5,-5:5), R31(-5:5,-5:5),
     > T14(-5:5,-5:5), T28(-5:5,-5:5),T38(-5:5,-5:5)
      INTEGER B1, B2, BETA(-5:5,-5:5)

cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      WRITE(*,*) ' '
      WRITE(*,*)'Code CUP2D for coupled cascade aeroacoustics - Version
     > 1.0 '
      WRITE(*,*)'Developed for NASA-Lewis by Pratt & Whitney under Cont
     >ract NAS3-25952 - Task 10'
      WRITE(*,*)'Theory documented in NASA CR-4506, Volume I'
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

c.. Read input, generate wavenumbers and reflection coefficients, establish
c.. external disturbance vectors from wake formulas or direct input, and
c.. print input geometry and flow conditions.
      CALL READIN(NH,NP,B1,B2,C1,C2,SC1,SC2,CT1,CT2,ST1,ST2,XS,XA,XC,
     > MXA,MXB,MXC,MS,MY1,MY2,LAM1,LAM2,ALF,BETA,R12,R21,R13,R31,
     > T14,T28,T38, AA,AB,AC, WREXT,WSEXT,POPSA,POPSC   )

c.. Generate matrices of influence functions
      CALL INFFNS(NH,NP,B1,B2,SC1,SC2,CT1,CT2,ST1,ST2,XS,
     >     MXB,LAM1,LAM2,ALF,BETA,R12,R21,R13,R31,T14,T28,T38,
     >     KMATRX,KRUP,KRDN,KSUP,KSDN                )

c.. Solve coupled system of equations for loading by matrix inversion.
      CALL SOLVE(NH,NP,KMATRX,WREXT,WSEXT,LR,LS)

c.. Compute output waves and print their sound pressure and sound power.

      CALL OUTPUT(NH,NP,B1,B2,MXA,MXB,MXC,MS,MY1,MY2,ALF,AB/AA,AB/AC,
     >       XS,XA,XC,POPSA,POPSC,KRUP,KRDN,KSUP,KSDN,LR,LS)

      END
C ______________________________________________________________________
C
```

Figure 4. Source code for CUP2D.

```
      SUBROUTINE READIN(NH,NP,B1,B2,C1,C2,SC1,SC2,CT1,CT2,ST1,ST2,
     > XS,XA,XC,MXA,MXB,MXC,MS,MY1,MY2,LAM1,LAM2,ALF,BETA,
     >R12,R21,R13,R31,T14,T28,T38,AA,AB,AC,WREXT,WSEXT,POPSA,POPSC)

c.. Reads input from data set on disk (UNIT 8); generates constants for the
c.. Smith common block; and calls routines for alpha and beta wavenumbers,
c.. reflection and transmission coefficients, and external input velocity
c.. vectors.

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION  MXA,MXB,MXC,MS,MY1,MY2,LAM1,LAM2
      INTEGER B1, B2, BETA(-5:5,-5:5)
      DOUBLE COMPLEX WREXT(5,51),WSEXT(5,51),ALF(9,-5:5,-5:5),
     >    R12(-5:5,-5:5),R21(-5:5,-5:5),R13(-5:5,-5:5),R31(-5:5,-5:5),
     >    T14(-5:5,-5:5),T28(-5:5,-5:5),T38(-5:5,-5:5)
      CHARACTER*70 COMMENT

c.. Read and compute data for common block
      OPEN(UNIT=8,FILE='cupin.dat')

c.. Read comment
      READ(8,*) COMMENT

c.. Read geometry from disk file and compute normalized chords
      READ(8,*) B1, B2, SC1, SC2, XS
      C1 = 6.2831853D0/(B1*SC1)                     ! Chord/radius, front row
      C2 = 6.2831853D0/(B2*SC2)                     ! Chord/radius, back row

c.. Read number of panels and number of harmonics
      READ(8,*) NP, NH

c.. Read operating conditions from disk file.  1 & 2 refer to upstream and
c.. downstream blade rows.  A, B, & C refer to regions upstream of blade row 1,
c.. between blade rows, and downstream of blade row 2.  POPSA & POPSC = ambient
c.. pressure in upstream and downstream regions divided by sea level standard
c.. pressure.
      READ(8,*) MXA, MXB, MXC, MS                   ! Axial & swirl Mach #'s
      READ(8,*) AA,  AB,  AC,  RHOA, RHOB, RHOC     ! Sound speeds & densities
      READ(8,*) MY1, MY2                            ! Blade rotational Mach #'s

c.. Read x locations for pressure output. (x over radius from front row LE.)
      READ(8,*) XA, XC                 ! Make XA < 0 and XB > 0.

c.. Compute remaining items for passage to other routines
      ST1 = (-MY1+MS) / SQRT(MXB**2 + (-MY1+MS)**2)   ! Sine(theta1)
      ST2 = (-MY2+MS) / SQRT(MXB**2 + (-MY2+MS)**2)   ! Sine(theta2)
      CT1 = SQRT(1D0 - ST1**2)                        ! Cosine(theta1)
      CT2 = SQRT(1D0 - ST2**2)                        ! Cosine(theta2)

      LAM1 = B2*(MY1-MY2)/(MXB/CT1)                   ! Reduced freq, front row
      LAM2 = B1*(MY1-MY2)/(MXB/CT2)                   ! Reduced freq, back row

c.. Compute axial wavenumbers (alpha's) and tangential wavenumbers (beta's)
      CALL ALFBET(B1,B2,NH,MXA,MXB,MXC,MS,MY1,MY2,
     >                                    AA,AB,AC,ALF,BETA)

c.. Get reflection and transmission coefficients
      CALL RTCOEF(NH,B1,B2,ALF,BETA,AA,AB,AC,RHOA,RHOB,RHOC,
     >          MXA,MXB,MXC,MS,MY1,MY2, R12,R21,R13,R31, T14,T28,T38)

c.. Read wake input data and compute upwash vectors. Leave unit 8 open to
c.. read from GTWAKE.
      CALL GTWAKE(NH,NP,C1,C2,SC1,CT1,CT2,ST1,ST2,XS,INTYPE,CD,
     > WREXT,WSEXT)

      CLOSE(8)
```

Figure 4. (continued) Source code for CUP2D.

```
c.. Compute Pambient/Pstandard to be used in OUTPUT for SPL's
      POPSA = RHOA*AA**2/(1.4*32.2*2116)
      POPSC = RHOC*AC**2/(1.4*32.2*2116)

c.. Print input data
      CALL PRNTIN(NH,NP,B1,B2,C1,C2,SC1,SC2,CT1,CT2,ST1,ST2,XS,
     > AA,AB,AC,RHOA,RHOB,RHOC,MXA,MXB,MXC,MS,MY1,MY2,
     > LAM1,LAM2,COMMENT,INTYPE,CD,WREXT, WSEXT,POPSA,POPSC )

      RETURN
      END
C ______________________________________________________________________
C
```

Figure 4. (continued) Source code for CUP2D.

```
      SUBROUTINE ALFBET(B1,B2,NH,MXA,MXB,MXC,MS,MY1,MY2,
     >                                AA,AB,AC,ALF,BETA)
c.. Computes alpha and beta wavenumbers from formulas derived in appendix B.
c.. The alphas are Smith's normalized by source radius R rather than by chord.
c.. Prints message if a resonance condition occurs for any combination of n & k.

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION MXA,MXB,MXC,MS,MY1,MY2
      INTEGER B1,B2,BETA(-5:5,-5:5)
      DOUBLE COMPLEX ALF(9,-5:5,-5:5)

           DENOMA = 1.0D0 - MXA**2
           DENOMB = 1.0D0 - MXB**2
           DENOMC = 1.0D0 - MXC**2
      DO 100 N = -NH, NH
      DO 100 K = -NH, NH
           IF (ABS(N)+ABS(K) .EQ. 0) GOTO 100
           BETA(N,K) = - (N*B1 - K*B2)
           OMEGA = N*B1*MY1 - K*B2*MY
           EA = DENOMA*BETA(N,K)**2 - (OMEGA*AB/AA)**2
           EB = DENOMB*BETA(N,K)**2 - (OMEGA + BETA(N,K)*MS)**2
           EC = DENOMC*BETA(N,K)**2 - (OMEGA*AB/AC)**2
           FA = MXA*OMEGA*AB/AA
           FB = MXB*(OMEGA + BETA(N,K)*MS)
           FC = MXC*OMEGA*AB/AC

c.. Check for resonance in upstream region, swirl region, and downstream region
c.. E=0 for resonance, E < 0 for propagation, E > 0 for decay.  Then compute
c.. alphas for pressure waves.  Use Eqs. B-8,9,10 in Region B and variations
c.. for Regions A and C.

c.. Do Region A first (upstream of swirl region).
           IF (EA .EQ. 0.0) THEN
              WRITE(*,1) N,K
1            FORMAT(1X,'Resonance in Region A for N=',I3,'  K=',I3)
              STOP 'Execution terminated due to resonance'
           ELSE IF (EA .LT. 0.0) THEN
              ALF(4,N,K) = (FA + SIGN(SQRT(-EA), FA))/DENOMA
              ALF(5,N,K) = (FA - SIGN(SQRT(-EA), FA))/DENOMA
           ELSE
              ALF(4,N,K) = CMPLX(FA, -SQRT(EA))/DENOMA
              ALF(5,N,K) = CMPLX(FA, +SQRT(EA))/DENOMA
           ENDIF

c.. Do Region B next (swirl region).
           IF (EB .EQ. 0.0) THEN
              WRITE(*,2) N,K
2            FORMAT(1X,'Resonance in Region B for N=',I3,'  K=',I3)
              STOP 'Execution terminated due to resonance'
           ELSE IF (EB .LT. 0.0) THEN
              ALF(1,N,K) = (FB + SIGN(SQRT(-EB), FB))/DENOMB
              ALF(2,N,K) = (FB - SIGN(SQRT(-EB), FB))/DENOMB
           ELSE
              ALF(1,N,K) = CMPLX(FB, -SQRT(EB))/DENOMB
              ALF(2,N,K) = CMPLX(FB, +SQRT(EB))/DENOMB
           ENDIF

c.. Finally, do Region C (downstream of swirl region)
           IF (EC .EQ. 0.0) THEN
              WRITE(*,3) N,K
3            FORMAT(1X,'Resonance in Region C for N=',I3,'  K=',I3)
              STOP 'Execution terminated due to resonance'
           ELSE IF (EC .LT. 0.0) THEN
              ALF(7,N,K) = (FC + SIGN(SQRT(-EC), FC))/DENOMC
```

Figure 4. (continued) Source code for CUP2D.

```
            ALF(8,N,K) = (FC - SIGN(SQRT(-EC), FC))/DENOMC
         ELSE
            ALF(7,N,K) = CMPLX(FC, -SQRT(EC))/DENOMC
            ALF(8,N,K) = CMPLX(FC, +SQRT(EC))/DENOMC
         ENDIF

c.. Wavenumbers for vorticity waves in Region B from eq. B-11 and variations
c.. for Regions A and C.
      ALF(3,N,K) = -(OMEGA + MS*BETA(N,K))/MXB
      ALF(6,N,K) = - OMEGA /MXA
      ALF(9,N,K) = - OMEGA /MXC

100   CONTINUE

      RETURN
      END
C ______________________________________________________________________
C
```

Figure 4. (continued) Source code for CUP2D.

```
      SUBROUTINE RTCOEF(NH,B1,B2,ALF,BETA,AA,AB,AC,RHOA,RHOB,RHOC,
     >           MXA,MXB,MXC,MS,MY1,MY2, R12,R21,R13,R31, T14,T28,T38)
c.. This subroutine computes reflection and transmission coefficients for the
c.. inlet and exit for the transverse velocity component. Coefficients derived
c.. in NASA CR-4506, Volume I, appendix D.

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION MXA,MXB,MXC,MS,MY1,MY2
      INTEGER B1,B2,BET,BETA(-5:5,-5:5)
      DOUBLE COMPLEX C1,C2,C3,C4,C8,C9,F1,F2,F3,F4,F8,F9,
     > G1,G2,G3,G4,G8,G9, E0, E1, E2, E3, E4, E5, E6, E7, E8, E9, E10,
     >         E11,E12,E13,E14,E15,E16,E17,E18,E19,E20,E21
      DOUBLE COMPLEX ALF(9,-5:5,-5:5),
     > R12(-5:5,-5:5), R21(-5:5,-5:5), R13(-5:5,-5:5), R31(-5:5,-5:5),
     > T14(-5:5,-5:5), T28(-5:5,-5:5), T38(-5:5,-5:5)

c.. Compute rho*c0 for 3 regions
      ROCA = RHOA*AA
      ROCB = RHOB*AB
      ROCC = RHOC*AC

C.. Compute reflection and transmission coefficients
      DO 10 N = -NH, NH
      DO 10 K = -NH, NH
         IF ((ABS(N)+ABS(K)) .EQ. 0) GO TO 10
         BET   = BETA(N,K)
         OMEGA = N*B1*MY1 - K*B2*MY2

C.. Coefficients from continuity equations
       C1 = ROCB*((1D0-MXB**2)*ALF(1,N,K) - MXB*(OMEGA+MS*BET))
       C2 = ROCB*((1D0-MXB**2)*ALF(2,N,K) - MXB*(OMEGA+MS*BET))
       C3 = ROCB*(-BET)
       C4 = ROCA*((1D0-MXA**2)*ALF(4,N,K) - MXA*OMEGA*AB/AA)
       C8 = ROCC*((1D0-MXC**2)*ALF(8,N,K) - MXC*OMEGA*AB/AC)
       C9 = ROCC*(-BET)

C.. Coefficients from axial momentum equations
       F1=ROCB*((1D0+MXB**2)*(OMEGA+MS*BET)-(1D0-MXB**2)*MXB*ALF(1,N,K))
       F2=ROCB*((1D0+MXB**2)*(OMEGA+MS*BET)-(1D0-MXB**2)*MXB*ALF(2,N,K))
       F3=ROCB*(2D0*MXB*BET)
       F4=ROCA*((1D0+MXA**2)* OMEGA - (1D0-MXA**2)*MXA*ALF(4,N,K)*AA/AB)
       F8=ROCC*((1D0+MXC**2)* OMEGA - (1D0-MXC**2)*MXC*ALF(8,N,K)*AC/AB)
       F9=ROCC*(2D0*MXC*BET*AC/AB)

C.. Coefficients from transverse momentum equations
      G1=ROCB*(MXB*BET-MXB*MS*(OMEGA+MS*BET)+(1D0-MXB**2)*MS*ALF(1,N,K))
      G2=ROCB*(MXB*BET-MXB*MS*(OMEGA+MS*BET)+(1D0-MXB**2)*MS*ALF(2,N,K))
      G3= ROCB*(MXB*ALF(3,N,K)-MS*BET)
      G4= ROCA*(MXA*BET*AA/AB)
      G8= ROCC*(MXC*BET*AC/AB)
      G9= ROCC*(MXC*ALF(9,N,K)*AC/AB)

       E0 = C4*F1 - C1*F4
       E1 = C4*F2 - C2*F4
       E2 = C3*F4 - C4*F3
       E3 = C4*G1 - C1*G4
       E4 = C4*G2 - C2*G4
       E5 = C3*G4 - C4*G3
       E6 = C2*G1 - C1*G2
       E7 = C3*G2 - C2*G3
       E8 = C2*F1 - C1*F2
       E9 = C3*F2 - C2*F3
       E10= C1*F8 - C8*F1
```

Figure 4. (continued) Source code for CUP2D.

```
      E11= C9*F1  - C1*F9
      E12= C1*G8  - C8*G1
      E13= C9*G1  - C1*G9
      E14= C8*F2  - C2*F8
      E15= C8*F9  - C9*F8
      E16= C8*G2  - C2*G8
      E17= C8*G9  - C9*G8
      E18= C8*F3  - C3*F8
      E19= C8*G3  - C3*G8
      E20= C1*F3  - C3*F1
      E21= C1*G3  - C3*G1

C.. Reflection coefficients
      R12(N,K) = (E2*E3-E0*E5)      / (E1*E5-E2*E4)
      R13(N,K) = (E1*E3-E0*E4)      / (E1*E5-E2*E4)
      R21(N,K) = (E15*E16-E14*E17)  / (E12*E15-E10*E17)
      R31(N,K) = (E15*E19-E17*E18)  / (E12*E15-E10*E17)

C.. Transmission coefficients
      T14(N,K) = (E7*E8-E6*E9)      / (E4*E9-E1*E7)
      T28(N,K) = (E6*E11-E8*E13)    / (E10*E13-E11*E12)
      T38(N,K) = (E11*E21-E13*E20)  / (E11*E12-E10*E13)

10    CONTINUE

      RETURN
      END
C ------------------------------------------------------------------------
C
```

Figure 4. (continued) Source code for CUP2D.

```
      SUBROUTINE GTWAKE(NH,NP,C1,C2,SC1,CT1,CT2,ST1,ST2,XS,
     > INTYPE, CD, WREXT,WSEXT)
c.. This routine reads data from the input file and generates the upwash vectors
c.. WREXT and WSEXT, representing external excitation of the system.  The
c.. disturbance can be described via various optional methods specified by the
c.. input value of INTYPE as follows.
c.. INTYPE = 1 is used to represent viscous wakes via the Silverstein formulas.
c.. The wake is specified by the drag coefficient CD, i.e. by only one number.
c.. The upwash vectors are then computed from the formulas in appendix E.
c.. INTYPE = 2 is the same as INTYPE 1 except that the wake harmonics above BPF
c.. are set to zero.
c.. INTYPE = 3 is the same as INTYPE 1 except that the velocity defect harmonics
c.. are input by the user rather than being computed from the wake formulas.
c.. For INTYPE = 4 the real and imaginary parts of WREXT and WSEXT are simply
c.. read from the file. Here the upwash vectors are completely specifed by the
c.. user for all harmonics: N = 1...NH and all control points I = 1...NP.

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION FW(5)
      DOUBLE COMPLEX EXPON, WREXT(5,51), WSEXT(5,51), AI

      PI     = 3.14159265D0
      AI     = (0.0D0,1.0D0)

c.. Read INTYPE to identify type of input disturbance
      READ(8,*) INTYPE
      IF (INTYPE .EQ. 1) THEN

c.. For INTYPE=1, read drag coefficient and use Silverstein formulas as given
c.. in appendix E.
             READ(8,*) CD              ! Drag coefficient
          DO 34 N = 1, NH
             DO 34 I = 1, NP
                Z2I = 0.5D0*(1D0-COS(PI*(2.0D0*I-1D0)/(2.0D0*NP)))
                Z1  = (XS/C1 + Z2I/(C1/C2)*CT2)/CT1
                WCOW1 = 1.21D0*SQRT(CD)/(Z1 - 0.70D0)
                YOC1 = 0.68D0*SQRT(CD*(Z1 - 0.85D0))
                Q = 1.77245D0*SC1/YOC1*CT1
                FN = 3.54491D0/Q*WCOW1*EXP(-(PI*N/Q)**2)
                EXPON =EXP(AI*2D0*PI*N*(
     >           (CT2*ST1/CT1-ST2)*(C2/C1)/SC1*Z2I + (XS/C1)/SC1*ST1/CT1))
                WSEXT(N,I) = CT2/CT1*(ST2*CT1-CT2*ST1)*FN*EXPON
                WREXT(N,I) = (0D0, 0D0)
34        CONTINUE

c.. For INTYPE=2, read drag coefficient and use Silverstein formulas as given
c.. in appendix E.  BUT, for harmonic order > BPF, set upwash to zero.
      ELSE IF (INTYPE .EQ. 2) THEN
             READ(8,*) CD              ! Drag coefficient
          DO 24 N = 1, NH
             DO 24 I = 1, NP
                Z2I = 0.5D0*(1D0-COS(PI*(2.0D0*I-1D0)/(2.0D0*NP)))
                Z1  = (XS/C1 + Z2I/(C1/C2)*CT2)/CT1
                WCOW1 = 1.21D0*SQRT(CD)/(Z1 - 0.70D0)
                YOC1 = 0.68D0*SQRT(CD*(Z1 - 0.85D0))
                Q = 1.77245D0*SC1/YOC1*CT1
                FN = 3.54491D0/Q*WCOW1*EXP(-(PI*N/Q)**2)
                EXPON =EXP(AI*2D0*PI*N*(
     >           (CT2*ST1/CT1-ST2)*(C2/C1)/SC1*Z2I + (XS/C1)/SC1*ST1/CT1))
                WSEXT(N,I) = CT2/CT1*(ST2*CT1-CT2*ST1)*FN*EXPON
                   IF (N .GT. 1) WSEXT(N,I) = (0D0, 0D0)
                WREXT(N,I) = (0D0, 0D0)
24        CONTINUE
```

Figure 4. (continued) Source code for CUP2D.

```
c.. For INTYPE=3, read harmonics of an upwash that convects with the mean flow.
c,, This is just like INTYPE 1 above except that FW(N) is read from input here
c.. and is independent of x.  By contrast, for INTYPE = 1 above, FN is computed
c.. from the Silverstein formulas as a function of chordwise position on the
c.. downstream blade row.  To interpret these formulas, see appendix E.
      ELSE IF (INTYPE .EQ. 3) THEN
              READ(8,*) (FW(N),N=1,NH)
           DO 14 N = 1, NH
             DO 14 I = 1, NP
               Z2I = 0.5D0*(1D0 - COS(PI*(2.0D0*I-1D0)/(2.0D0*NP)))
               EXPON =EXP(AI*2D0*PI*N*(
     >          (CT2*ST1/CT1-ST2)*(C2/C1)/SC1*Z2I + (XS/C1)/SC1*ST1/CT1))
               WSEXT(N,I) = CT2/CT1*(ST2*CT1-CT2*ST1)*FW(N)*EXPON
               WREXT(N,I) = (0D0, 0D0)
14        CONTINUE

c.. For INTYPE=4, read real and imaginary parts of vectors of external velocity
c.. disturbance as direct input.  Rotor input first, then stator.
      ELSE IF (INTYPE .EQ. 4) THEN
           DO 10 N=1,NH
           DO 10 I=1,NP
              READ(8,*) WREXTR, WREXTI
              WREXT(N,I)=CMPLX(WREXTR,WREXTI)
10        CONTINUE
           DO 12 N=1,NH
           DO 12 I=1,NP
              READ(8,*) WSEXTR,WSEXTI
              WSEXT(N,I)=CMPLX(WSEXTR,WSEXTI)
12        CONTINUE

      ELSE
         STOP 'Input type (INTYPE) for WSEXT AND WREXT not defined'
      ENDIF

      RETURN
      END
C ----------------------------------------------------------------------
C
```

Figure 4. (continued) Source code for CUP2D.

```
      SUBROUTINE PRNTIN(NH,NP,B1,B2,C1,C2,SC1,SC2,CT1,CT2,ST1,ST2,XS,
     > AA,AB,AC,RHOA,RHOB,RHOC,MXA,MXB,MXC,MS,MY1,MY2,
     > LAM1,LAM2,COMMENT, INTYPE,CD,WREXT, WSEXT,POPSA,POPSC)
c.. This routine prints the input data (some of it manipulated) to the screen.
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION  MXA,MXB,MXC,MS,MY1,MY2,LAM1,LAM2
      INTEGER B1,B2
      DOUBLE COMPLEX WREXT(5,51), WSEXT(5,51)
      CHARACTER*70 COMMENT

c.. Print input data and computed quantities
      WRITE(*,*) ' '
      WRITE(*,*) 'COMMENT: ',COMMENT
      CALL TIMDAT                          ! WRITES TIME AND DATE OF EXECUTION
      WRITE(*,*) ' '
      WRITE(*,11) B1, B2
11    FORMAT(1X,'B1=',I3,'     B2=',I3)
      WRITE(*,13) SC1, SC2
13    FORMAT(1X,'Gap/Chord(1) = ',F6.3,',  Gap/Chord(2) = ',F6.3)
      WRITE(*,31) XS
31    FORMAT(1X,'(Rotor LE to Stator LE)/(Local Radius of Rotor)=',F6.3,
     >' (input)')
      WRITE(*,35) (XS-C1*CT1)/C1
35    FORMAT(1X,'Axial Spacing Between Blade Rows/Rotor Chord',F7.4,'
     > (computed) ')
      IF (INTYPE .LT. 3) WRITE(*,1) CD
1     FORMAT(1X,'Drag Coefficient = ',F5.3)
      WRITE(*,*) ' '
      WRITE(*,15) NP, NH
15    FORMAT(1X,'Number of panels=',I2,', Number of harmonics=',I2)
      WRITE(*,*) ' '
      WRITE(*,*)'    Mxa    Mxb    Mxc    Ms    My1    My2   Mrel1  Mre
     >l2 '
      WRITE(*,17) MXA, MXB ,MXC, MS, MY1, MY2, MXB/CT1, MXB/CT2
17    FORMAT(1X,8F7.3)
      WRITE(*,*) ' '
      WRITE(*,*) '    RHOa    RHOb    RHOc     Aa      Ab      Ac '
      WRITE(*,19) RHOA, RHOB, RHOC, AA, AB, AC
19    FORMAT(1X, 3F8.5, 3F8.1)
      WRITE(*,*) ' '
      WRITE(*,*) 'Remainder of printout is computed from input above'
      WRITE(*,21) LAM1*C1, LAM2*C2
21    FORMAT(1X,'Smiths reduced freqs @ BPF front row,rear row =',2F8.3)
      WRITE(*,27) 57.29578*ASIN(ST1), 57.29578*ASIN(ST2)
27    FORMAT(1X,'Theta1, Theta2 (in degrees) = ', 2F8.3)
      WRITE(*,29) 57.29578*ATAN(MS/MXB)
29    FORMAT(1X,'Swirl Angle (in degrees) = ', F7.2)
      WRITE(*,16) POPSA, POPSC
16    FORMAT(1X,'Ambient Pressure/Sea Level STD (uptream, downstream) ='
     > ,2F7.3)
      WRITE(*,*)' '
      WRITE(*,*) '       EXTERNAL VELOCITY IMPOSED ON CASCADE'
      WRITE(*,*) '   N   I   WREXT(real, imag)     WSEXT(real, imag)'
      DO 20 N=1,NH
      DO 22 I=1,NP
      WRITE(*,33) N,I,WREXT(N,I),WSEXT(N,I)
33    FORMAT(1X,2I4,'  ',2F8.4,'        ',2F8.4)
22    CONTINUE
      WRITE(*,*)' '
20    CONTINUE
      WRITE(*,*) ' '

      RETURN
      END
C ______________________________________________________________________
```

Figure 4. (continued) Source code for CUP2D.

```
      SUBROUTINE TIMDAT         ! This routine is specific to the Sun Workstations
      CHARACTER*24 FDATE        ! Modify this routine for other computers
      WRITE(*,*) 'Time of execution: ',FDATE()
      END
C ____________________________________________________________________________
C
```

Figure 4. (continued) Source code for CUP2D.

```
      SUBROUTINE INFFNS(NH,NP,B1,B2,SC1,SC2,CT1,CT2,ST1,ST2,XS,
     >    MXB,LAM1,LAM2,ALF,BETA,R12,R21,R13,R31,T14,T28,T38,
     >    KMATRX,KRUP,KRDN,KSUP,KSDN          )
c.. Calls routines to compute elements of KMATRX, the matrix of influence
c.. functions.  KMATRX is then returned to the main program.  Algebra is
c.. based on NASA CR-4506, Volume I, Section 3.3.

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION  MXB, LAM1, LAM2, KMATRX(1020,1020)
      INTEGER  B1, B2, BETA(-5:5,-5:5)
      DOUBLE COMPLEX  ALF(9,-5:5,-5:5),
     >   R12(-5:5,-5:5), R21(-5:5,-5:5), R13(-5:5,-5:5), R31(-5:5,-5:5),
     >   T14(-5:5,-5:5), T28(-5:5,-5:5), T38(-5:5,-5:5),
     >   KRUP(5,-5:5,51),KRDN(5,-5:5,51),KSUP(5,-5:5,51),KSDN(5,-5:5,51)

      C1 = 6.2831853D0/(B1*SC1)          ! chord/radius, row # 1
      C2 = 6.2831853D0/(B2*SC2)          ! chord/radius, row # 2


C.. Zero the Kmatrix before starting to compute the elements
      DO 10 MU = 1, 4*NP*NH
      DO 10 NU = 1, 4*NP*NH
         KMATRX(MU,NU) = 0.0D0
10    CONTINUE

C.. Effect of rotor loading on stator
      CALL RCOEFS(NH,NP,C1,C2,SC1,CT1,CT2,ST1,ST2,XS,
     > MXB,LAM1,ALF,BETA,R12,R21,R13,R31,T14,T28,T38,KMATRX,KRUP,KRDN)

C.. Effect of stator loading on rotor
      CALL SCOEFS(NH,NP,C1,C2,SC2,CT1,CT2,ST1,ST2,XS,
     > MXB,LAM2,ALF,BETA,R12,R21,R13,R31,T14,T28,T38,KMATRX,KSUP,KSDN)

C.. Effect of rotor loading on rotor
      CALL GENKRR(NH,NP,C1,C2,SC1,SC2,CT1,ST1,MXB,LAM1,KMATRX   )

C.. Effect or stator loading on stator
      CALL GENKSS(NH,NP,C1,C2,SC1,SC2,CT2,ST2,MXB,LAM2,KMATRX   )

      RETURN
      END

C ____________________________________________________________________________
C
```

Figure 4. (continued) Source code for CUP2D.

```
      SUBROUTINE RCOEFS(NH,NP,C1,C2,SC1,CT1,CT2,ST1,ST2,XS,
     > MXB,LAM1,ALF,BETA,R12,R21,R13,R31,T14,T28,T38,KMATRX,KRUP,KRDN)

c.. Generates elements of the matrix of influence coefficients KMATRX that give
c.. the upwash caused by rotor loading at control points on the stator and
c.. rotor.  These are computed from KRS(N,K,I,J), effect of rotor on stator,
c.. and KRR'(N,I,J)1, effect of stator on stator. The prime on KRR' indicates
c.. that only the part of KRR associated with waves reflected from the actuator
c.. disk is computed here.  The remainder is computed in GENKRR using original
c.. routines from the Smith code.  N counts the rotor loading harmonics, I the
c.. control points, and J the load elements.  K counts the cascade wave index in
c.. the rotor frame which becomes the time harmonic index in the stator frame.

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION MXB,MR1,MR2,LAM1,KMATRX(1020,1020)
      INTEGER BET,BETA(-5:5,-5:5)
      DOUBLE COMPLEX AI,ALF1,ALF2,ALF3,EXPE21,EXPE31,EXPE1,
     > E1I,E2I,E3I,KR1,KR2,KR3,V1,V2,VR1,VR2,VR3,KRS,KRR,
     > ALF(9,-5:5,-5:5), KRUP(5,-5:5,51), KRDN(5,-5:5,51),
     > R12(-5:5,-5:5), R21(-5:5,-5:5), R13(-5:5,-5:5), R31(-5:5,-5:5),
     > T14(-5:5,-5:5), T28(-5:5,-5:5), T38(-5:5,-5:5)
      AI = (0.0D0, 1.0D0)
      PI = 3.14159265D0

      WRITE(*,*) 'Entering RCOEFS'
      MR1    = MXB/CT1                ! Relative Mach #, row 1
      MR2    = MXB/CT2                ! Relative Mach #, row 2
      CON    = MR1/SC1                ! A constant
      XE     = XS + C2*CT2            ! Axial coordinate of stator exit
      NPNH2 = NP*NH*2
      DO 10 N =   1, NH
      DO 10 K = -NH, NH
      BET = BETA(K,N)
      ALF1 = ALF(1,K,N)
      ALF2 = ALF(2,K,N)
      ALF3 = ALF(3,K,N)

      EXPE21     = EXP(AI*(ALF2-ALF1)*XE)
      EXPE31     = EXP(AI*(ALF3-ALF1)*XE)
      EXPE1      = EXP(-AI*ALF1*XE)

c.. Get Smith's factors for transverse velocity components
      CALL GETVS(BET,N*LAM1,MXB,ST1,CT1,V1,V2,V3)

      DO 10 J = 1, NP                              ! Loop on load elements
      Z0J=0.5D0*(1D0 - COS(PI*(J-1D0)/NP))        ! Chordwise locations for loads

      KR1 =  CON*V1*EXP(-AI*(ALF1*CT1+BET*ST1)*C1*Z0J)   !
      KR2 =  CON*V2*EXP(-AI*(ALF2*CT1+BET*ST1)*C1*Z0J)   ! eq. 43
      KR3 =  CON*V3*EXP(-AI*(ALF3*CT1+BET*ST1)*C1*Z0J)   !

c.. Compute VR1, VR2, VR3 from eq. 44.
c.. Hold EXP(-AI*ALF1*XE) out of VR1 now to avoid overflow in KRS later
      VR1 = (   (KR1*R12(K,N)+KR2)*R21(K,N)*EXP(AI*ALF2*XE)
     >        + (KR1*R13(K,N)+KR3)*R31(K,N)*EXP(AI*ALF3*XE)   )
     >          /(1D0-R12(K,N)*R21(K,N)*EXPE21-R13(K,N)*R31(K,N)*EXPE31)
      VR2 = (KR1+VR1*EXPE1)*R12(K,N)
      VR3 = (KR1+VR1*EXPE1)*R13(K,N)

c.. KRUP and KRDN are passed out of the subroutine for later use in computing
c.. pressure in the non-swirl regions upstream and downstream.
      KRUP(N,K,J) =     (KR1 + VR1*EXPE1)  * T14(K,N)          ! eq. 85
      KRDN(N,K,J) =     (KR2+VR2)*T28(K,N)*EXP(AI*ALF2*XE)     ! eq. 97
     >                + (KR3+VR3)*T38(K,N)*EXP(AI*ALF3*XE)
```

Figure 4. (continued) Source code for CUP2D.

```
c.. Loop on stator control points
      DO 10 I = 1, NP
      ZI=0.5D0*(1D0-COS(PI*(2D0*I - 1D0)/(2D0*NP)))
      E1I = (ALF1*CT2+BET*ST2)*C2*ZI
      E2I = (ALF2*CT2+BET*ST2)*C2*ZI
      E3I = (ALF3*CT2+BET*ST2)*C2*ZI

c.. Effect of rotor on stator, eq. 51
      IF (K .NE. 0) THEN
      KRS = 1/MR2 * (
     >    (BET*CT2-ALF1*ST2)*       VR1  *EXP(AI*(ALF1*(XS-XE)+E1I))
     > +  (BET*CT2-ALF2*ST2)*(KR2+VR2)*EXP(AI*(ALF2* XS     +E2I))
     > +  (ALF3*CT2+BET*ST2)*(KR3+VR3)*EXP(AI*(ALF3* XS     +E3I))   )

c..  Place elements in KMATRX, forming real elements, from complex KRS, eq. 69
      KMATRX(NPNH2+(2*ABS(K)-2)*NP+I,(2*N-2)*NP+J) =
     >KMATRX(NPNH2+(2*ABS(K)-2)*NP+I,(2*N-2)*NP+J)+REAL(KRS)
      KMATRX(NPNH2+(2*ABS(K)-2)*NP+I,(2*N-1)*NP+J) =
     >KMATRX(NPNH2+(2*ABS(K)-2)*NP+I,(2*N-1)*NP+J)-IMAG(KRS)
      KMATRX(NPNH2+(2*ABS(K)-1)*NP+I,(2*N-2)*NP+J) =
     >KMATRX(NPNH2+(2*ABS(K)-1)*NP+I,(2*N-2)*NP+J)+ISIGN(1,K)*IMAG(KRS)
      KMATRX(NPNH2+(2*ABS(K)-1)*NP+I,(2*N-1)*NP+J) =
     >KMATRX(NPNH2+(2*ABS(K)-1)*NP+I,(2*N-1)*NP+J)+ISIGN(1,K)*REAL(KRS)

      ENDIF

c.. Compute the portion of the KRR coefs caused by the reflected waves, eq. 46.
      KRR = 1/MR1*
     > (   (BET*CT1-ALF1*ST1)*VR1*EXP(AI*(ALF1*CT1+BET*ST1)*C1*ZI)*EXPE1
     >   + (BET*CT1-ALF2*ST1)*VR2*EXP(AI*(ALF2*CT1+BET*ST1)*C1*ZI)
     >   + (ALF3*CT1+BET*ST1)*VR3*EXP(AI*(ALF3*CT1+BET*ST1)*C1*ZI)       )

c.. Form real elements, do sum over K, and place in KMATRX.
         KMATRX((2*N-2)*NP+I,(2*N-2)*NP+J) =
     >   KMATRX((2*N-2)*NP+I,(2*N-2)*NP+J) + REAL(KRR)     ! eq. 62
         KMATRX((2*N-1)*NP+I,(2*N-2)*NP+J) =
     >   KMATRX((2*N-1)*NP+I,(2*N-2)*NP+J) + IMAG(KRR)     ! eq. 63

10    CONTINUE

c.. Fill in the remaining sections of the rotor-on-rotor quarter of the matrix
c.. from the second parts of Eqs. 62 and 63.
      DO 20 N = 1, NH
      DO 20 J = 1, NP
      DO 20 I = 1, NP
            KMATRX((2*N-1)*NP+I,(2*N-1)*NP+J) =
     >     KMATRX((2*N-2)*NP+I,(2*N-2)*NP+J)
            KMATRX((2*N-2)*NP+I,(2*N-1)*NP+J) =
     >    -KMATRX((2*N-1)*NP+I,(2*N-2)*NP+J)
20    CONTINUE

      RETURN
      END
c ------------------------------------------------------------------------
C
```

Figure 4. (continued) Source code for CUP2D.

```
      SUBROUTINE GETVS(BETA,NLAM,MX,ST,CT,V1,V2,V3)
c.. Generates V1 and V2 (Smith's v1'/beta and v2'/beta) and V3 (Smith's v3'/
c.. alpha3) for either rotor waves or stator waves.
c.. For stator waves, call with BETA(N,K),NLAM=N*LAM2=N*B1*(MY1-MY2)/MR2,MXB,
c.. ST=SIN(THETA2), CT=COS(THETA2).
c.. For rotor waves,  call with BETA(K,N),NLAM=N*LAM1=N*B2*(MY1-MY2)/MR1,MXB,
c.. ST=SIN(THETA1), CT=COS(THETA1).  Derivation given in appendix C.

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION NLAM, MX
      DOUBLE COMPLEX ROOT, G, V1, V2
      INTEGER BETA

      ABAR = NLAM**2 + BETA**2 + 2.0D0*NLAM*BETA*ST       ! eq. C-18
      E    = BETA**2 - ABAR*(MX/CT)**2                    ! eq. C-19

c.. E < 0 for propagation, E > 0 for decay. Any E = 0 (resonance) cases
c.. will be caught by the prior call to subroutine ALFBET.

      IF (E .LT. 0.0D0) THEN
         ROOT = CMPLX(SQRT(-E), 0.0D0)
      ELSE
         ROOT = CMPLX(0.0D0, -SQRT(E))
      ENDIF

      F    =  BETA + NLAM*ST
      G    =  NLAM*BETA*CT/ROOT

      V1 = ( -F + G)/(2.0D0*ABAR)               ! eq. C-15
      V2 = (  F + G)/(2.0D0*ABAR)               ! eq. C-16
      V3 = - NLAM*CT/ABAR                       ! eq. C-17

      RETURN
      END
C ------------------------------------------------------------------------
C
```

Figure 4. (continued) Source code for CUP2D.

```
      SUBROUTINE SCOEFS(NH,NP,C1,C2,SC2,CT1,CT2,ST1,ST2,XS,
     > MXB,LAM2,ALF,BETA,R12,R21,R13,R31,T14,T28,T38,KMATRX,KSUP,KSDN)
c.. Generates elements of the matrix of influence coefficients KMATRX that give
c.. the upwash caused by stator loading at control points on the rotor and
c.. stator.  These are computed from KSR(N,K,I,J), effect of stator on rotor,
c.. and KSS'(N,I,J)1, effect of stator on stator.  The prime on KSS' indicates
c.. that only the part of KSS associated with waves reflected from the actuator
c.. disk is computed here.  The remainder is computed in GENKSS using original
c.. routines from the Smith code.  N counts the stator loading harmonics, I the
c.. control points, and J the load elements.  K counts the cascade wave index in
c.. the stator frame which becomes the time harmonic index in the rotor frame.

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION MXB,MR1,MR2,LAM2,KMATRX(1020,1020)
      INTEGER BET,BETA(-5:5,-5:5)
      DOUBLE COMPLEX AI,ALF1,ALF2,ALF3,EXPE1,E1I,E2I,E3I,
     > KS1,KS2,KS3,V1,V2,VS1,VS2,VS3,KSR,KSS,
     > ALF(9,-5:5,-5:5),
     > KSUP(5,-5:5,51),KSDN(5,-5:5,51),
     > R12(-5:5,-5:5),R21(-5:5,-5:5),R13(-5:5,-5:5),R31(-5:5,-5:5),
     > T14(-5:5,-5:5),T28(-5:5,-5:5),T38(-5:5,-5:5)
      AI = (0.0D0, 1.0D0)
      PI = 3.14159265D0

      WRITE(*,*)'Entering SCOEFS'
      MR1    = MXB/CT1              ! Relative Mach #, row 1
      MR2    = MXB/CT2              ! Relative Mach #, row 2
      CON    = MR2/SC2              ! A constant
      XE     = XS + C2*CT2          ! Axial coordinate of stator exit
      NPNH2 = NP*NH*2
      DO 10 N =   1, NH
      DO 10 K = -NH, NH
      BET  = BETA(N,K)
      ALF1 = ALF(1,N,K)
      ALF2 = ALF(2,N,K)
      ALF3 = ALF(3,N,K)

      CALL GETVS(BET,N*LAM2,MXB,ST2,CT2,V1,V2,V3)

      DO 10 J = 1, NP                              ! Loop on load elements
      Z0J = 0.5D0*(1D0 - COS(PI*(J-1D0)/NP))       ! Chordwise locations for loads

      KS1 =  CON*V1*EXP(-AI*(ALF1*CT2+BET*ST2)*C2*Z0J)    !
      KS2 =  CON*V2*EXP(-AI*(ALF2*CT2+BET*ST2)*C2*Z0J)    ! eq. 22
      KS3 =  CON*V3*EXP(-AI*(ALF3*CT2+BET*ST2)*C2*Z0J)    !

c.. Compute VS1, VS2, VS3 from eq. 26
c.. Hold out EXPE1=exp(-AI*ALF1*XE) from VS1 here to avoid overflow later in KSR
      EXPE1 = EXP(-AI*ALF1*XE)

      VS1 = (   ( R12(N,K)*R21(N,K)*EXP(AI*ALF2*XE)
     >          + R13(N,K)*R31(N,K)*EXP(AI*ALF3*XE))*KS1*EXP(-AI*ALF1*XS)
     >          + R21(N,K)*KS2*EXP(AI*ALF2*(XE-XS))
     >          + R31(N,K)*KS3*EXP(AI*ALF3*(XE-XS))          )
     >      /(1.0D0 - R12(N,K)*R21(N,K)*EXP(AI*(ALF2-ALF1)*XE)
     >          - R13(N,K)*R31(N,K)*EXP(AI*(ALF3-ALF1)*XE) )

      VS2 = (KS1*EXP(-AI*ALF1*XS)+VS1*EXPE1)*R12(N,K)
      VS3 = (KS1*EXP(-AI*ALF1*XS)+VS1*EXPE1)*R13(N,K)

c.. KSUP and KSDN are passed out of the subroutine for later use in computing
c.. pressure in the non-swirl regions upstream and downstream.
      KSUP(N,K,J) = (KS1*EXP(-AI*ALF1*XS)+VS1*EXPE1) * T14(N,K)          ! eq. 77
      KSDN(N,K,J) = (KS2*EXP(-AI*ALF2*XS)+VS2)*EXP(AI*ALF2*XE)*T28(N,K) ! eq. 88
     >          +(KS3*EXP(-AI*ALF3*XS)+VS3)*EXP(AI*ALF3*XE)*T38(N,K)
```

Figure 4. (continued) Source code for CUP2D.

```
c.. Loop on rotor control points
      DO 10 I = 1, NP
      ZI  = 0.5D0*(1D0-COS(PI*(2*I-1D0)/(2*NP)))
      E1I = (ALF1*CT1+BET*ST1)*C1*ZI
      E2I = (ALF2*CT1+BET*ST1)*C1*ZI
      E3I = (ALF3*CT1+BET*ST1)*C1*ZI

c.. Effect of stator on rotor, eq. 37
      IF (K .NE. 0) THEN
      KSR =  1/MR1* (
     >     (BET *CT1  - ALF1*ST1)*KS1*EXP(AI*E1I)*EXP(-AI*ALF1*XS)
     >   + (BET *CT1  - ALF1*ST1)*VS1*EXP(AI*E1I)*EXPE1
     >   + (BET *CT1  - ALF2*ST1)*VS2*EXP(AI*E2I)
     >   + (ALF3*CT1  +  BET*ST1)*VS3*EXP(AI*E3I)   )

c..  Place elements in KMATRX, forming real elements, from complex KSR, eq. 70
      KMATRX((2*ABS(K)-2)*NP+I,NPNH2+(2*N-2)*NP+J) =
     > KMATRX((2*ABS(K)-2)*NP+I,NPNH2+(2*N-2)*NP+J)+REAL(KSR)
      KMATRX((2*ABS(K)-2)*NP+I,NPNH2+(2*N-1)*NP+J) =
     > KMATRX((2*ABS(K)-2)*NP+I,NPNH2+(2*N-1)*NP+J)-IMAG(KSR)
      KMATRX((2*ABS(K)-1)*NP+I,NPNH2+(2*N-2)*NP+J) =
     >KMATRX((2*ABS(K)-1)*NP+I,NPNH2+(2*N-2)*NP+J)+ISIGN(1,K)*IMAG(KSR)
      KMATRX((2*ABS(K)-1)*NP+I,NPNH2+(2*N-1)*NP+J) =
     > KMATRX((2*ABS(K)-1)*NP+I,NPNH2+(2*N-1)*NP+J)+ISIGN(1,K)*REAL(KSR)

      ENDIF

c.. Compute the portion of the KSS coefs caused by the reflected waves, eq. 31.
      KSS =  1/MR2*
     >(  (BET*CT2 - ALF1*ST2)*VS1*
     >          EXP(AI*(ALF1*(XS-XE)+(ALF1*CT2+BET*ST2)*C2*ZI))
     >  +(BET*CT2 - ALF2*ST2)*VS2*
     >          EXP(AI*(ALF2*XS+      (ALF2*CT2+BET*ST2)*C2*ZI))
     >  +(ALF3*CT2 + BET*ST2)*VS3*
     >          EXP(AI*(ALF3*XS+      (ALF3*CT2+BET*ST2)*C2*ZI))       )

c.. Form real elements, do sum over K, and place in KMATRX
      KMATRX(NPNH2+(2*N-2)*NP+I,NPNH2+(2*N-2)*NP+J) =
     >   KMATRX(NPNH2+(2*N-2)*NP+I,NPNH2+(2*N-2)*NP+J) + REAL(KSS)    ! eq. 64
      KMATRX(NPNH2+(2*N-1)*NP+I,NPNH2+(2*N-2)*NP+J) =
     >   KMATRX(NPNH2+(2*N-1)*NP+I,NPNH2+(2*N-2)*NP+J) + IMAG(KSS)    ! eq. 65

10    CONTINUE

c.. Fill in the remaining sections of the stator on stator quarter of the matrix
c.. from the second parts of Eqs. 64 and 65.
      DO 20 N = 1, NH
      DO 20 J = 1, NP
      DO 20 I = 1, NP
          KMATRX(NPNH2+(2*N-1)*NP+I,NPNH2+(2*N-1)*NP+J) =
     >    KMATRX(NPNH2+(2*N-2)*NP+I,NPNH2+(2*N-2)*NP+J)
          KMATRX(NPNH2+(2*N-2)*NP+I,NPNH2+(2*N-1)*NP+J) =
     >   -KMATRX(NPNH2+(2*N-1)*NP+I,NPNH2+(2*N-2)*NP+J)
20    CONTINUE

      RETURN
      END
C ----------------------------------------------------------------------
C
```

Figure 4. (continued) Source code for CUP2D.

```
      SUBROUTINE GENKRR(NH,NP,C1,C2,SC1,SC2,CT1,ST1,MXB,LAM1,   KMATRX)
c.. This generates the input needed to call Smith's matrix generation routines
c.. for the effect of the rotor on itself via the direct waves.  It fills the
c.. WHEAD common block and calls Smith's routine DSWK.  This returns the real
c.. and imaginary parts of the matrix. These are placed in the appropriate
c,, locations in KMATRX, adding to the elements already computed by RCOEFS that
c.. account for the effect of the rotor on itself via the reflected waves.

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /WHEAD/ SC,STAG,MACH,LAM,PHASE,DEG,PI,COSST,SINST,
     >               MACH2,B,BETA2,BC,BC2
      DOUBLE PRECISION MXB, LAM1, KR(51,51),KI(51,51),KMATRX(1020,1020)
      DOUBLE PRECISION MACH,LAM,MACH2

      WRITE(*,*)'Entering GENKRR'

c.. Fill /WHEAD/ common block (except for LAM and PHASE)
      SC     = SC1
      STAG   = ASIN(ST1)
      MACH   = MXB/CT1
      MACH2  = MACH**2
      BETA2  = 1D0 - MACH2
      B      = SQRT(BETA2)
      DEG    = 57.29578D0
      PI     = 3.141592650D0
      COSST  = CT1
      SINST  = ST1
      BC2    = 1D0 - MXB**2
      BC     = SQRT(BC2)

c.. Loop on rotor loading harmonic
      DO 300 N=1, NH
      LAM = N*LAM1*C1
      PHASE = 2.0D0*PI*N*C1/C2*SC1/SC2

c.. Call Smith's original matrix generation routine
      CALL DSWK(KR,KI,NP,IW)
      IF (IW .EQ. 1) THEN                      ! Smith's resonance check
      WRITE(*,1) N
1     FORMAT(1X,'DSWK RETURNED IW=1 TO GENKRR FOR N=',I2)
      STOP 'EXECUTION TERMINATED DUE TO RESONANCE'
      ENDIF

c.. Add Smith's real and imaginary matrix elements into KMATRX, see Eqs. 62 & 63
          DO 100 I = 1, NP
          DO 100 J = 1, NP
             KMATRX((2*N-2)*NP+I,(2*N-2)*NP+J) =
     >          KMATRX((2*N-2)*NP+I,(2*N-2)*NP+J)  + KR(I,J)
             KMATRX((2*N-1)*NP+I,(2*N-1)*NP+J) =
     >          KMATRX((2*N-1)*NP+I,(2*N-1)*NP+J)  + KR(I,J)
             KMATRX((2*N-1)*NP+I,(2*N-2)*NP+J) =
     >          KMATRX((2*N-1)*NP+I,(2*N-2)*NP+J)  + KI(I,J)
             KMATRX((2*N-2)*NP+I,(2*N-1)*NP+J) =
     >          KMATRX((2*N-2)*NP+I,(2*N-1)*NP+J)  - KI(I,J)
100       CONTINUE
300       CONTINUE

      RETURN
      END
C ----------------------------------------------------------------------------
C
```

Figure 4. (continued) Source code for CUP2D.

```
      SUBROUTINE GENKSS(NH,NP,C1,C2,SC1,SC2,CT2,ST2,MXB,LAM2,    KMATRX)
c.. This generates the input needed to call Smith's matrix generation routines
c.. for the effect of the stator on itself via the direct waves.  It fills the
c.. WHEAD common block and calls Smith's routine DSWK.  This returns the real
c.. and imaginary parts of the matrix. These are placed in the appropriate
c.. locations in KMATRX, adding to the elements already computed by SCOEFFS that
c.. account for the effect of the stator on itself via the reflected waves.

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /WHEAD/ SC,STAG,MACH,LAM,PHASE,DEG,PI,COSST,SINST,
     >               MACH2,B,BETA2,BC,BC2
      DOUBLE PRECISION MXB, LAM2, KR(51,51),KI(51,51),KMATRX(1020,1020)
      DOUBLE PRECISION MACH,LAM,MACH2

      WRITE(*,*)'Entering GENKSS'

C... Fill /WHEAD/ common block (except for LAM and PHASE)
      SC    = SC2
      STAG  = ASIN(ST2)
      MACH  = MXB/CT2
      MACH2 = MACH**2
      BETA2 = 1.0D0 - MACH2
      B     = SQRT(BETA2)
      DEG   = 57.29578D0
      PI    = 3.14159265D0
      COSST = CT2
      SINST = ST2
      BC2   = 1.0D0 - MXB**2
      BC    = SQRT(BC2)

C... Loop on stator loading harmonic
      NPNH2 = NP*NH*2
      DO 300 N=1, NH
      LAM   =  N*LAM2*C2
      PHASE = -2.0D0*PI*N*C2/C1*SC2/SC1

C... Call Smith's original matrix generation routine
      CALL DSWK(KR,KI,NP,IW)
      IF (IW .EQ. 1) THEN                    ! Smith's resonance check
      WRITE(*,1) N
1     FORMAT(1X,'DSWK RETURNED IW=1 TO GENKSS FOR N=',I2)
      STOP 'EXECUTION TERMINATED DUE TO RESONANCE'
      ENDIF

C.. Add Smith's real and imaginary matrix elements into KMATRX, see Eqs. 64 & 65
          DO 100 I = 1, NP
          DO 100 J = 1, NP
      KMATRX(NPNH2+(2*N-2)*NP+I,NPNH2+(2*N-2)*NP+J) =
     >   KMATRX(NPNH2+(2*N-2)*NP+I,NPNH2+(2*N-2)*NP+J)  + KR(I,J)
      KMATRX(NPNH2+(2*N-1)*NP+I,NPNH2+(2*N-1)*NP+J) =
     >   KMATRX(NPNH2+(2*N-1)*NP+I,NPNH2+(2*N-1)*NP+J)  + KR(I,J)
      KMATRX(NPNH2+(2*N-1)*NP+I,NPNH2+(2*N-2)*NP+J) =
     >   KMATRX(NPNH2+(2*N-1)*NP+I,NPNH2+(2*N-2)*NP+J)  + KI(I,J)
      KMATRX(NPNH2+(2*N-2)*NP+I,NPNH2+(2*N-1)*NP+J) =
     >   KMATRX(NPNH2+(2*N-2)*NP+I,NPNH2+(2*N-1)*NP+J)  - KI(I,J)
100   CONTINUE
300   CONTINUE

      RETURN
      END
C ____________________________________________________________________
C
```

Figure 4. (continued) Source code for CUP2D.

```
      SUBROUTINE SOLVE(NH,NP,KMATRX,WR,WS,LR,LS)

c.. This routine finds the loading on both the rotor and the stator
c.. simultaneously by matrix inversion. Externally imposed upwash velocities
c.. enter the routine in complex form (WR,WS) and are used to form a one
c.. dimensional real vector WASH.  KMATRX is inverted using the LINPACK routines
c.. and multiplied by WASH.  The result is the one dimensional load vector LOAD.
c.. This is decomposed into the complex load vectors LR and LS and these are
c.. sent to the LOADS routine for output.

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION  KMATRX(1020,1020), LOAD(1020), WASH(1020)
      DOUBLE COMPLEX  WR(5,51), WS(5,51), LR(5,51), LS(5,51)

      ZERO  = 0.0D0
      NPNH2 = NP*NH*2               ! constant needed for element shifting
      NPNH4 = NP*NH*4               ! size of one side of real matrix

c.. Invert matrix of influence coefficients
      CALL MATINV(KMATRX, NPNH4)
c.. KMATRX is now KMATRX inverse

c.. Form real upwash vector from complex upwash vectors for rotor and stator.
c.. Minus signs because objective is to find the loading that produces upwash
c.. to cancel wake upwash.  Note, rotor upwash is zero for the usual rotor/
c.. stator interaction problem, but the code could deal with rotor vibration or
c.. stator potential field excitation also.
      DO 10 K = 1, NH
      DO 10 I = 1, NP
         WASH(          (2*K-2)*NP+I) = - REAL (WR(K,I))    ! eq. 71
         WASH(          (2*K-1)*NP+I) = - IMAG (WR(K,I))    !
         WASH( NPNH2 +  (2*K-2)*NP+I) = - REAL (WS(K,I))    !
         WASH( NPNH2 +  (2*K-1)*NP+I) = - IMAG (WS(K,I))    !
10    CONTINUE

c.. Zero load vector before matrix multiplication
      DO 20 NU = 1, NPNH4
        LOAD(NU) = ZERO
20    CONTINUE

c.. Matrix multiplication to get load vector, which is in real form.
      DO 30 NU = 1, NPNH4
      DO 30 MU = 1, NPNH4
         LOAD(NU) = LOAD(NU) + KMATRX(NU,MU) * WASH(MU)
30    CONTINUE

c.. Form complex load vectors for rotor and stator from single real load vector
      DO 40 N = 1, NH
      DO 40 J = 1, NP
        LR(N,J)=CMPLX(LOAD(       (2*N-2)*NP+J),LOAD(       (2*N-1)*NP+J)) ! eq.
        LS(N,J)=CMPLX(LOAD(NPNH2+(2*N-2)*NP+J),LOAD(NPNH2+(2*N-1)*NP+J)) ! 72
40    CONTINUE

      CALL LOADS(NH,NP,LR,LS)

      RETURN
      END
C ____________________________________________________________________________
C
```

Figure 4. (continued) Source code for CUP2D.

```
      SUBROUTINE LOADS(NH,NP,LR,LS)
c.. Computes and prints integrated loading on both blade rows for multiple
c.. harmonics.  Formulas equivalent to Smith's eq. 56.  Values are lift per
c.. unit span divided by chord*rho*relv**2.  Treatment of first and last terms
c.. handled automatically by inversion process.

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE COMPLEX LR(5,51), LS(5,51), CLR, CLS

      WRITE(*,*) ' '
      WRITE(*,*)'          ***  VALUES OF LIFT COEFFICIENTS ***'
      WRITE(*,*) ' N          CLROTOR(N)                  CLSTATOR(N)'
      WRITE(*,*) '          real        imag            real        imag '

      DO 20 N=1, NH
           CLR = (0D0, 0D0)
           CLS = (0D0, 0D0)
             DO 10 J=1, NP
                CLR = CLR - LR(N,J)
                CLS = CLS - LS(N,J)

cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c.. Turn on the next 2 lines for printout of chordwise load distributions
C       WRITE(*,5) N, J, LR(N,J), LS(N,J)
C5      FORMAT(1X,I2,I5,2G15.5,'    ',2G15.5)
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

10            CONTINUE
           WRITE(*,1) N, CLR, CLS
1         FORMAT(1X,I2,'   ', 2F10.5,'       ',2F10.5 )

20    CONTINUE
      WRITE(*,*) ' '

      RETURN
      END
C ---------------------------------------------------------------------------
C
```

Figure 4. (continued) Source code for CUP2D.

```
      SUBROUTINE OUTPUT(NH,NP,B1,B2,MXA,MXB,MXC,MS,MY1,MY2,ALF,ABA,ABC,
     >     XS,XA,XC,POPSA,POPSC,KRUP,KRDN,KSUP,KSDN,LR,LS )
c.. Calculates sound pressure at axial locations XA and XB and sound power
c.. (average per unit area) upstream and downstream based on loading from
c.. SOLVE and influence functions from INFFNS.

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION MXA, MXB, MXC, MS, MY1, MY2, PWLAT(5), PWLCT(5)
      INTEGER B1, B2, ORDER
      DOUBLE COMPLEX  A4,A8,G4,G8, PUP, PDN, ALF(9,-5:5,-5:5), AI,
     >   LR   (5,51),      LS   (5,51),      LUP (5,-5:5),    LDN (5,-5:5),
     >   LRUP(5,-5:5),    LRDN(5,-5:5),    LSUP(5,-5:5),    LSDN(5,-5:5),
     >   KRUP(5,-5:5,51),KRDN(5,-5:5,51),KSUP(5,-5:5,51),KSDN(5,-5:5,51)

      AI     = (0.0D0,1.0D0)

      WRITE(*,*)  ' '
      WRITE(*,*)'Axial locations for sound pressure output in radii from
     > rotor leading edge '
      WRITE(*,5) XA, XC
5     FORMAT(1X, 'For PRESup, Xa = ',F5.3,',   For PRESdn, Xc = ',F5.3)

      WRITE(*,*)' '
      WRITE(*,*)'Decibel Levels for Pressure Waves and Power Levels
     >     |    Cutoff Ratios'
      WRITE(*,*)'     N     K  FREQ  nB1-kB2 PRESup  PRESdn    PWLup    PWL
     >dn    |    A      B      C'

c.. Zero the wave accumulators
      DO 10 N =    1, NH
      DO 10 K = -NH, NH
         LRUP(N,K)  = (0D0, 0D0)
         LRDN(N,K)  = (0D0, 0D0)
         LSUP(N,K)  = (0D0, 0D0)
         LSDN(N,K)  = (0D0, 0D0)
10    CONTINUE

c.. Sum contributions to waves over load elements
      DO 20 N =    1, NH
      DO 20 K = -NH, NH
      DO 20 J =    1, NP
          LRUP(N,K)  = LRUP(N,K)  + KRUP(N,K,J)*LR(N,J)    ! eq. 84
          LRDN(N,K)  = LRDN(N,K)  + KRDN(N,K,J)*LR(N,J)    ! eq. 96
          LSUP(N,K)  = LSUP(N,K)  + KSUP(N,K,J)*LS(N,J)    ! eq. 77
          LSDN(N,K)  = LSDN(N,K)  + KSDN(N,K,J)*LS(N,J)    ! eq. 89
20    CONTINUE

c.. Add rotor waves to stator waves upstream and downstream
      DO 30 N = 1, NH
      DO 32 K = 1, NH
          LUP(N, K)  =         LRUP(K, N)   + LSUP(N, K)    ! eq. 106
          LUP(N,-K)  = -CONJG(LRUP(K,-N))  + LSUP(N,-K)
          LDN(N, K)  =         LRDN(K, N)   + LSDN(N, K)    ! eq. 116
          LDN(N,-K)  = -CONJG(LRDN(K,-N))  + LSDN(N,-K)
32    CONTINUE

c.. Following terms computed without steady loading effect from rotor
          LUP(N,0)  = LSUP(N,0)
          LDN(N,0)  = LSDN(N,0)
30    CONTINUE

c.. Compute modal and total powers upstream and downstream
      DO 50 N =    1, NH
           PWRAT = 0.0D0              ! Total power accumulator, upstream
           PWRCT = 0.0D0              ! Total power accumulator, downstream
```

Figure 4. (continued) Source code for CUP2D.

```
      DO 52 K = -NH, NH
           FREQ    = N*B1*MY1 - K*B2*MY2
           ORDER   =     N*B1 - K*B2        ! Negative of beta(n,k)

c.. CTRAT's below are cutoff ratios (infinity is rigged to print 9.99), eq. B-6
           IF (ORDER .EQ. 0) THEN
              CTRATA = 9.99D0
              CTRATB = 9.99D0
              CTRATC = 9.99D0
           ELSE
            CTRATA = SQRT((ABA*FREQ)**2/((1D0-MXA**2)*ORDER**2))
            CTRATB = SQRT((FREQ-ORDER*MS)**2/((1D0-MXB**2)*ORDER**2))
            CTRATC = SQRT((ABC*FREQ)**2/((1D0-MXC**2)*ORDER**2))
            CTRATA = MIN(ABS(CTRATA),9.99D0)
            CTRATB = MIN(ABS(CTRATB),9.99D0)
            CTRATC = MIN(ABS(CTRATC),9.99D0)
           ENDIF

c.. Output only waves with cutoff ratios > 0.2 in up or downstream region
           IF (CTRATA .GT. 0.2D0) GOTO 51
           IF (CTRATC .LE. 0.2D0) GOTO 52

51         A4 = ALF(4,N,K)                      ! axial wavenumber upstream
           A8 = ALF(8,N,K)                      ! axial wavenumber downstream
           G4 = - (ABA*FREQ + MXA*A4)           ! eq. 81
           G8 = - (ABC*FREQ + MXC*A8)           ! eq. 93

           CALL GETPWL(CTRATA,MXA,A4,G4,LUP(N,K),-1, PWRA, PWLA)
           CALL GETPWL(CTRATC,MXC,A8,G8,LDN(N,K), 1, PWRC, PWLC)
           PWRAT = PWRAT + PWRA
           PWRCT = PWRCT + PWRC

c.. Compute pressures at phi=0 and x = xa and xc
           PUP =  G4*LUP(N,K)*EXP(AI*A4*XA)
           PDN =  G8*LDN(N,K)*EXP(AI*A8*(XC-XE))

      WRITE(*,1) N, K, FREQ, ORDER, DBEL(PUP,POPSA), DBEL(PDN,POPSC),
     >               PWLA, PWLC, CTRATA, CTRATB, CTRATC
1     FORMAT(1X, 2I5, F7.2, I6, 4F8.1, '     ', 3F6.2)

52    CONTINUE
      PWLAT(N) = MAX(0D0, 10D0*LOG10(PWRAT*1.0D13+1.0D-30))
      PWLCT(N) = MAX(0D0, 10D0*LOG10(PWRCT*1.0D13+1.0D-30))

      WRITE(*,3)N, PWLAT(N), PWLCT(N)
3     FORMAT(1X,'--------------     Total power for N=',I2,' ',2F8.1)
      WRITE(*,*)' '

50    CONTINUE

      RETURN
      END
C ______________________________________________________________________
C
```

Figure 4. (continued) Source code for CUP2D.

```
      SUBROUTINE GETPWL(CTRAT,MX,A,G,L,IX, POWER,PWL)
c.. Computes sound power level according to theory in Eqs. 98-116.
c.. Checks that power is real (within numerical accuracy) and that real part
c.. has the correct sign for flux away from blades.

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION MX
      DOUBLE COMPLEX A, G, L, PWR

c.. No power for waves that are cut off.
      IF (CTRAT .LE. 1D0) THEN
          POWER = 0D0
          PWL   = 0D0
          RETURN
      ENDIF

c.. Treat power as complex to verify correct behavior.  See Eqs. 112 & 115.
c.. SIGN function below is needed because derivation was for power flux in the
c.. +x direction.  The expression below must be real and > 0 for regions A & C.
      PWR = ((1D0+MX**2)*A*G + MX*(A**2+G**2))*ABS(L)**2 *SIGN(1,IX)
c.. Check that PWR is real.
      TEST = ABS(  IMAG(PWR)/(REAL(PWR)+1.D-20)   )
      IF (TEST .GT. .001D0) THEN
      STOP 'Execution terminated because power flux is complex'
      ENDIF

c.. Check that power flux is outgoing on either side of the source.
      IF (REAL(PWR) .LT. 0.) THEN
      WRITE(*,*)'IX = ', IX
      STOP 'Execution terminated because power flux is negative'
      ENDIF

      POWER = REAL(PWR)

      IF (POWER .LT. 1.0D-13) THEN
      PWL = 0D0
      ELSE
      PWL = 10.0D0*LOG10(POWER*1.0D13)        ! eq. 113
      ENDIF

      RETURN
      END
C______________________________________________________________________

      DOUBLE PRECISION FUNCTION DBEL(X,POPS)
c.. Computes SPL dB from harmonic peak value normalized by ambient rho*c**2.
c.. Uses POPS, the ratio of local ambient pressure to 2116 psf.
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE COMPLEX X

      TEMP=20.0D0*LOG10(.70711D0*ABS(X)*1.4D0*POPS*5.0651D9+1.D-35)
      DBEL=MAX(TEMP,0.0D0)

      END
C______________________________________________________________________
C
```

Figure 4. (continued) Source code for CUP2D.

```
C
C  Calculation of Kernel Matrix Elements from Original Smith Code
C
      SUBROUTINE DSWK(KR,KI,NP,IW)
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION MACH,LAM,MACH2,MACH4,MACH6,KR(51,51),KI(51,51)
      COMMON /WHEAD/ SC,STAG,MACH,LAM,PHASE,DEG,PI,COSST,SINST,
     >   MACH2,B,B2,BC,BC2
      COMMON/WAVEC/ XU,APU,APD,ANU,AND,PUR,PDR,PUI,VU,VD
      DIMENSION ICHECK(51,51),ZE(51),ZP(51)
C
C.... CONSTANTS FOR VORTEX SHEET CALCULATION
      X     = LAM*SC*COSST
      Y     = LAM*SC*SINST + PHASE
      VORT = 0.5*LAM*SINH(X)/(COSH(X) - COS(Y))
C
C.... CONSTANTS FOR LOG SINGULARITY CORRECTION
      MACH4 = MACH2*MACH2
      MACH6 = MACH2*MACH4
      B4    = B2*B2
      B6    = B2*B4
      A1    = 1.0 - 0.5*MACH2/B2
      A2    = 1.0 - 0.5/B2 + 0.9*MACH2/B4
      A3    = 0.5*(1.0 - 1.0/B2 + MACH2/(6.0*B4) + 1.0/(3.0*B4)
     >         - 0.375*MACH4/B6 + MACH6/(6.0*B6))
C
C.... MATCHING AND VORTEX POINTS
      DO I = 1,NP
         EPSIL = PI*FLOAT(2*I - 1)/FLOAT(2*NP)
         ZE(I) = 0.5*(1.0 - COS(EPSIL))
         PSI = PI*FLOAT(I - 1)/FLOAT(NP)
         ZP(I) = 0.5*(1.0 - COS(PSI))
      ENDDO
C
C.... ZERO COUNTS AND ARRAYS
      IR = 0
      ICOUNT = 0
      IW = 0
      NP2 = NP*NP
      DO I = 1,NP
         DO J = 1,NP
            ICHECK(I,J) = 0
            KR(I,J) = 0.0
            KI(I,J) = 0.0
         ENDDO
      ENDDO
C
C     ASSEMBLE MATRIX
C     I( = M + 1 IN PAPER) GIVES VORTEX POSITION
C.... J( = L + 1 IN PAPER) GIVES MATCHING POINT
C
30    CALL WAVE(IR,IW)
      IF(IW.EQ.1) RETURN
      DO I = 1,NP
         DO J = 1,NP
            IF(ICHECK(I,J).EQ.1) GO TO 131
            POS = ZE(I) - ZP(J)
            IF(POS.GT.0.0) GO TO 90
C
C.......... UPSTREAM POINT
            XP = EXP(XU*POS)
            YP = APU*POS
            QR = XP*COS(YP)
            QI = XP*SIN(YP)
            TERMR = (PUR*QR - PUI*QI)/SC
```

Figure 4. (continued) Source code for CUP2D.

```
            TERMI = (PUR*QI + PUI*QR)/SC
            GO TO 100
C
C.......... DOWNSTREAM POINT
90          XP = EXP( - XU*POS)
            YP = APD*POS
            QR = XP*COS(YP)
            QI = XP*SIN(YP)
            TERMR = (PDR*QR - PUI*QI)/SC
            TERMI = (PDR*QI + PUI*QR)/SC
C
C.......... ADD TO MATRIX
100         KR(I,J) = KR(I,J) + TERMR
            KI(I,J) = KI(I,J) + TERMI
C
C.......... CHECK CONVERGENCE OF SERIES
C.. The next 3 lines modified from Smith's code on 8/19/91 by DBH

             X = ABS(TERMR) + ABS(TERMI)
             Y = ABS(KR(I,J)) + ABS(KI(I,J))
             IF((X/Y).GT.1.0D - 7) GO TO 131
C            X = TERMR*TERMR + TERMI*TERMI
C            Y = KR(I,J)*KR(I,J) + KI(I,J)*KI(I,J)
C            IF((X/Y).GT.1.0D - 11) GO TO 131


             ICHECK(I,J) = 1
            ICOUNT = ICOUNT + 1
C
C.......... CORRECT FOR LOG SINGULARITY (LAST TIME THROUGH)
            SUM = 0.0
            EPSIL = PI*FLOAT(2*I - 1)/FLOAT(2*NP)
            PSI = PI*FLOAT(J - 1)/FLOAT(NP)
            NPM1 = NP - 1
            DO JR = 1,NPM1
               FJR = FLOAT(JR)
               SUM = SUM + COS(FJR*EPSIL)*COS(FJR*PSI)/FJR
            ENDDO
             SUM = 2.0*SUM + LOG(4.0*ABS(POS))
            SUM = SUM*LAM/(2.0*PI*B)
            PLAM = LAM*POS
            PLAM2 = PLAM*PLAM
            PLAM3 = PLAM2*PLAM
            KR(I,J) = KR(I,J) + SUM*(A1*PLAM - A3*PLAM3)
            KI(I,J) = KI(I,J) + SUM*(1.0 - A2*PLAM2)
C
C.......... ADD VORTICITY WAVE
            IF(POS.LE.0.0) GO TO 131
            KR(I,J) = KR(I,J) + VORT*COS(PLAM)
            KI(I,J) = KI(I,J) - VORT*SIN(PLAM)
131         CONTINUE
         ENDDO
      ENDDO
C
C.... CHECK FOR COMPLETION
      IF(ICOUNT.EQ.NP2) RETURN
      IF(IR.GT.0) THEN
         IR = -IR
      ELSE
         IR = -IR + 1
      ENDIF
      GO TO 30
      END
```

Figure 4. (continued) Source code for CUP2D.

```
C
C  CALCULATION OF ACOUSTIC WAVE PROPERTIES
C
      SUBROUTINE WAVE(IR,IW)
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION MACH,LAM,MACH2
      COMMON /WHEAD/ SC,STAG,MACH,LAM,PHASE,DEG,PI,COSST,SINST,
     >    MACH2,B,B2,BC,BC2
      COMMON/WAVEC/ XU,APU,APD,ANU,AND,PUR,PDR,PUI,VU,VD
C
      BETAH=(PHASE-2.0*PI*FLOAT(IR))/SC
      BETAH2=BETAH*BETAH
      A=LAM*LAM+BETAH2+2.0*LAM*BETAH*SINST
      D=MACH2*(LAM+BETAH*SINST)*COSST/BC2
      E=BETAH2-MACH2*A
      IF(E.NE.0.0) GO TO 32
      WRITE(*,31) IR
31    FORMAT(' Resonance at IR=',I4)
      IW=1
      GO TO 60
32    F=SQRT(ABS(E))
      FB=F/BC2
      H=(BETAH+LAM*SINST)/(2.0*A)
      P=BETAH*LAM*COSST/(F*2.0*A)
      IF(E.GT.0.0) GO TO 50
C
C  WAVE NUMBERS, PROPAGATING CASE
C
      ACUI=D+FB
      ACDI=D-FB
      APU=ACUI*COSST+BETAH*SINST
      APD=ACDI*COSST+BETAH*SINST
      ANU=BETAH*COSST-ACUI*SINST
      AND=BETAH*COSST-ACDI*SINST
      PUR=ANU*(P-H)
      PDR=AND*(P+H)
      PUI=0.0
      XU=0.0
      VU=(P-H)*(LAM+APU)/SC
      VD=(P+H)*(LAM+APD)/SC
      GO TO 60
C
C  WAVE NUMBERS, DECAYING CASE
C
50    APU=D*COSST+BETAH*SINST
      APD=APU
      ANU=BETAH*COSST-D*SINST
      AND=ANU
      PUR=-ANU*H-FB*SINST*P
      PDR=-PUR
      PUI=ANU*P-FB*SINST*H
      XU=FB*COSST
60    RETURN
      END
C
```

Figure 4. (continued) Source code for CUP2D.

```
      SUBROUTINE MATINV(A,N)
C.. This routine written by D.B. Hanson to call the LINPACK routines for
C.. inversion of real matrices.  Call for matrix A.  Inverse returned in
C.. same array.

      DOUBLE PRECISION  A(1020,1020),WORK(1020),DET(2),RCOND,Z(1020)
      INTEGER  IPVT(1020)

      WRITE(*,*) 'Entering MATINV'
      CALL DGECO(A,1020,N,IPVT,RCOND,Z)
      WRITE(*,*) 'Condition number of KMATRX = ', RCOND
      CALL DGEDI(A,1020,N,IPVT,DET,WORK,1)
      RETURN
      END
```

Figure 4. (continued) Source code for CUP2D.

Section 5
Array Dimensions

This section shows how the arrays are dimensioned in case they need to be changed to accommodate more harmonics or panels on the blades. Interpret N_h and N_p below to be the maximum permitted values of number of harmonic and number of panels.

ALF(9, $-N_h:N_h$, $-N_h:N_h$)

BETA($-N_h:N_h$, $-N_h:N_h$)

KRUP, KRDN, KSUP, KSDN(N_h, $-N_h:N_h$, N_P)

R12, R21, R13, R31, T14, T28, T38($-N_h:N_h$, $-N_h:N_h$)

WREXT, WSEXT(N_h, N_p)

LR, LS(N_h, N_p)

KMATRX($4*N_h*N_p$, $4*N_h*N_p$)

FW(N_h)

KR, KI(N_p, N_p)

WR, WS(N_h, N_p)

LOAD, WASH($4*N_h*N_p$)

LUP, LDN(N_h, $-N_h:N_h$)

LRUP, LRDN, LSUP, LSDN(N_h, $-N_h:N_h$)

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	January 1994	Final Contractor Report

4. TITLE AND SUBTITLE

Coupled 2-Dimensional Cascade Theory for Noise and Unsteady Aerodynamics of Blade Row Interaction in Turbofans

Volume 2–Documentation for Computer Code CUP2D

5. FUNDING NUMBERS

WU–535–03–10
C–NAS3–25952

6. AUTHOR(S)

Donald B. Hanson

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

United Technologies Corporation
Pratt and Whitney Division
400 Main Street
East Hartford, Connecticut 06096

8. PERFORMING ORGANIZATION REPORT NUMBER

E–7766

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio 44135–3191

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

NASA CR–4506

11. SUPPLEMENTARY NOTES

Project Manager, Dennis L. Huff, organization code 2770, (216) 433–3913.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Category 07

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***

A 2D linear aeroacoustic theory for rotor/stator interaction with unsteady coupling was derived and explored in Volume 1 of this report. Computer program CUP2D has been written in FORTRAN embodying the theoretical equations. This volume (Volume 2) describes the structure of the code, installation and running, preparation of the input file, and interpretation of the output. A sample case is provided with printouts of the input and output. The source code is included with comments linking it closely to the theoretical equations in Volume 1.

14. SUBJECT TERMS

Acoustics; Unsteady aerodynamics; Turbofan noise; Source noise modeling; Rotor/stator interaction

15. NUMBER OF PAGES

43

16. PRICE CODE

A03

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	