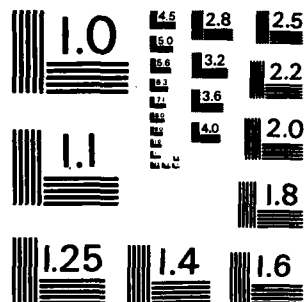


AD A137 064 COMPUTER PROGRAM FOR EVALUATING THE IVES TRANSFORMATION 1/1
IN TURBOMACHINERY... J. CALSPAN ADVANCED TECHNOLOGY
CENTER BUFFALO NY W J RAE JUL 83 CALSPAN 7177 A 1
DECLASSIFIED AFOSR IR HT 1284 149620 R3 C 0096 1/26/92 RI

END
DATE
FILMED
2-84
DTL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AFOSR-TR- 83 - 1284

7

**REVISED COMPUTER PROGRAM FOR EVALUATING THE
IVES TRANSFORMATION IN TURBOMACHINERY CASCADES**

**William J. Rae
Calspan Advanced Technology Center
P.O. Box 400
Buffalo, New York 14225**

July 1983

**Prepared for:
AIR FORCE OFFICE OF SCIENTIFIC RESEARCH
BOLLING AIR FORCE BASE, DC 20332**

**CONTRACT NO. F49620-83-C-0096
AFOSR SCIENTIFIC REPORT**

The views and conclusions contained in this document are those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research or the U.S. Government.

DTIC FILE COPY

**RECEIVED
JAN 20 1984
A**

**Approved for public release;
distribution unlimited.**

84 01 19 109

AD A137064

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFOSK-TR-83-1284	2. GOVT ACCESSION NO. AD A137064	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) REVISED COMPUTER PROGRAM FOR EVALUATING THE IVES TRANSFORMATION IN TURBOMACHINERY CASCADES		5. TYPE OF REPORT & PERIOD COVERED Scientific Report
7. AUTHOR(s) William J. Rae Physical Sciences Department		6. PERFORMING ORG. REPORT NUMBER 7177-A-1
9. PERFORMING ORGANIZATION NAME AND ADDRESS Calspan Advanced Technology Center P. O. Box 400 Buffalo, New York 14225		8. CONTRACT OR GRANT NUMBER(s) F49620-83-C-0096
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Office of Scientific Research / NA Bolling Air Force Base Washington, DC 20332		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61102 F 2307 A
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		12. REPORT DATE July 1983
		13. NUMBER OF PAGES 84
		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Turbomachinery Turbines Compressors Conformal Mapping Grid Generation Transonic Flow Finite-Difference Solutions		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This report contains the description of a computer program for evaluating the Ives transformation, which maps a cascade of turbine or compressor blades con- formally into a rectangle.		

DD FORM 1473

JAN 73

EDITION OF 1 NOV 68 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

FOREWORD

This document supersedes the previous AFOSR Scientific Report cited below. The present report contains a number of improvements in the numerical methods used, and is applicable to a wider range of blade geometries. In addition, a number of errors in Ref. 1 have been corrected, and more complete descriptions are given for certain features of the method.

The author is very indebted to Joseph P. Nenni, John R. Moselle, and Marcia J. Williams for their assistance in the development of this program.



1. Rae, W.J., "A Computer Program for the Ives Transformation in Turbomachinery Cascades", Calspan Report No. 6275-A-3, AFOSR TR-81-0154 (November 1980).

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH (AFOSR)
NOTICE OF TRANSMITTAL TO DTIC
This technical report has been reviewed and is
approved for public release IAW AFR 190-12.
Distribution is unlimited.

ii MATTHEW J. KERPER
Chief, Technical Information Division

ABSTRACT

This report contains the description of a computer program for evaluating the Ives transformation, which maps a cascade of turbine or compressor blades conformally into a rectangle.

TABLE OF CONTENTS

<u>Section</u>	<u>Page</u>
FOREWORD	ii
ABSTRACT	iii
1 INTRODUCTION	1
2 CONFORMAL TRANSFORMATION METHOD	3
3 METRIC EVALUATIONS	33
4 COMPUTER PROGRAM	34
5 CONCLUDING REMARKS	39
APPENDIX A - DETAILS OF THE FAST FOURIER TRANSFORM PROCEDURES	A-1
APPENDIX B - COMPUTER PROGRAM LISTING	B-1
APPENDIX C - DICTIONARY OF VARIABLES	C-1
APPENDIX D - LISTING OF METRIC GENERATOR PROGRAM	D-1
REFERENCES	R-1

Section 1

INTRODUCTION

Some of the most important recent advances in computational fluid dynamics have been made possible by the use of algorithms which solve the equations of motion in boundary-conforming coordinates. Thus the development of methods for carrying out these coordinate transformations has received considerable attention (see, for example, Reference 2). The methods in current use can be divided into three categories: those which apply algebraic shearing or stretching of the coordinates, those which are generated numerically by solving a Poisson equation, and those which are based on a conformal transformation.

This report is concerned with one example from the third category, namely the transformation introduced by Ives and Liutermoza.³ It is capable of transforming a cascade of airfoils into a rectangle, thus facilitating the application of the solution algorithms mentioned above. The content of this report is a review of the transformation itself, together with a practical numerical procedure for applying it to a given cascade. This procedure involves a number of choices, involving branch-point locations, tolerances for various iteration sequences, and formulas for the evaluation of certain special functions.

Section 2 below contains a description of the transformation itself, including modifications that enable the mapping of blades having a rounded trailing edge, and the calculation of grids in which one of the coordinate lines joins the trailing edge to the point at downstream infinity.⁴

2. Thompson, J.F., ed., Numerical Grid Generation, Elsevier Science Publishing Co., New York (1982).
3. Ives, D.C. and Liutermoza, J.F., "Analysis of Transonic Cascade Flow Using Conformal Mapping and Relaxation Techniques", AIAA Journal **15** (1977), pp. 647-652.
4. Rae, W.J., "Modifications of the Ives-Liutermoza Conformal-Mapping Procedure for Turbomachinery Cascades", ASME Paper 83-GT-116 (March 1983).

The final sections contain a description of the computer code and comments on its range of applicability.

Section 2

CONFORMAL TRANSFORMATION METHOD

The method of Ives & Liutermoza consists of a sequence of transformations, which map a two-dimensional cascade into a rectangle. The notation and coordinate system used to define the cascade are shown in Fig. 1. The x -coordinate is measured in the axial direction, and y is perpendicular to x .

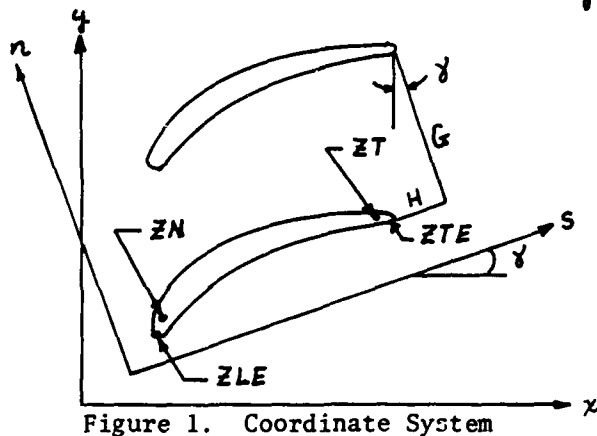


Figure 1. Coordinate System

The quantities s , n and the angle γ denote the "streamwise, normal" coordinates, in terms of which the blade profiles are sometimes defined. These reduce to the x , y set if H is taken as zero.* The origins of both of these coordinate systems are arbitrary.

These coordinates define a complex variable z :

$$z = s + in \quad (2-1)$$

The points ZN and ZT are taken anywhere near the centers of curvature of the leading and trailing edges, while ZLE and ZTE are points which divide the "pressure" side of the blade (i.e., its concave surface) from the "suction" side (its convex surface). These points can be chosen anywhere on the

* Actually, H must not be set identically equal to zero, but to the value $\pm 10^{-9}$, where the $\pm$ signs apply to compressors or turbines, respectively.

leading- and trailing-edge contours; ZTE is the point that will be connected to the "point" at downstream infinity by one of the grid lines, if that option is chosen (i.e., ISHEAR=1).

For the case of a sharp trailing edge (ITE=0), ZT must equal ZTE, and for a sharp leading edge (ILE=0), ZN must equal ZLE. The included angle at a sharp trailing edge, τ , must be specified. This is illustrated in Fig. 2, for the cascade used by Rae and Homicz.⁵

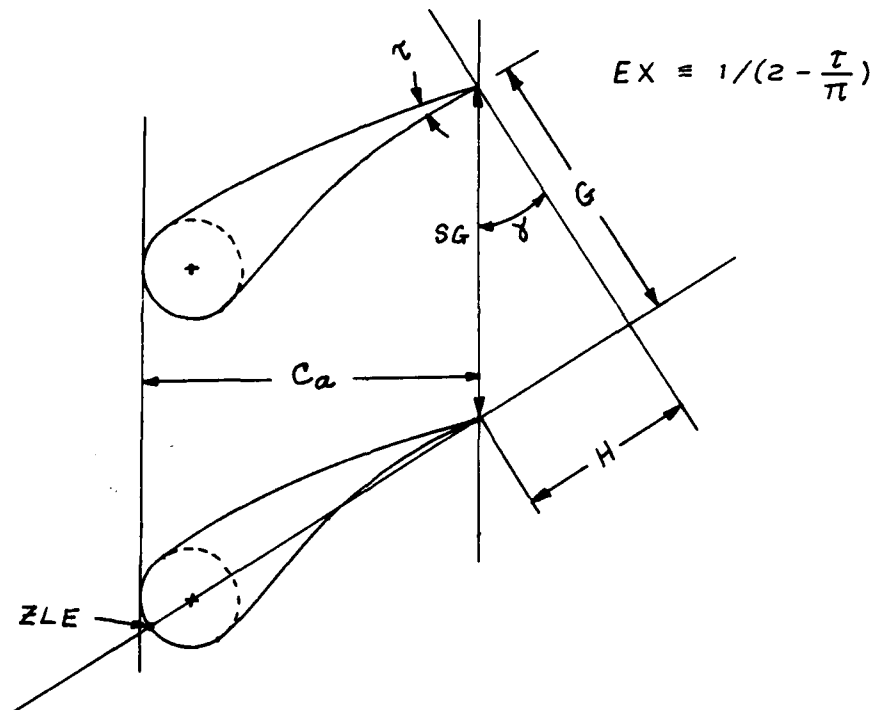


Figure 2. Blade Geometry of Ref. 5

This blade row has a slant-gap/axial chord ratio $SG/C_a = 1$, where SG is the slant gap:

5. Rae, W.J., and Homicz, G.F., "A Rectangular-Coordinate Method for Calculating Nonlinear Transonic Potential Flowfields in Compressor Cascades", AIAA Paper 78-248 (January 1978).

$$SG = \sqrt{H^2 + G^2} \quad (2-2)$$

and a stagger angle γ of 32.91° . This geometry is used, below, to illustrate the steps in the Ives transformation. A comparable set of illustrations, for a cascade of turbine blades with rounded trailing edges, is given in Ref. 4.

Figure 2 shows the relation between the trailing-edge angle τ and an exponent EX (which is used in one of the transformation steps described below). This relation applies only for a sharp trailing edge. For a rounded trailing edge, EX is not related to the 180-degree trailing-edge angle, but is chosen as a number in the range 0.2 to 0.4, as described below.

The blade shape is input as two tables of coordinate pairs, one for the pressure surface, and one for the suction surface. These coordinates, plus the leading- and trailing-edge points ZLE and ZTE are then arranged in an array indexed by KJ, where $KJ=1$ at the trailing edge, and where the numbering proceeds around the pressure side to the leading edge (KJLE), and then along the suction side to the trailing edge, where the point denoted by KJMX is a repeat of that denoted by $KJ=1$. This notation is shown in Fig. 3.

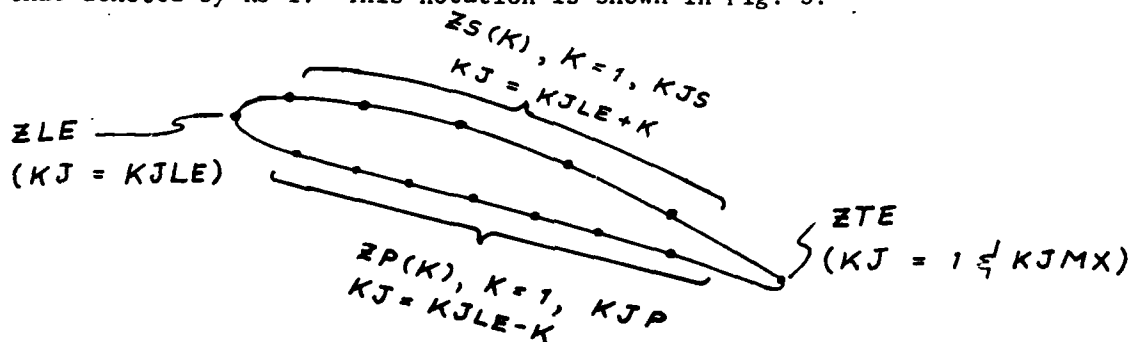


Figure 3. Notation for Blade-Surface Coordinates

The quantities KJS and KJP need not be equal; they are limited to a maximum value of 80 by a dimension statement in the current version of the program.

The first step in the Ives transformation is

$$q(z) = \frac{\sin \left\{ \pi \frac{z - z_T}{H + iG} \right\}}{\sin \left\{ \pi \frac{z - z_N}{H + iG} \right\}} \equiv \frac{\sin \zeta_1}{\sin \zeta_3} \equiv \frac{\zeta_2}{\zeta_4} \quad (2-3)$$

The fact that only differences of z -values are used is what accounts for the arbitrariness of the origin in Fig. 1. On the suction (S) and pressure (P) sides, the function $q(z)$ has the Fortran equivalents:

$$q(z) = \begin{cases} RDS(K) e^{iTHS(K)} \\ RDP(K) e^{iTHP(K)} \end{cases} \quad (2-4)$$

The arguments of the sine functions can be written in a simpler form, as follows:

$$\begin{aligned} \zeta_{1,3} &= \pi \frac{\Delta z (H - iG)}{H^2 + G^2} \\ &= \frac{\pi}{SG} \left\{ \text{Re } \Delta z \cdot \sin \gamma + \text{Im } \Delta z \cdot \cos \gamma \right. \\ &\quad \left. + i \left[\text{Im } \Delta z \cdot \sin \gamma - \text{Re } \Delta z \cdot \cos \gamma \right] \right\} \end{aligned} \quad (2-5)$$

where Δz stands for $z - z_T$ or $z - z_N$ in the expressions for ζ_1 and ζ_3 , respectively. By noting that (see Fig. 4)

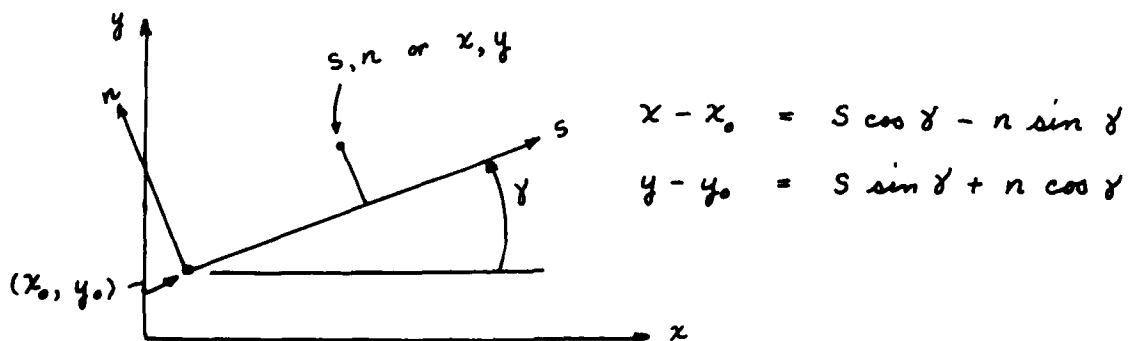


Figure 4. Coordinate Relations

it follows that

$$\begin{aligned}\zeta_1 &= \frac{\pi}{SG} \left\{ (s-s_T) \sin \delta + (n-n_T) \cos \delta + i \left[(n-n_T) \sin \delta - (s-s_T) \cos \delta \right] \right\} \\ &= \frac{\pi}{SG} \left\{ y-y_T - i(x-x_T) \right\}\end{aligned}\quad (2-6)$$

Similarly,

$$\zeta_3 = \frac{\pi}{SG} \left\{ y-y_N - i(x-x_N) \right\} \quad (2-7)$$

The sine transformations:

$$\zeta_2 = \sin \zeta_1, \quad \zeta_4 = \sin \zeta_3$$

map the strip $-\pi/2 \leq \text{Re}[\zeta_{1,3}] \leq +\pi/2$ into the entire ζ_2 (or ζ_4) plane, with cuts along the real axis from $-\infty$ to -1 and from $+1$ to $+\infty$:

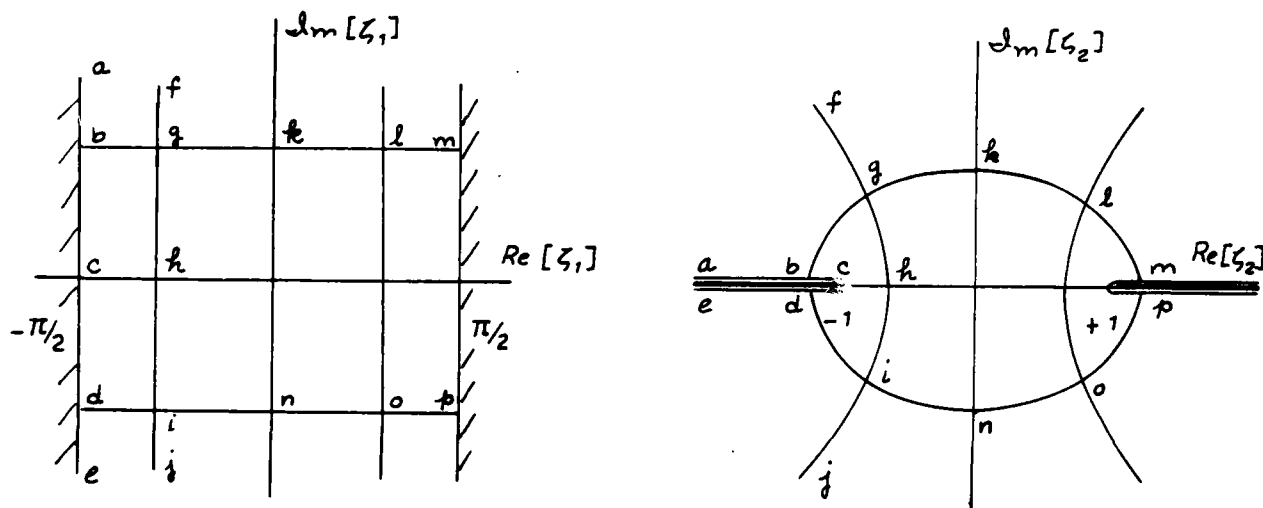


Figure 5. The Mapping $\zeta_2 = \sin \zeta_1$

In order to enforce this single-valuedness, the argument of the sine function should be displaced by the appropriate multiple of $H + iG$, whenever the real part of its argument falls outside the strip noted above, i.e., if

$|Re \zeta| > \frac{\pi}{2}$, then set $\zeta = (\zeta + n\pi) - n\pi$, where n is chosen so as to make $|Re (\zeta + n\pi)| < \frac{\pi}{2}$. It turns out, however, that this displacement of the argument need not be made explicitly, when evaluating the complex sine. This can be seen by noting that

$$\sin(R e^{i\theta}) = \sin(R \cos \theta) \cosh(R \sin \theta) + i \cos(R \cos \theta) \sinh(R \sin \theta)$$

and that

$$\begin{aligned} \sin[(R e^{i\theta} + n\pi) - n\pi] &= \cos n\pi \cdot \sin(R \cos \theta + n\pi + i R \sin \theta) \\ &= \sin(R \cos \theta) \cosh(R \sin \theta) + i \cos(R \cos \theta) \sinh(R \sin \theta) \end{aligned}$$

Thus, no special treatment is required for the argument of the complex sine. The result of this transformation is shown in Fig. 6, where S and P denote the suction and pressure sides. The quantity $q(z)$ is then formed from the ratio of the sines; this curve is shown in Fig. 7. The interior of the blade lies outside the closed curve in this plane. Because this function is to be raised to a power, its argument must be defined on the suction and pressure sides in such a way that the exponentiation will map the trailing-edge region into a straight segment. In the computer program, these arguments are first found from the Fortran DATAN2 function, which returns angles in the range from $-\pi$ to $+\pi$. These arguments are then adjusted, by defining the argument for $KJ=2$, and by then examining the points $KJ=3$ to $KJMX$, adding or subtracting 2π to the argument whenever the cut (along the negative real axis) in the DATAN2 function is traversed counterclockwise or clockwise, respectively. The arguments defined for $KJ=2$ use the following convention: for $H > 0$ (i.e., a stagger corresponding to a compressor blade row) the argument at $KJ=2$ is taken to be negative. For all other cases, the argument at $KJ=1$ is accepted as returned by the DATAN2 function.

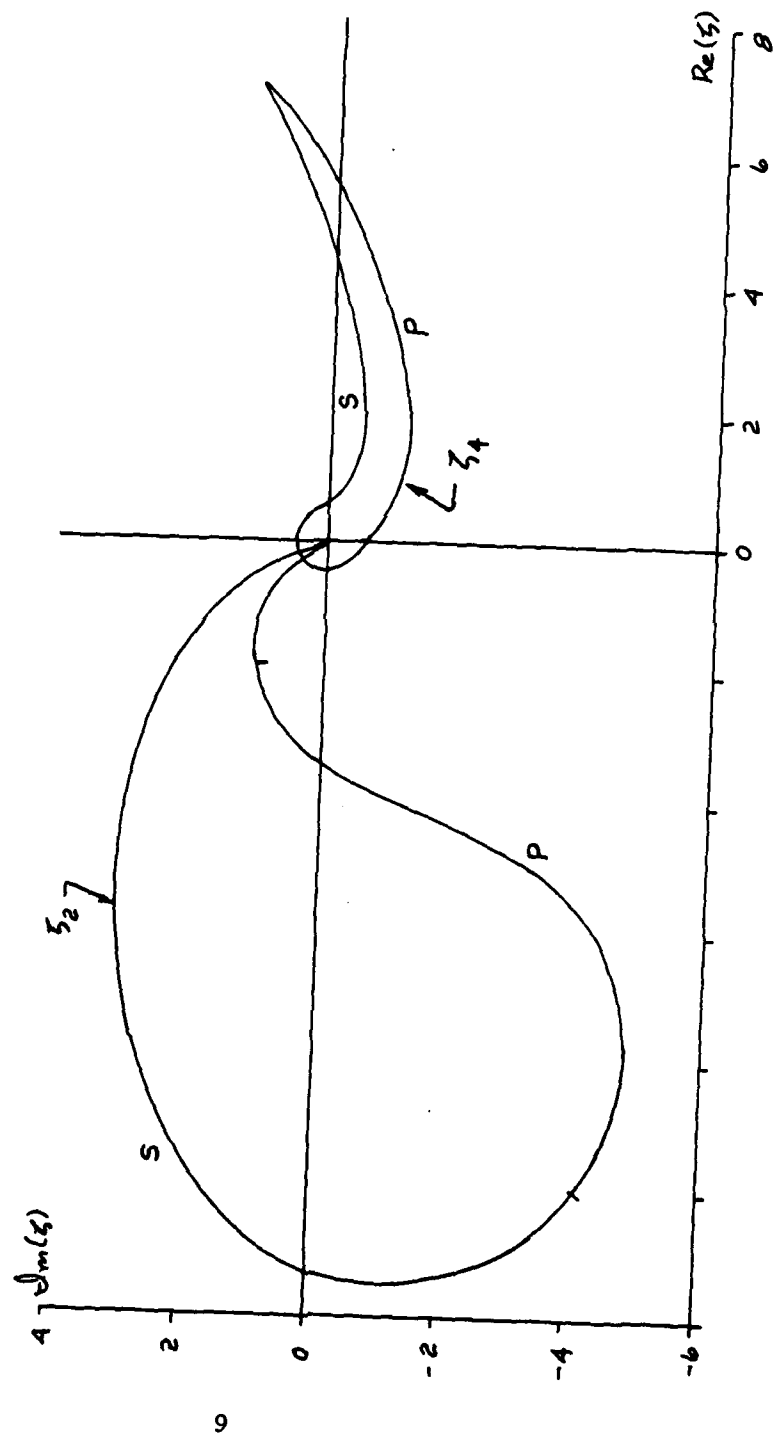


Figure 6. The Transformations ζ_2 and ζ_4

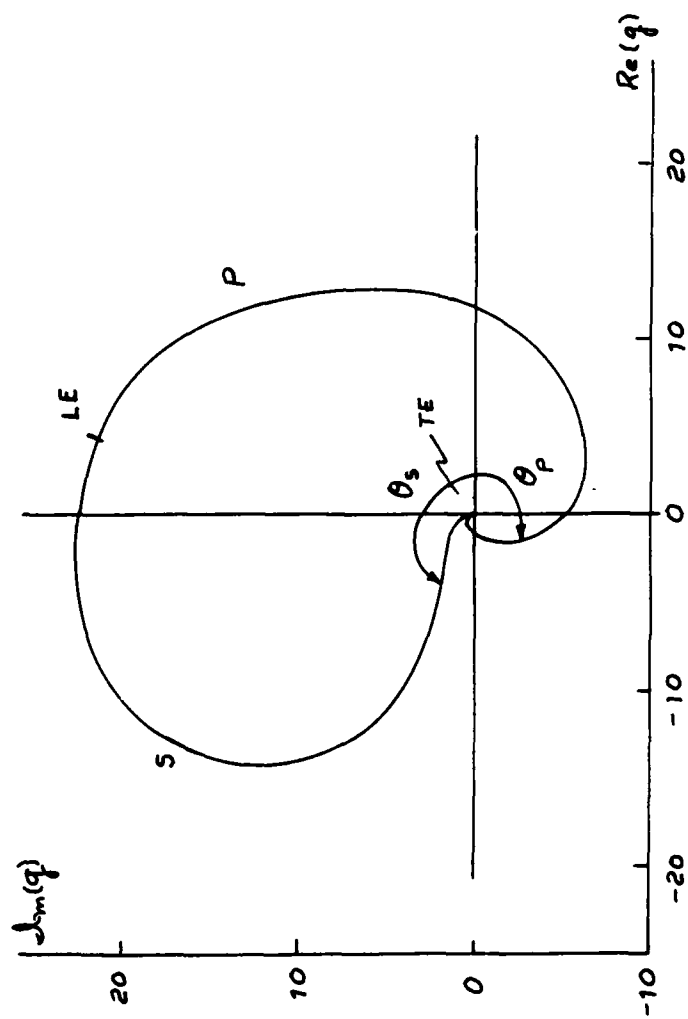


Figure 7. The Mapping $q(z)$

The next step is the exponentiation, defined as:

$$\Omega = [g(z)]^{1/K} ; \quad K = 2 - \frac{\tau}{\pi} = EXINV \equiv \frac{1}{EX} \quad (2-8)$$

The result of this step is shown in Fig. 8. As noted above, the value of EX for a sharp-trailing-edge blade is fixed by the angle τ , whereas for a round trailing edge a range of values of EX , in the neighborhood of 0.3, will produce a curve in the Ω -plane that is "star-shaped", i.e., its polar coordinates will have a radius that is a single-valued function of the angular coordinate.

The next transformation is a bilinear one, whose purpose is to produce a curve in the ω -plane that can be mapped into a unit circle in a subsequent step:

$$c \frac{\omega - a}{\omega - b} = \Omega ; \quad \omega = \frac{a - b \frac{\Omega}{c}}{1 - \frac{\Omega}{c}} \quad (2-9)$$

where a , b , and c are complex constants. Three conditions must be assigned, in order to evaluate these constants: Ives suggests, for two of them, that the images of upstream and downstream infinity be mapped into $\omega = \pm 1$. As the third condition, he recommends that the centroid of the blade-surface image in the ω -plane be forced to be close to the origin. This condition was applied in Ref. 1; however, it has been found simpler to impose a condition on the ratio between the maximum and minimum radii in the ω -plane, as outlined below. The calculation of the centroidal location has been retained in the present code, for informational purposes. (Details on how this is calculated can be found in Ref. 1).

The locations of the images of upstream and downstream infinity in the g - and Ω -planes can be expressed as follows: in general,

$$g(z) \equiv \frac{\sin \zeta_1}{\sin \zeta_3} = \frac{\sin(\operatorname{Re} \zeta_1) \cosh(\operatorname{Im} \zeta_1) + i \cos(\operatorname{Re} \zeta_1) \sinh(\operatorname{Im} \zeta_1)}{\sin(\operatorname{Re} \zeta_3) \cosh(\operatorname{Im} \zeta_3) + i \cos(\operatorname{Re} \zeta_3) \sinh(\operatorname{Im} \zeta_3)}$$

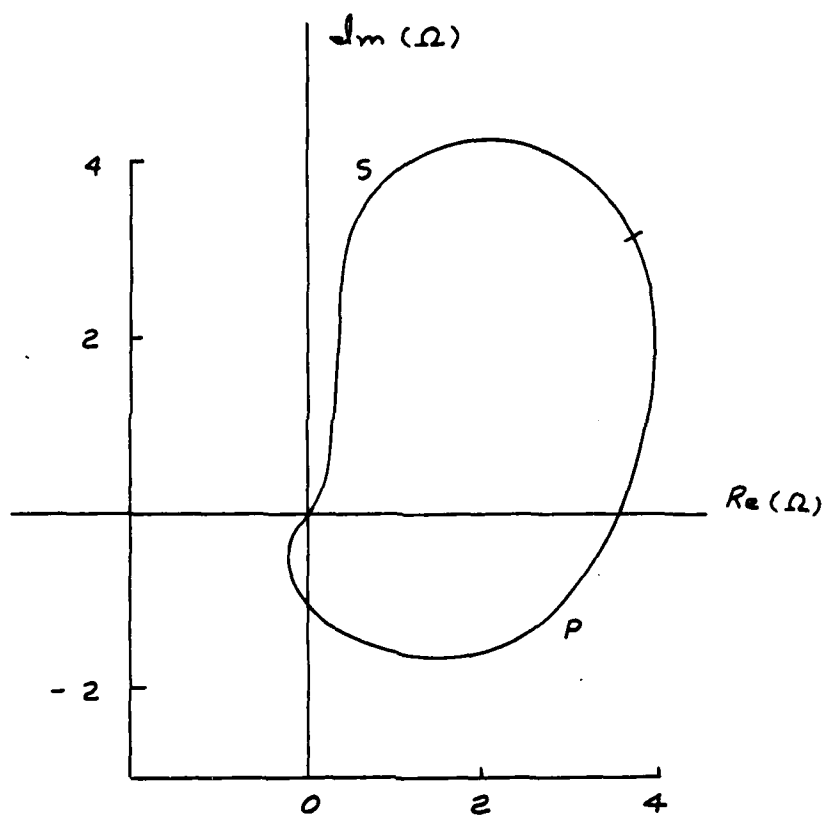


Figure 8. The Mapping $\Omega(z)$

As $z \rightarrow -\infty$ along line ① of Fig. 9,

$$\operatorname{Re} \zeta_1 = \text{finite}, \operatorname{Im} \zeta_1 \rightarrow +\infty, \operatorname{Re} \zeta_3 = 0, \operatorname{Im} \zeta_3 \rightarrow +\infty$$

Using the large-argument approximations for the hyperbolic functions then gives

$$q(-\infty) = e^{\operatorname{Im}(\zeta_1 - \zeta_3)} e^{-i \operatorname{Re} \zeta_1} = e^{x \cdot K} e^{i x_A \cdot K} \quad (2-10)$$

where a little algebra reveals that

$$x \cdot K \equiv \operatorname{Im}(\zeta_1 - \zeta_3) = \frac{\pi}{S_G} \{ \operatorname{Re}(\bar{z}_T - \bar{z}_N) \cos \gamma - \operatorname{Im}(\bar{z}_T - \bar{z}_N) \sin \gamma \}$$

$$x_A \cdot K \equiv \operatorname{Re}(\zeta_1) = -\frac{\pi}{S_G} \{ \operatorname{Re}(\bar{z}_T - \bar{z}_N) \sin \gamma + \operatorname{Im}(\bar{z}_T - \bar{z}_N) \cos \gamma \}$$

As $z \rightarrow +\infty$ along line ② of Fig. 9,

$$\operatorname{Re} \zeta_1 = 0, \operatorname{Im} \zeta_1 \rightarrow -\infty, \operatorname{Re} \zeta_3 = \text{finite}, \operatorname{Im} \zeta_3 \rightarrow -\infty$$

and these lead to

$$q(+\infty) = e^{-\operatorname{Im}(\zeta_1 - \zeta_3)} e^{-i \operatorname{Re}(\zeta_3)}$$

where $\operatorname{Re}(\zeta_3) = -\operatorname{Re}(\zeta_1)$. Thus

$$q(+\infty) = e^{-x \cdot K} e^{-i x_A \cdot K} \quad (2-11)$$

and

$$\Omega^- = [q(-\infty)]^{1/K} = e^x e^{i x_A}, \quad \Omega^+ = [q(+\infty)]^{1/K} = e^{-x} e^{-i x_A} \quad (2-12)$$

Note that

$$\Omega^+ \Omega^- = 1$$

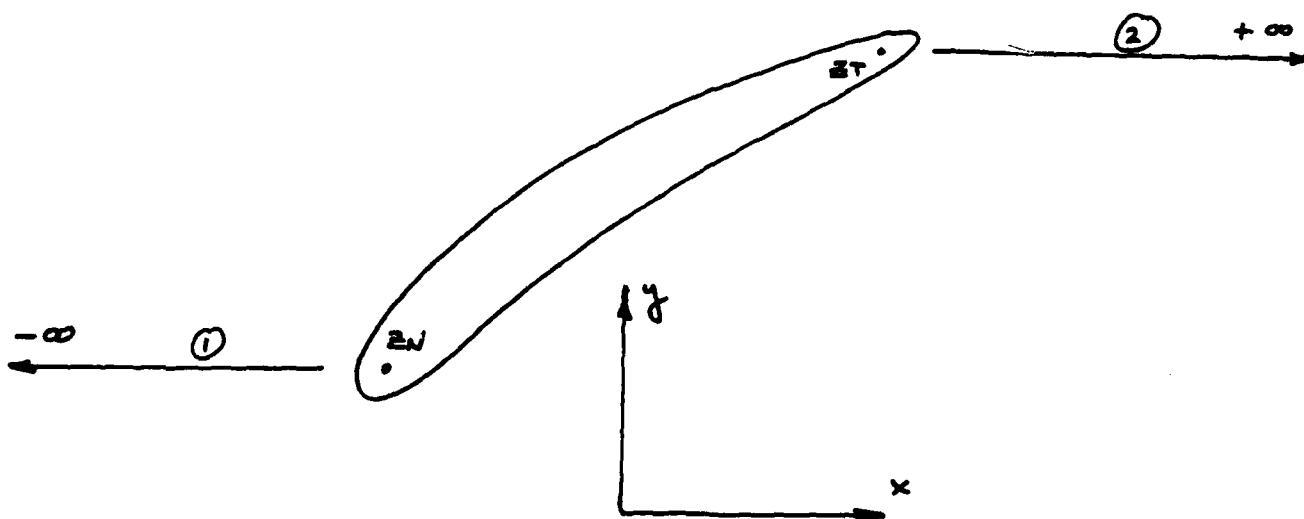


Figure 9. Location of the Points at Infinity

The constants a and b can be expressed in terms of C by the two equations:

$$C \frac{-1-a}{-1-b} \equiv \Omega^-$$

$$C \frac{1-a}{1-b} \equiv \Omega^+$$

The solution is:

$$\begin{aligned} b &= \frac{-2C + \Omega^- + \Omega^+}{\Omega^+ - \Omega^-} ; \quad a = 1 - (1-b) \frac{\Omega^+}{C} \\ &= -Ec + F ; \quad = \frac{E}{C} - F \end{aligned} \quad (2-13)$$

where E and F are known quantities:

$$E \equiv \frac{2}{\Omega^+ - \Omega^-} , \quad F \equiv \frac{\Omega^+ + \Omega^-}{\Omega^+ - \Omega^-} \quad (2-14)$$

In terms of the parameters E , F , and C , the transformation can be written as

$$\omega = \frac{E - Fc + (Ec - F) \Omega}{C - \Omega} \quad (2-15)$$

$$\Omega = \frac{(\omega + F)c - E}{\omega - F + Ec} \quad (2-16)$$

The latter relations can be used in an iteration procedure to find a value of C that will minimize the ratio R_{MAX}/R_{MIN} , where R_{MAX} and R_{MIN} are the maximum and minimum values of $|\omega|$ over the set defined by $KJ=1$ to $KJMX$. The iteration process is as follows: on alternate iterations, values of C are chosen that will either reduce R_{MAX} or increase R_{MIN} . This is done by solving Equation 2-16 for C :

$$C = \frac{\Omega \omega + E - F \Omega}{\omega + F - E \Omega} \quad (2-17)$$

On the first iteration, $C = 1 + i$ is used; the values of R_{MIN} and the index $KJ=KJMN$ at which it occurs are then found. Next, a new value of C is calculated,

using Eq. 2-17, such that the new value of ω (KJMN) will equal 1.1 times the value just found. For this new mapping into the ω -plane, the value of RMAX and the index KJ=KJMX at which it occurs are found. For the third iteration, a value of C is used such as to generate a new value of ω (KJMX) equal to 0.9 times the previous one. This alternating cycle is then continued until the ratio RMAX/RMIN is less than the tolerance RTOL. This tolerance is assigned a default value of 3.0; values up to 6.0 have been handled successfully by the subsequent steps in the transformation.

The iteration on C can be bypassed, if C is already known, by setting IGOT=1 and reading in the value of C . Figure 10, reproduced from Ref. 1, was generated this way.

Next, the blade-surface image in the ω -plane is mapped into the unit circle in the ζ - plane with the trailing edge at $\zeta = 1$ by a variant of the Theodorsen-Garrick transformation:

$$\omega = \zeta \exp \left\{ \sum_{j=0}^N (A_j + i B_j) \zeta^j \right\} \quad (2-18)$$

To determine the coefficients, the values of ω and ζ on the blade surface are written as

$$\omega = r(\theta) e^{i\theta}, \quad \zeta = 1 e^{i\phi} \quad (2-19)$$

Then the real and imaginary parts of the transformation are

$$\ln r = A_0 + \sum_{j=1}^N [A_j \cos j\phi - B_j \sin j\phi] \quad (2-20)$$

$$\theta = \phi + B_0 + \sum_{j=1}^N [A_j \sin j\phi + B_j \cos j\phi] \quad (2-21)$$

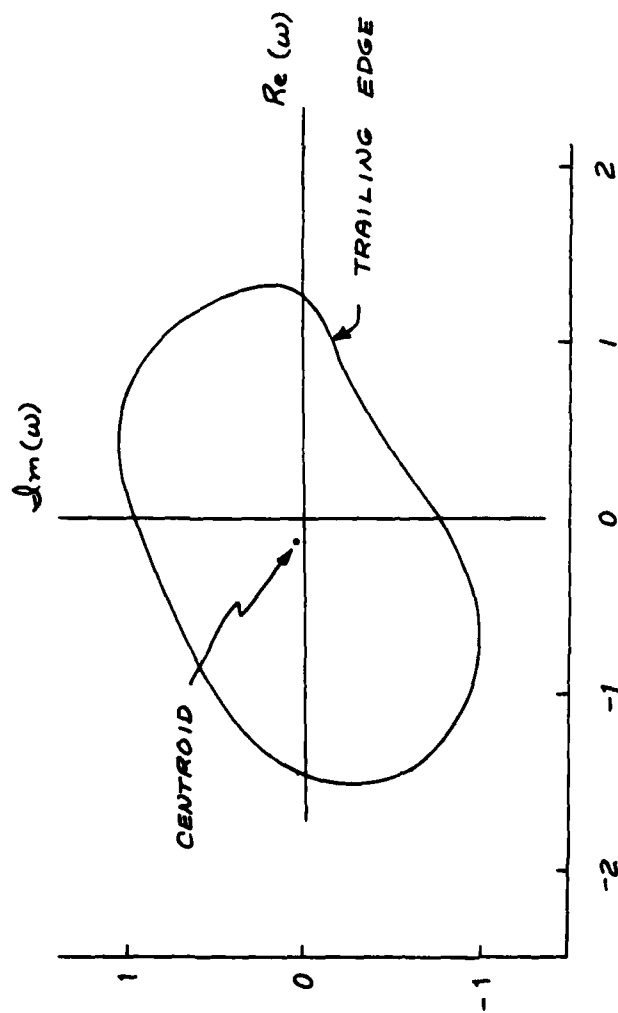


Figure 10. The Mapping $\omega(z)$

The coefficients are then determined by the following iteration procedure: an equally-spaced array of values of ϕ is set up, and all the coefficients are initially set equal to zero. Then the second equation gives $\theta = \phi$ as the first approximation for θ . For each of these values of θ , a corresponding value of $\ln r$ can be found, from a spline fit to the coordinates of the blade-surface image in the ω -plane. These known values of $\ln r$ are then used in the first of the equations above, to find the next approximation to the A_i and B_i coefficients. These coefficients can then be used to give the next approximation to $\theta(\phi)$, and the process is continued until convergence is reached, to some preassigned tolerance. Fast Fourier transform techniques⁶ can be used in the processes of evaluating the second equation, for known values of the coefficients, and of determining the coefficients from the first equation with known values of $\ln r$. These techniques were applied in the present calculations. The details are given in Appendix A.

Sufficient conditions for convergence of this iteration process have been discussed by Warschawski.⁷ For the present case, these conditions are not met; in particular, it is required that the maximum and minimum values of $r(\theta)$ obey the relation:

$$\sqrt{\frac{R_{MAX}}{R_{MIN}}} - 1 < 0.295 \quad \text{or} \quad \frac{R_{MAX}}{R_{MIN}} < 1.678$$

This condition is not met in the present case, where R_{MAX}/R_{MIN} is approximately 2.5. It was found, however, that the iteration process would converge

6. Cooley, J.W., Lewis, P.A.W., and Welch, P.D., "The Fast Fourier Transform Algorithm: Programming Considerations in the Calculations of Sine, Cosine and Laplace Transforms", *Journal of Sound and Vibration* 12, (1970) pp. 315-337.
7. Warschawski, S.E., "On Theodorsen's Method of Conformal Mapping of Nearly Circular Regions", *Quarterly of Applied Mathematics* 3, (1945) pp. 12-28.

if a relaxation parameter was used, i.e., the values of θ called for by the second equation [called θ^*] were not used in the first equation, but were replaced by $\theta_{\text{new}} = 0.1 \theta^* + 0.9 \theta_{\text{old}}$. With this relaxation factor, the iterations were convergent: after 68 iterations, the maximum change in any of the values of θ was less than 10^{-2} radians. The variation of θ with ϕ is shown in Fig. 11. Calculations for other cases, not shown here, have required relaxation factors as low as 0.02 for convergence. A recent review paper by Henrici (Reference 8) calls attention to the applicability of under-relaxation in this problem.

The next transformation is

$$\eta = \gamma \frac{\zeta - \alpha}{\zeta - \beta} \quad (2-22)$$

where α , β , and γ are chosen so as to place the images of $z = \pm \infty$ at $\eta = \pm S$, while the blade surface continues to be the unit circle. These images are located, respectively, at $\omega = \pm 1$, and $\zeta = \zeta_A, \zeta_B$. Explicit formulas for α , β , and γ in terms of ζ_A and ζ_B are given by Ives³ as

$$x = \frac{2 - |\zeta_A + \zeta_B|^2 + 2 |\zeta_A \zeta_B|^2}{|\zeta_A - \zeta_B|^2}$$

$$S = \min \left\{ \sqrt{|x + \sqrt{x^2 - 1}|}, \sqrt{|x - \sqrt{x^2 - 1}|} \right\}$$

$$\alpha = \frac{2 \zeta_A \zeta_B + [S^2 (\zeta_A - \zeta_B) - (\zeta_A + \zeta_B)] / \tilde{\zeta}_A}{S^2 (\zeta_A - \zeta_B) + (\zeta_A + \zeta_B) - 2 / \tilde{\zeta}_A}$$

8. Henrici, P., "Fast Fourier Methods in Computational Complex Analysis". SIAM Review 21, (1979) pp. 481-527.

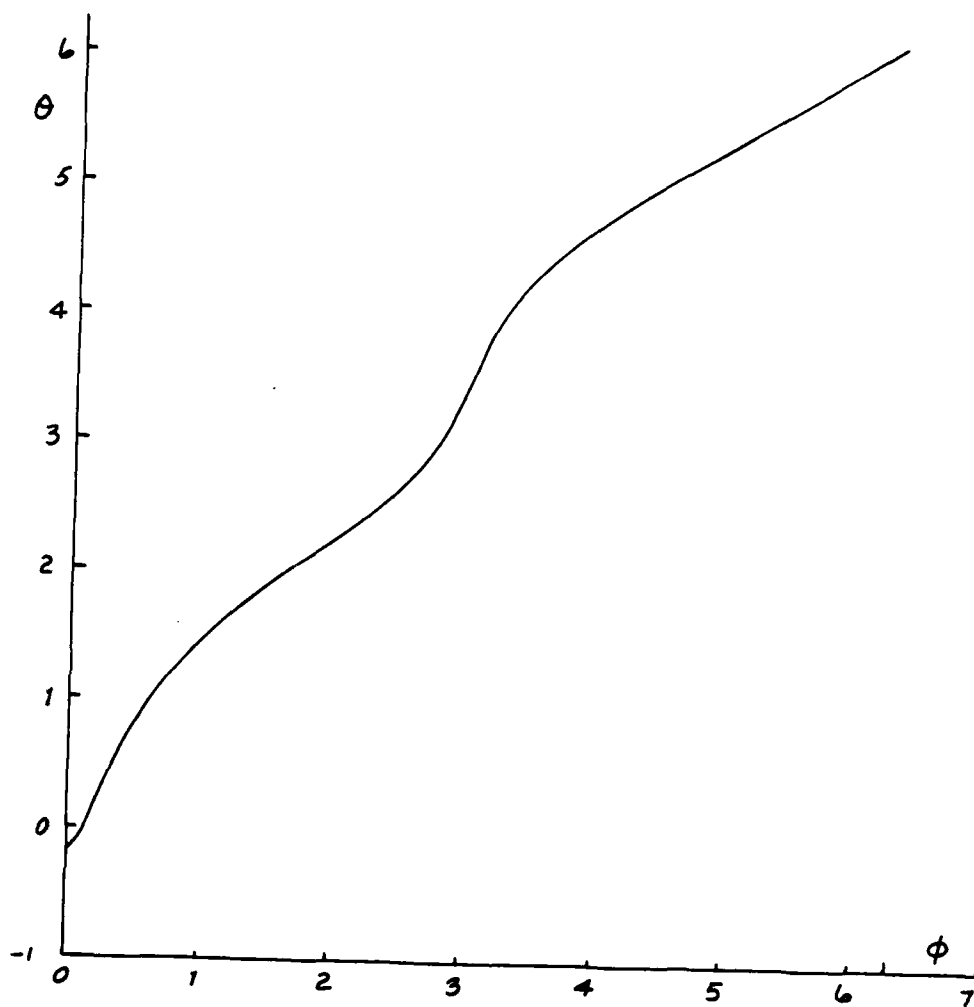


Figure 11. Variation of θ vs. ϕ on the Blade Surface

$$\beta = \frac{2 \zeta_A \zeta_B - \alpha (\zeta_A + \zeta_B)}{\zeta_A + \zeta_B - 2\alpha}$$

$$\gamma = S \frac{\zeta_A - \beta}{\zeta_A - \alpha} \quad (2-23)$$

Mokry (Reference 9) has pointed out that the formula for S can be simplified, as follows: define

$$C = \frac{\zeta_A \bar{\zeta}_B - 1}{\zeta_A - \zeta_B}$$

Then

$$S = |C| - \sqrt{|C|^2 - 1}, \quad \gamma = \frac{C - S|C| \bar{\zeta}_B}{|C| \bar{\zeta}_B - SC}$$

$$\alpha = \frac{\zeta_B C - S|C|}{C - S|C| \bar{\zeta}_B}, \quad \beta = \frac{|C| - SC \zeta_B}{|C| \bar{\zeta}_B - SC} \quad (2-24)$$

The computer program described below does not use these simplifications.

Next, it is necessary to find ζ_A and ζ_B , given $\omega = \pm 1$. This was done by Newton-Raphson iteration:

$$\zeta^{(n+1)} = \zeta^{(n)} - \frac{G(\zeta)}{\frac{dG}{d\zeta}}$$

9. Mokry, M., "Comment on Analysis of Transonic Cascade Flow Using Conformal Mapping and Relaxation Techniques", AIAA Journal 16, No. 1, (January 1978) p. 96.

where

$$G(\zeta) = \zeta \exp \left\{ \sum_{j=0}^N (A_j + iB_j) \zeta^j \right\} - \omega$$

$$\frac{dG}{d\zeta} = \frac{\omega}{\zeta} \left[1 + \sum_{j=1}^N j(A_j + iB_j) \zeta^{j-1} \right]$$

In order to find an initial guess for ζ , a preliminary calculation was made, for

$$|\zeta| = 0.49 \text{ (.1) } 0.99, \quad \arg \zeta = -60^\circ \text{ (10) } + 60^\circ$$

From this set, the value of ζ which gave ω nearest to +1 was chosen as the initial guess. This set was then repeated with ζ replaced by $-\zeta$, to obtain the value of ζ that gave ω nearest to -1. The locations of key points in the ζ and η planes are shown in Fig. 12.

When the values of ζ_A and ζ_B are known, the blade-surface-image points can be mapped from the ζ -plane to the η -plane. In both planes, these points are located on unit circles, so it is only necessary to interpolate in Fig. 12 to find the ζ -values for the points defining the blade surface. This is done with a call to the spline-fit subroutine, and the values returned by it are replaced by linear interpolation in regions where the θ, ϕ curve is so steep that the spline fit returns non-monotonic values.

The final transformation now uses an elliptic function to map the unit circle in the η -plane (with cuts along the real axis from $\pm S$ to the circle) into a rectangle:

$$\eta = S \operatorname{sn}(\tilde{\xi}, k) \quad (2-25)$$

where the parameter k is given by

$$k = S^2 \quad (2-26)$$

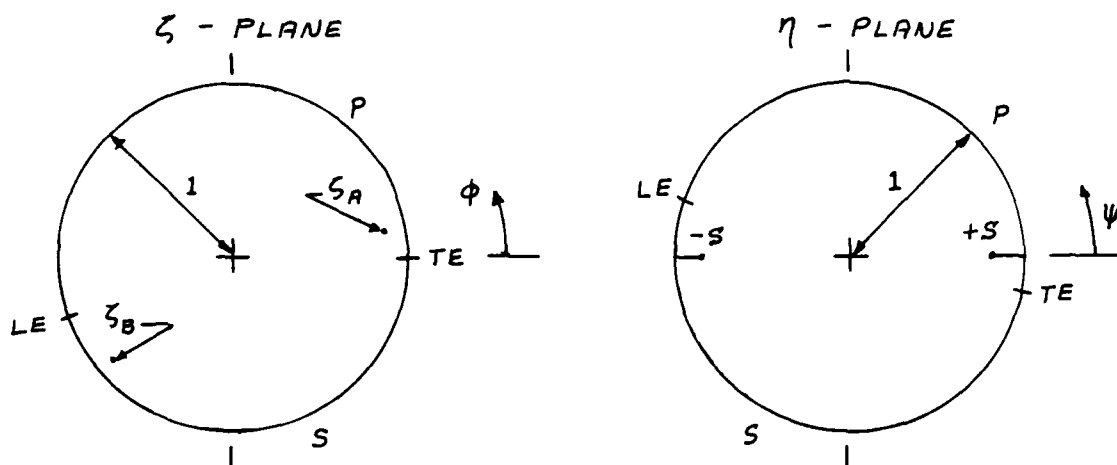


Figure 12. The ζ and η Planes

The inverse of this transformation is

$$\tilde{\xi} = \int_0^{\eta/s} \frac{dt}{\sqrt{1-t^2} \sqrt{1-k^2 t^2}} \quad (2-27)$$

This latter transformation is used to find the images, in the $\tilde{\xi}$ -plane, of the blade-surface points. This requires an expression for the real and imaginary parts of the incomplete elliptic integral of the first kind; convenient formulae for this purpose are given by Nielsen and Perkins¹⁰ who show that, if

$$\eta = S(\tau + i\delta) \quad (2-28)$$

then

$$\tilde{\xi} = F(\sqrt{\lambda}, k) + iF(\sqrt{\sigma}, k') \quad (2-29)$$

where

$$k' = \sqrt{1-k^2} = \sqrt{1-S^2} \quad (2-30)$$

10. Nielsen, J.N. and Perkins, E.W., "Charts for the Conical Part of the Downwash Field of Swept Wings at Supersonic Speeds", NACA Technical Note 1780, (December 1948), Appendix C.

and where $F(\dots)$ denotes the incomplete elliptic integral of the first kind, with real arguments λ and σ given as functions of τ, δ and k :

$$\begin{aligned}\lambda &= \left[1 + \tau^2 + \delta^2 - \sqrt{(1 - \tau^2)^2 + \delta^2(\delta^2 + 2 + 2\tau^2)} \right] \left[1 + k^2(\tau^2 + \delta^2) \right. \\ &\quad \left. - \sqrt{(1 - k^2\tau^2)^2 + k^2\delta^2(2 + 2k^2\tau^2 + k^2\delta^2)} \right] / 4k^2\tau^2 \\ \sigma &= [\tau^2 + \delta^2 - \lambda] / [\tau^2 + \delta^2 - \lambda + 1 - \lambda k^2(\tau^2 + \delta^2)]\end{aligned}\quad (2-31)$$

These formulae are equivalent to those following Eqn. 115.01 of Byrd and Friedman;¹¹ the formula for λ has been rearranged slightly from the form given by Nielsen and Perkins, to avoid the occurrence of negative values under the square root sign, which can sometimes happen in the numerical evaluations when $\delta = 0$. These formulas are correct along the branch cuts, where $\delta = 0$ and $|\eta|/\delta > 1$.

Numerical evaluations of these elliptic integrals were done using twelve terms in the formulae of Luke:¹²

$$\begin{aligned}F(y, k) &\equiv \int_0^y \frac{dt}{\sqrt{1-t^2} \sqrt{1-k^2t^2}} = \int_0^{\phi = \sin^{-1}y} \frac{d\psi}{\sqrt{1-k^2\sin^2\psi}} \\ &\approx F_{12}(\phi, k) = \frac{1}{25} \left[\phi + 2 \sum_{m=1}^{12} \frac{\tan^{-1}(\sigma_m \tan \phi)}{\sigma_m} \right]\end{aligned}\quad (2-32)$$

11. Byrd, P.F., and Friedman, M.D., Handbook of Elliptic Integrals for Engineers and Physicists. Springer Verlag, Berlin (1954).

12. Luke, Y.L., "Approximations for Elliptic Integrals", Mathematics of Computation, 22 (1968) 627-634.

where

$$\phi < \pi/2, \quad \sigma_m = \sqrt{1 - k^2 \sin^2 \theta_m}, \quad \theta_m = m\pi/25$$

For the case where $\phi = \pi/2$ (the complete integral) the approximate formula is

$$F_{12}\left(\frac{\pi}{2}, k\right) = \frac{\pi}{50} \left[1 + 2 \sum_{m=1}^{12} \frac{1}{\sigma_m} \right] \quad (2-33)$$

The signs in the formulas above pertain to the first quadrant in the η -plane ($\tau \geq 0, \delta \geq 0$), which maps into a rectangle in the first quadrant of the $\tilde{\xi}$ -plane, with sides located on the lines

$$\begin{aligned} \operatorname{Re}(\tilde{\xi}) &= K(k) = \int_0^1 \frac{dt}{\sqrt{1-t^2} \sqrt{1-S^4 t^2}} \\ \operatorname{Im}(\tilde{\xi}) &= \frac{1}{2} K'(k) = \frac{1}{2} \int_0^1 \frac{dt}{\sqrt{1-t^2} \sqrt{1-(1-S^4) t^2}} \end{aligned} \quad (2-34)$$

The remaining three quadrants in the η -plane map into the remaining three quadrants in the $\tilde{\xi}$ -plane, as described in Reference (13), p. 377; the cuts from $\pm S$ to the unit circle along the real axis become the left and right sides of the rectangle in the $\tilde{\xi}$ -plane:

13. Erdelyi, A., et al. Higher Transcendental Functions, Volume 2, p. 377, McGraw-Hill Book Company, New York (1953).

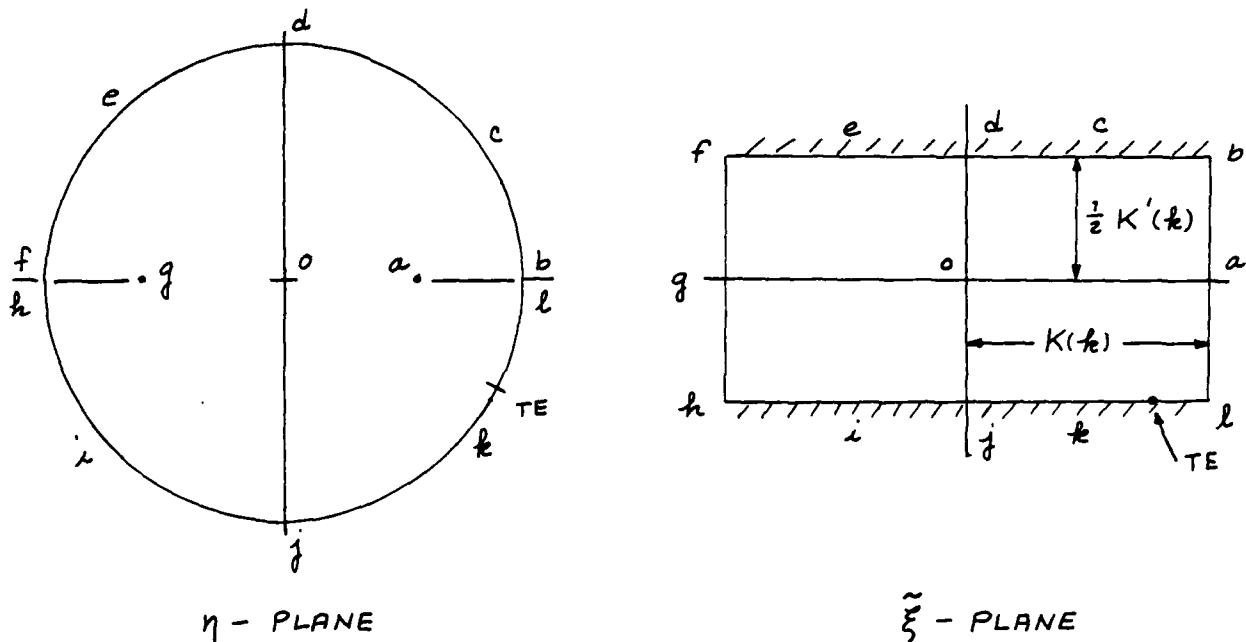


Figure 13. The η and $\tilde{\xi}$ Planes

Finally, the $\tilde{\xi}$ - plane is re-normalized, so as to lie between -1 and +1 on both axes:

$$\tilde{z}_{\text{MAPPED}} \equiv \frac{\text{Re}(\tilde{\xi})}{K(k)} + i \frac{\text{Im}(\tilde{\xi})}{\frac{1}{2} K'(k)} \quad (2-35)$$

A grid is now to be set up in the $\tilde{\xi}$ plane, and mapped back to the z - plane. This process is facilitated by first rearranging the quadrants in the $\tilde{\xi}$ - plane, by using the periodicity of the elliptic functions as follows: in the first and second quadrants, let $\hat{\xi} = \tilde{\xi}$, and in the third and fourth let $\hat{\xi} = -(\tilde{\xi} + 2K(k))$ and use the relations (see for example, Eqs. 122.00 and 122.04 of Reference 11:)

$$\operatorname{sn}(\tilde{\xi}) = -\operatorname{sn}(\tilde{\xi} + 2K) = \operatorname{sn}[-(\tilde{\xi} + 2K)] = \operatorname{sn}[-K - (\tilde{\xi} + K)] \quad (2-36)$$

The $\hat{\xi}$ plane then has the form:

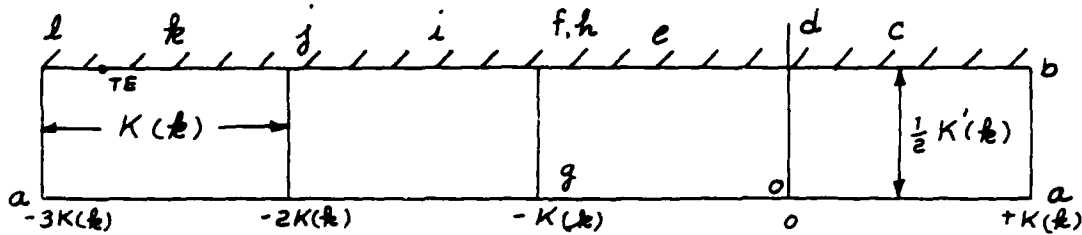


Figure 14. The $\hat{\xi}$ Plane

Two types of grid can be selected in the $\hat{\xi}$ plane: a rectangular one (if ISHEAR=0) or a sheared one (if ISHEAR=1). The latter is the default.

For the rectangular grid, equally spaced points are assigned, according to

$$\hat{\xi}_R \equiv \operatorname{Re}(\hat{\xi}) = (K-1) \Delta \hat{\xi}_{\text{real}}, \quad K = 1, 2, \dots, KMX \quad (2-37)$$

$$\hat{\xi}_I \equiv \operatorname{Im}(\hat{\xi}) = \frac{1}{2} K'(k) - (L-1) \Delta \hat{\xi}_{\text{imag}}, \quad L = 1, 2, \dots, LMX$$

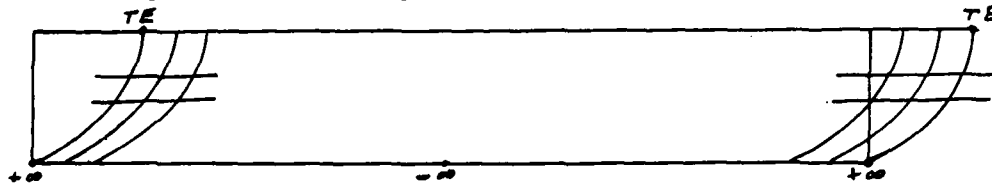
where

$$\Delta \hat{\xi}_{\text{real}} = \frac{4K(k)}{KMX-1}, \quad \Delta \hat{\xi}_{\text{imag}} = \frac{\frac{1}{2} K'(k)}{LMX-1} \quad (2-38)$$

Because the mapping is conformal, the images of these grid lines will intersect at right angles when mapped back to the physical plane.

The rectangular grid has the property that in general the trailing edge is not connected, by a grid line, to the point at downstream infinity. However, such a connection is a desirable feature in certain flowfield codes

(see Ref. 14, for example). To allow this feature, the ISHEAR=1 option establishes a sheared grid, consisting of the same $\text{Im}(\hat{\xi})$ lines as above, but replacing the $\text{Re}(\hat{\xi})$ lines by a set of parabolas which intersect the blade surface at 90 degrees, and are displaced from a base parabola that connects the trailing edge to the image of downstream infinity:



This grid is given by

$$\begin{aligned}\hat{\xi}_I &= \frac{1}{2} K'(\frac{k}{L}) - (L-1) \Delta \hat{\xi}_{imag}, \quad L = 1, 2, \dots, LMX \\ \hat{\xi}_R &= \hat{\xi}_R(TE) + (K-1) \Delta \hat{\xi}_{real} \\ &\quad + \frac{[\hat{\xi}_I - \frac{1}{2} K'(\frac{k}{L})]^2}{const}, \quad K = 1, 2, \dots, KMX \\ &\quad L = 1, 2, \dots, LMX\end{aligned}\quad (2-39)$$

where

$$const = \frac{-[K'(\frac{k}{L})]^2}{4[3K(\frac{k}{L}) + \hat{\xi}_R(TE)]}; \quad \hat{\xi}_R(TE) = \text{Re} \left\{ \hat{\xi} [KJ=1] \right\} \quad (2-40)$$

Some points on these parabolas will lie outside the range

$$-3K(\frac{k}{L}) \leq \hat{\xi}_R \leq +K(\frac{k}{L})$$

When this occurs, equivalent points are found by adding or subtracting $\pm K(\frac{k}{L})$, the period of the elliptic sine. In addition, the base parabola is always joined to the lower-left corner of Fig. 15, by subtracting $4K(\frac{k}{L})$ from the real part of $\hat{\xi}_{TE}$, if the latter is greater than $-K(\frac{k}{L})$.

This completes the definition of the grid in the $\hat{\xi}$ -plane. Each of these grid points must now be mapped back to the physical plane. The first

14. Nenni, J.P. and Rae, W.J., "Experience with the Development of an Euler Code for Rotor Rows", ASME Paper 83-GT-36 (March 1983).

transformation is: $\eta = \mathcal{S} \operatorname{sn}(\tilde{\xi}; k) = \mathcal{S} \operatorname{sn}(\xi; k)$ (2-41)

The elliptic sine of a complex argument is expressed as (Ref. 11, Eq. 125.01)

$$\operatorname{sn}(u + i v, k) = \frac{\operatorname{sn}(u, k) \operatorname{dn}(v, k') + i \operatorname{cn}(u, k) \operatorname{dn}(u, k) \operatorname{sn}(v, k') \operatorname{cn}(v, k')}{1 - \operatorname{sn}^2(v, k') \operatorname{dn}^2(u, k)} \quad (2-42)$$

The functions in this expression are evaluated by the Arithmetic-Geometric Mean method (see Ref. 15, p. 571):

Set

$$a_0 = 1, \quad b_0 = k', \quad c_0 = k$$

and then calculate

$$\begin{aligned} a_n &= \frac{1}{2} (a_{n-1} + b_{n-1}), \quad b_n = \sqrt{a_{n-1} b_{n-1}} \\ c_n &= \frac{1}{2} (a_{n-1} - b_{n-1}) \end{aligned} \quad (2-43)$$

until $c_n = 0$ to a prescribed tolerance (10^{-7} was used in the present case.)

Then form

$$\varphi_N = 2^N \cdot a_n \cdot u$$

and calculate $\varphi_{N-1}, \varphi_{N-2}, \dots, \varphi_0$ from

$$\varphi_{n-1} = \frac{1}{2} \left\{ \varphi_n + \arcsin \left(\frac{c_n}{a_n} \sin \varphi_n \right) \right\} \quad (2-44)$$

Then the desired results are given by

-
15. Abramowitz, M. and Stegun, I.A., Handbook of Mathematical Functions, National Bureau of Standards, Applied Mathematics Series 55 (1964).

$$\operatorname{sn}(u, k) = \sin \varphi_0, \quad \operatorname{cn}(u, k) = \cos \varphi_0,$$

$$\operatorname{dn}(u, k) = \frac{\cos \varphi_0}{\cos(\varphi_1 - \varphi_0)} \quad (2-45)$$

These values of η are then mapped back to the ζ - plane by

$$\zeta = \frac{\beta \eta - \alpha \gamma}{\eta - \gamma} \quad (2-46)$$

and thence to the ω - plane by

$$\omega = \zeta \exp \left[\sum_{j=0}^N (A_j + i B_j) \zeta^j \right] \quad (2-47)$$

Lastly, the value of Z must be found by inverting:

$$\Omega \equiv C \frac{\omega - a}{\omega - b} = \left\{ \frac{\sin \pi \frac{Z - Z_T}{H + iG}}{\sin \pi \frac{Z - Z_N}{H + iG}} \right\}^{1/K} \equiv [g(Z)]^{1/K} \quad (2-48)$$

This is written as

$$F(Z) = g(Z) - \Omega^K \quad (2-49)$$

and is solved by a Newton-Raphson procedure

$$Z^{(n+1)} = Z^{(n)} - \frac{F(Z^{(n)})}{F'(Z^{(n)})} \quad (2-50)$$

where

$$F'(Z) = \frac{\pi}{H + iG} \frac{\sin \frac{\pi(Z_T - Z_N)}{H + iG}}{\sin^2 \pi \frac{Z - Z_N}{H + iG}} \quad (2-51)$$

The final result of this process is shown in Fig. 15, for a rectangular grid (examples of sheared grids can be found in Refs. 4 and 15). Note that the grid line in Fig. 15 which goes to downstream infinity does not originate at the trailing edge.

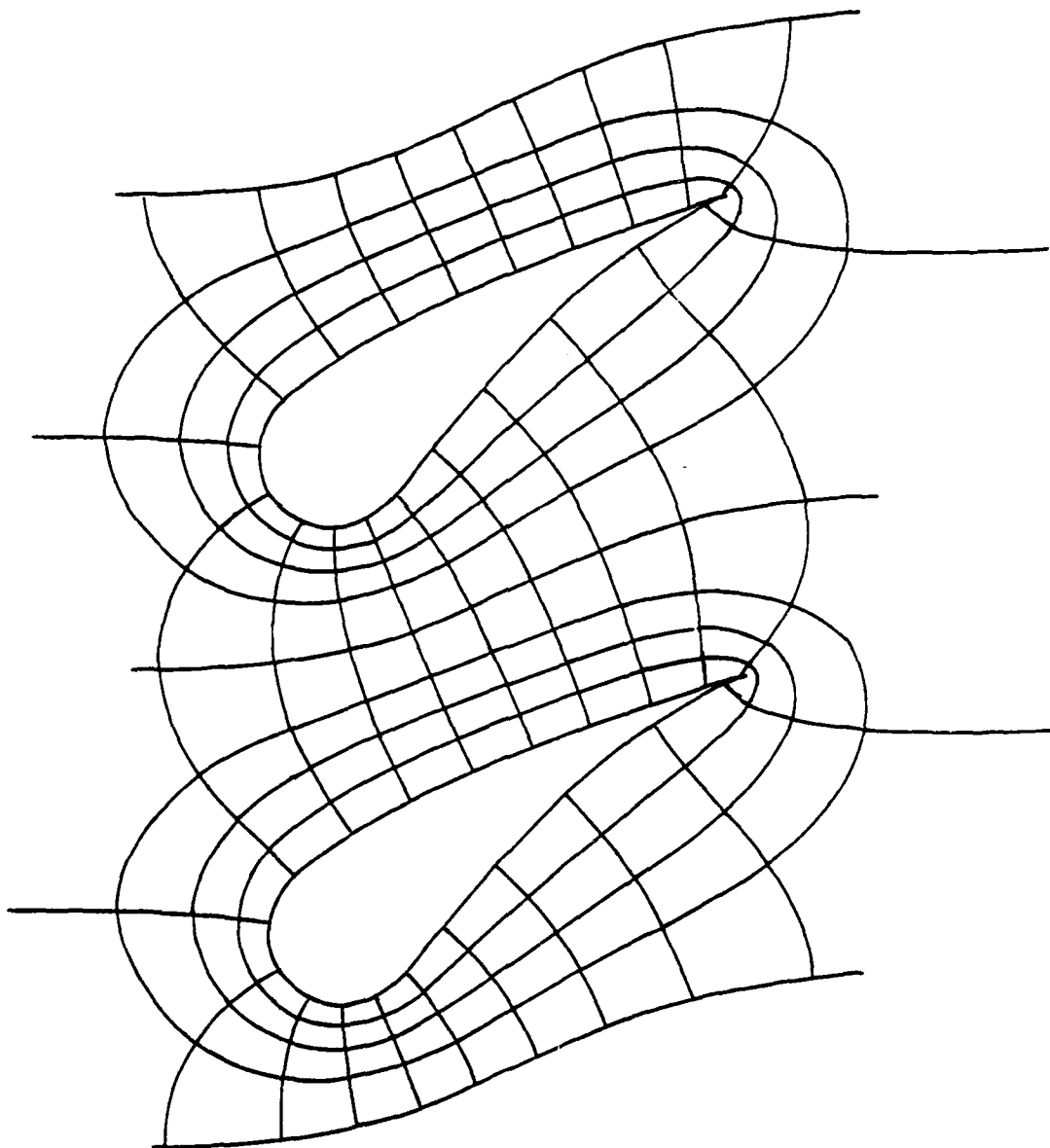


Figure 15. Coordinate Mapping

Section 3

METRIC EVALUATIONS

The coordinate transformation enters the flowfield solution algorithm only in the metric derivatives. These can be evaluated by differencing the coordinates themselves, or in the case of a conformal transformation, by evaluating the analytic expressions for them. These analytic expressions are derived in Ref. 1, and the code listed in that report contains the Fortran statements required to evaluate these expressions. However, as pointed out in Ref. 14, the truncation error resulting from the use of analytic metrics in the finite-difference flowfield code is large enough to cause major instabilities in the solution algorithm. It is highly preferable to use metrics that are found by differencing the coordinates in the same manner as the flowfield variables are differenced. A program to achieve this, for the grid conventions used in Ref. 14, is given in Appendix D.

Section 4

COMPUTER PROGRAM

A listing of the computer program is given in Appendix B, and a dictionary of variables is given in Appendix C. This section contains a general description of the program, plus some specific details.

In order to handle complex arithmetic, all variables beginning with the letter Z are declared to be complex by an implicit type specification at the beginning of the program.

The input is generally described by comment cards in the deck. Certain blade-shape parameters must be read in: EX, G, H, ZLE, ZTE, ZN, ZT. Also, if IGOT=1, ZC must be read in. The blade shape itself is defined by pairs of coordinates in the S, n plane - KJS on the suction side and KJP on the pressure side. In the version shown here, these were read in as a table in subroutine SHAPE.

The blade surface coordinates are numbered from 1 to KJMX; point number 1 is the trailing edge, 2 through KJLM are on the pressure side from trailing edge to leading edge, point KJLE is the leading-edge point (ZLE), KJLP through KJMX are on the suction side from leading edge to trailing edge, and point KJMX repeats the trailing edge.

The complex sine functions are calculated next; then the iterations to determine the parameter C are done. The initial guess provided for C is $ZC = 1 + i$. It may happen in some cases that a better guess is required: in particular, it is necessary that the value of C must lie outside the blade-surface curve in the Ω -plane. If it does not, then the interior of this curve in the Ω -plane is mapped to the exterior of the blade-surface image in the ω -plane. This fact can be seen from the discussion by Kober (Ref. 16) of the bilinear transformation applied to circles.

-
16. Kober, H., Dictionary of Conformal Transformations, Dover Publications, New York (1957).

After the parameter C has been found, the transformation to the ζ plane is carried out, using the fast Fourier transform procedure (see Appendix A). The actual calculations of the Fourier coefficients are done in subroutine FFT2, a proprietary program of International Mathematical and Statistical Libraries, Inc. (IMSL). This routine computes the fast Fourier transform of a complex vector of length equal to a power of two (here 2^6). The coefficients of the input vector are given in normal order by the array named as the first argument of the call; the coefficients of the output vector are overstored in this array, in reverse binary order. The subroutine SHUFL is then used to restore this output to the normal order. The coefficients in the series expression for ζ are determined iteratively in a relaxation process that is terminated when the maximum change in θ falls below the tolerance ANGERR, or when IMX iterations are done.

In doing these iterations, it is necessary to know values of $\ln r$ at given values of θ ; these are found by a spline fit in subroutine CISPLN, which is a straightforward implementation of the formulas given by Ahlberg, et al.¹⁷

In certain cases (typically when R_{MAX}/R_{MIN} is large) the calculated variation of θ with ϕ may be non-monotonic; if this occurs, the calculation should be repeated, with a smaller relaxation factor. The progress of the ϕ/θ iterations is printed, showing at each iteration the largest change in θ and the number of reversals (i.e., the number of occurrences of non-monotonic variation).

Next, the parameters ζ_A and ζ_B are found, starting with "best-guess" values calculated in the sectors described in Section 2. Once these are found, the mapping to the η -plane follows. The calculations that link the Ω -plane and the ζ -plane are done in subroutine OMETA, which sums the Theodorsen-Garrick series, using complex arithmetic. This subroutine has been modified slightly from that appearing in Ref. 1, as follows: the previous code evaluated sums of the form

17. Ahlberg, J.H., Nilson, E.N., and Walsh, J.L., The Theory of Splines and Their Applications Academic Press, New York (1967).

$$ZSUM = \sum_{JP=1}^{65} ZCC(JP) \zeta^{JP-1} \quad (4-1)$$

by a sequence of multiplications and additions:

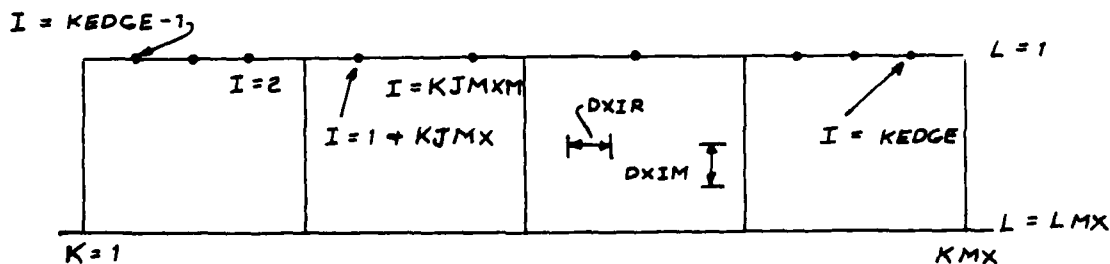
$$ZSUM = ZCC(65) \zeta^{64} + ZCC(64) \zeta^{63} + \dots + ZCC(1) \quad (4-2)$$

The current version uses (Ref. 18, p. 28)

$$ZSUM = \langle \{ [ZCC(65)\zeta + ZCC(64)] \zeta + ZCC(63) \} \zeta + \dots \quad (4-3)$$

Finally, the blade-surface image is mapped into the $\hat{\xi}$ plane, using the elliptic-function formulas of Nielsen and Perkins¹⁰ and of Luke,¹² as outlined in Section 2. This completes the mapping of the blade surface.

It is now possible to set up a grid in the $\hat{\xi}$ - plane, and map it back into the physical plane. This involves straightforward evaluations of the transformation functions. The only new complications are the need to evaluate the Jacobian elliptic sine (done in subroutine JCELFN) and to provide a good initial guess for the Newton-Raphson iteration used in finding $Z(\Omega)$. This guess is provided by starting each series of calculations on the image of the blade surface in the $\hat{\xi}$ plane, and interpolating to find the corresponding point in the S, Ω plane. The interpolation is done as follows: the blade-surface image values of $\hat{\xi}$ are stored in the array ZA1(I), where I ranges from 1 to KJMX. Each of these maps into a point in the S, Ω plane, which is stored in the array ZA2(KJ), where KJ also ranges from 1 to KJMX. Thus the $\hat{\xi}$ plane has the appearance:



18. Hartree, D.R., Numerical Analysis, 2nd Edition, Oxford, Clarendon Press (1958).

Points in the uniform grid are denoted by K, L, where

$$\begin{aligned} \operatorname{Re}(\hat{\xi}) &= -3K(\xi) + (K-1)DXIR \quad K=1, KMX \\ \operatorname{Im}(\hat{\xi}) &= \frac{K'(\xi)}{2} - (L-1)DXIM \quad L=1, LMX \end{aligned} \quad (4-4)$$

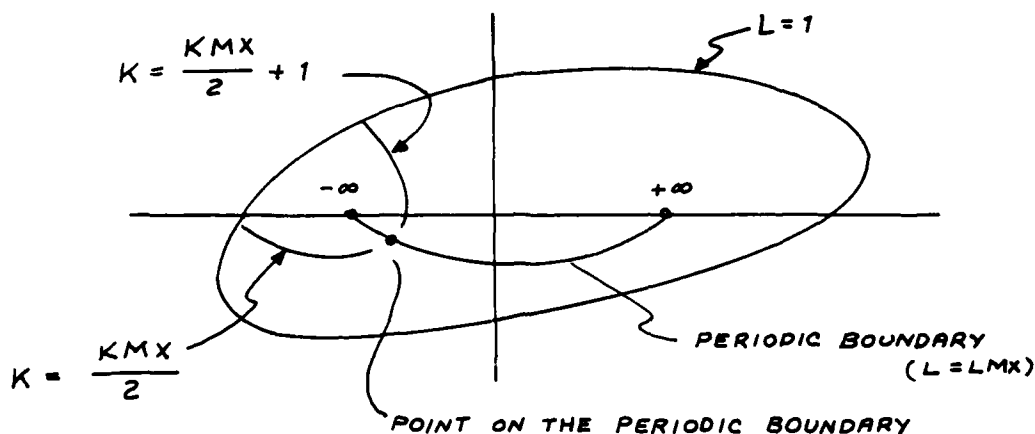
The value of I for which the real part of $ZA(I)$ is largest is denoted as KEDGE. Then, for a chosen value of K (with $L=1$) the array $ZA(I)$ is searched (first for $I = \text{KEDGE}$ to KMX , then for $I = 2$ to $\text{KEDGE}-1$) until a value, called I^* , is found such that $\operatorname{Re}[ZA(I^*)] < \operatorname{Re}(\hat{\xi})$. If none is found, it is concluded that $I^* = \text{KEDGE}$. Linear interpolation is then used, between the points I^* and $I^* - 1$, to provide the initial guess. Fifty iterations are allowed for this step, at each value of K and L. If no solution is found, the value (0,0) is printed.

The calculation of the images of the grid points in the various planes is bypassed for the points at upstream and downstream infinity, and at the trailing edge. (The trailing-edge point will be a grid point if $\text{ISHEAR}=1$). The z -plane locations of upstream and downstream infinity are arbitrarily assigned to finite locations given by linear extrapolation from the two adjacent L-values.

The point at upstream infinity will be a grid point only if KMX is odd; in this case $\text{IOE}=1$, and the image calculations are bypassed.

Adjustments to the grid-point image locations in the z -plane are sometimes required for points on the periodic boundary ($L=LMX$) for values of K near 1, $KMX/2$, and KMX . At these points, the Newton-Raphson iterations used in going from the Ω -plane to the z -plane sometimes cannot distinguish between points that are separated by $H+iG$. The problem can be seen best in the ω -plane. (The sketch below is for an even value of KMX ; the same picture applies for an odd value):

* In Ref. 1, a quadratic interpolation was used. This introduced many complications, in order to avoid conditions where the interpolation base points straddle a sharp leading or trailing edge, and where I^* is near KEDGE or $\text{KEDGE}-1$. The logic in Ref. 1 was not adequate to handle all such cases.



Note that the same point in the ω -plane (and thus also in the Ω -plane) can map into either of two points in the $\bar{z}$ -plane, which differ by $H + iG$. The selection is guided toward the correct value by starting on the blade ($L=1$) and working toward the periodic boundary ($L=LMX$), but it can happen that the wrong branch is chosen during the iterations. To avoid this, the imaginary parts of $\bar{z}$ and $\bar{z}_N$ are compared, for K values near the leading edge, and the imaginary parts of $\bar{z}$ and $\bar{z}_T$ are compared near the trailing edge, and the quantity $i \cdot SG$ is added or subtracted (depending on the value of K) where necessary.

Finally, the real and imaginary parts of $\bar{z}$ are punched on cards (if PNCHZA=TRUE). These values can be used, in a separate program, to calculate the metrics of the transformation (see Appendix D).

Section 5
CONCLUDING REMARKS

The program described above has been applied to very few cases - the thick blades used for illustrative purposes in this report, and the cases shown in Refs. 4 and 15. Because of this limited experience, the range of applicability of the program is largely unknown. On the basis of current experience, it appears that the Theodorsen-Garrick step may not converge for gap/chord ratios less than around 0.8. The set of numerical tolerances, maximum iteration counts, and relaxation factors used may need to be adjusted for certain shapes. In addition, new approximations may be required as initial guesses in various places, in order to handle such features as leading edges with very small radii of curvature.

Appendix A
DETAILS OF THE FAST FOURIER TRANSFORM PROCEDURES

Equations 2-20 and 2-21 contain $2N+2$ constants, which are evaluated as follows: first, they are satisfied at a discrete number of points, denoted by ϕ_K :

$$\phi_K = \frac{2\pi(K-1)}{2N}, \quad K = 1, 2, \dots, 2N \quad (A-1)$$

where N was chosen to be 64, in the present case. Thus:

$$\ln r_K = A_0 + \sum_{j=1}^{N-1} (A_j \cos j \phi_K - B_j \sin j \phi_K) + (-1)^K A_N \quad (A-2)$$

$$\theta_K - \phi_K = B_0 + \sum_{j=1}^{N-1} (B_j \cos j \phi_K + A_j \sin j \phi_K) + (-1)^K B_N \quad (A-3)$$

Each right-hand side now contains $2N$ coefficients. The correspondence between either of these and the Fast Fourier Transform (FFT) as presented in Ref. 6 is given by the next two equations: consider the expression

$$Y(j) = \sum_{n=0}^{2N-1} C(n) W_{2N}^{nj} ; \quad j = 0, 1, 2, \dots, 2N-1 ; W_{2N} = e^{i \frac{2\pi}{2N}} \quad (A-4)$$

where values $Y(\cdot)$ are real, and the $2N$ values of $C(\cdot)$ are in general complex, but must satisfy the following redundancy condition, in order that the $Y(\cdot)$ values be real:

$$C(n) = \tilde{C}(2N-n), \quad n = 1, 2, \dots, N-1 \quad (A-5)$$

$C(0)$ and $C(N)$ pure real

where the tilde denotes the complex conjugate. When these conditions are met, Eq. A-4 can be written as

$$Y(l) = C_R(0) + (-1)^l C_R(N) + 2 \sum_{n=1}^{N-1} \left\{ C_R(n) \cos \frac{n l \pi}{N} - C_I(n) \sin \frac{n l \pi}{N} \right\} \\ l = 0, 1, \dots, 2N-1 \quad (A-6)$$

This form can now be used, in conjunction with Eq. 2-20 or 2-21, to facilitate application of the FFT to the complex form given in Eq. A-4.

In the case of Eq. A-2, values of the coefficients A_0 through A_N and B_1 through B_{N-1} are found, from given values of $\ln r$. This is procedure 4 of Ref. 6, which takes the following steps:

$$\left. \begin{aligned} 1. \text{ Set } X_1(k) &= Y(2k) = (\ln r)_{2k} \\ X_2(k) &= Y(2k+1) = (\ln r)_{2k+1} \end{aligned} \right\} k = 0, 1, \dots, N-1 \quad (\text{A-7})$$

$$2. \text{ Set } X(j) = X_1(j) + i X_2(j), \quad j = 0, 1, \dots, N-1 \quad (\text{A-8})$$

3. Calculate the N -point Discrete Fourier Transform of

$$A(n) = A_1(n) + i A_2(n) = \frac{1}{N} \sum_{j=0}^{N-1} X(j) W_N^{-nj}; \quad n = 0, 1, \dots, N-1 \quad (\text{A-9})$$

$$W_N = e^{i \frac{2\pi}{N}}$$

4. By periodicity, set $A(N) = A(0)$

5. Apply Eq. 34 of Ref. 6, in order to extract $A_1(n)$ and $A_2(n)$ from $A(n)$:

$$\left. \begin{aligned} A_1(n) &= \frac{1}{2} \left\{ \tilde{A}(N-n) + A(n) \right\} \\ A_2(n) &= \frac{i}{2} \left\{ \tilde{A}(N-1) - A(n) \right\} \end{aligned} \right\} n = 0, 1, \dots, \frac{N}{2} \quad (\text{A-10})$$

Note that these expressions use $A(n)$ for $n = 0, 1, \dots, N$ to give $A_1(n)$ and $A_2(n)$ for $n = 0, 1, \dots, N/2$

6. These values of $A_1(n)$ and $A_2(n)$ then give $C(n)$ for the same range:

$$C(n) = \frac{1}{2} \left[A_1(n) + W_{2N}^{-n} A_2(n) \right], \quad n = 0, 1, \dots, N/2 \quad (\text{A-11})$$

For the range of n from $\frac{N}{2} + 1$ to $N-1$, use Eq. 36 of Ref. 6, with n replaced by $N-n$ (and noting that $W_{2N}^{-N} = -1$):

$$\begin{aligned} C(n) &= \tilde{C}(2N-n) \\ &= \frac{1}{2} \left\{ A_1(N-n) + W_{2N}^n A_2(N-n) \right\} \quad n = \frac{N}{2} + 1, \frac{N}{2} + 2, \dots, N-1 \end{aligned} \quad (\text{A-12})$$

This equation, applied for $n = \frac{N}{2}+1, \frac{N}{2}+2, \dots, N-1$ uses A_1 and A_2 with index $\frac{N}{2}-1, \frac{N}{2}-2, \dots, 1$ to get $C(n)$ for $n = \frac{N}{2}+1, \frac{N}{2}+2, \dots, N-1$. The process is completed by setting

$$C(N) = \frac{1}{2} \{A_1(0) - A_2(0)\} \quad (A-13)$$

7. The A_j and B_j coefficients are retrieved from:

$$\begin{aligned} A_0 &= R_e [C(0)] , \quad A_N = R_e [C(N)] \\ A_j &= 2 R_e [C(j)] , \quad B_j = 2 \text{Im} [C(j)] , \quad j = 1, 2, \dots, N-1 \end{aligned} \quad (A-14)$$

At this point, B_0 and B_N are undetermined. Following Ives, B_N is set equal to zero, and B_0 is chosen so as to place the trailing edge at $\phi=0$ (this latter selection of B_0 is actually carried out in a subsequent step, noted below).

In the case of Eq. A-3, the A 's and B 's are considered known, and are used to evaluate ϕ_k . The coefficient B_0 can be found from

$$\phi_{TE} = B_0 + \sum_{j=1}^{N-1} B_j \quad (B_N = 0) \quad (A-15)$$

Actually, it is simpler to evaluate the right-hand side of

$$\phi_k - \phi_k - B_0 = \sum_{j=1}^{N-1} (B_j \cos j \phi_k + A_j \sin j \phi_k), \quad k = 0, 1, \dots, N-1 \quad (A-16)$$

and then find B_0 from

$$B_0 = (\phi_k - \text{RHS})_{k=0} \quad (A-17)$$

The actual evaluation of the right-hand side takes the following steps (Procedure 5 of Ref. 6): by comparison of Eqs. A-3 and A-6:

1. Set $C(0) = 0, \quad C(N) = 0$

$$C(n) = \frac{1}{2} [B_n - i A_n], \quad n = 1, 2, \dots, N-1 \quad (A-18)$$

Note that the C 's determined here are different from those used in Procedure 4; the A_j and B_j values are the same, but their relation to C_j is different.

2. Values of C_j are then used to find $A_1(n)$ and $A_2(n)$: Equations 40 and 41 of Ref. 6 are rewritten, using

$$C(n) = \tilde{C}(2N-n)$$

Replace n by $N+n$:

$$C(N+n) = \tilde{C}(2N-N-n) = \tilde{C}(N-n)$$

Thus Eqs. 40 and 41 of Ref. 6 are

$$\left. \begin{aligned} A_1(n) &= C(n) + \tilde{C}(N-n) \\ A_2(n) &= [C(n) - \tilde{C}(N-n)] W_{2N}^n \end{aligned} \right\} n = 0, 1, \dots, N/2 \quad (\text{A-19})$$

These are all the values needed for A_1 and A_2 .

3. Find $A(n)$, $n=0, 1, \dots, N-1$ from Eqs. 42 and 43 of Ref. 6:

$$\left. \begin{aligned} A(n) &= A_1(n) + i A_2(n) \\ A(N-n) &= \tilde{A}_1(n) + i \tilde{A}_2(n) \end{aligned} \right\} n = 0, 1, \dots, N/2 \quad (\text{A-20})$$

This gives, on the left-hand side, all values from $n=0$ to $n=N$.

4. Calculate

$$X(j) = \sum_{n=0}^{N-1} A(n) W_N^{nj}, \quad j = 0, 1, \dots, N-1 \quad (\text{A-21})$$

5. Finally:

$$\left. \begin{aligned} \phi_{2k} - \phi_{2k} - B_0 &= \text{Re} [X(k)] \\ \phi_{2k+1} - \phi_{2k+1} - B_0 &= \text{Im} [X(k)] \end{aligned} \right\} k = 0, 1, \dots, N-1 \quad (\text{A-22})$$

The first of these equations, with $k=0$, is used to find B_0 .

In order to apply these formulas, it is necessary to have a relation

between r and θ :

$$r_K = r(\theta_K) , \quad K = 1, 2, \dots, 2N \quad (\text{A-23})$$

which is found from a spline fit to the blade-surface image in the ω - plane.

The discrete Fourier transform and its inverse are given by

$$A(n) = \frac{1}{N} \sum_{j=0}^{N-1} X(j) W_N^{-nj} , \quad n = 0, 1, 2, \dots, N-1 \quad (\text{A-24})$$

$$X(j) = \sum_{n=0}^{N-1} A(n) W_N^{nj} , \quad j = 0, 1, 2, \dots, N-1 \quad (\text{A-25})$$

The IMSL routine FFT2 evaluates the second of these, i.e., it returns

$X(j)$, given the values $A(n)$. To evaluate the first of these, FFT2 is used with input $\tilde{X}(j)/N$, and with output interpreted to be $\tilde{A}(n)$; this is Procedure 1 of Reference 6:

$$N \tilde{A}(n) = \sum_{j=0}^{N-1} \tilde{X}(j) W_N^{nj} \quad (\text{A-26})$$

In the FORTRAN version of these and other procedures, it is convenient to use indices that begin at one, rather than zero, by setting $j+1 = j'$ (FORTRAN symbol JP). The corresponding table, for example of the coefficient B_j , is

j	JP	B _j	B(JP)
0	1	B ₀	B(1)
1	2	B ₁	B(2)
2	3	B ₂	B(3)
N-1	N	B _{N-1}	B(N)
N	N+1	B _N	B(N+1)

In addition, care must be taken with the argument $N-n$; for example, Eqs. are written as

$$\left. \begin{aligned} Z A 1 (NP) &= Z C C (NP) + \widetilde{Z C C} (N+2 - NP) \\ Z A 2 (NP) &= [Z C C (NP) - \widetilde{Z C C} (N+2 - NP)] W_{2N}^n \end{aligned} \right\} \begin{aligned} NP &= 1, 2, \dots, \frac{N}{2} + 1 \\ (A-27) \end{aligned}$$

It can be verified that the quantity $N + 2 - NP$ in the arguments above preserves the correct ordering - for example when $N = 64$, and $n = 0$, $\widetilde{C}(N-n) = \widetilde{C}(64)$. This would be stored in $ZCC(64 + 2 - 1) = ZCC(65)$.

Appendix B COMPUTER PROGRAM LISTING

LEVEL 21.7 (DEC 72)

OS/360 FORTRAN H

COMPILER OPTIONS - NAME= MAIN,OPT=02,LINECNT=60,SIZE=0000K,
SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,ID,XREF

```

C
C PROGRAM IVLTMP - THE IVES - LIUTERMOZA CONFORMAL TRANSFORMATION FOR
C TURBOMACHINERY CASCADES (AIAA JOURNAL,VOL. 5,1977, PP 647 - 652)
C DOCUMENTATION IS GIVEN IN:
C W. J. RAE, A COMPUTER PROGRAM FOR THE IVES TRANSFORMATION
C IN TURBOMACHINERY CASCADES, CALSPAN CORPORATION REPORT 6275-A-3,
C NOVEMBER 1980
C W.J. RAE, MODIFICATIONS OF THE IVES - LIUTERMOZA CONFORMAL-
C MAPPING PROCEDURE FOR TURBOMACHINERY CASCADES, ASME PAPER
C 83-GT-116, MARCH 1983
C W.J. RAE, REVISED COMPUTER PROGRAM FOR EVALUATING THE IVES
C TRANSFORMATION IN TURBOMACHINERY CASCADES, CALSPAN CORPORATION
C REPORT 7177-A-1, JULY 1983
C
ISN 0002      IMPLICIT REAL*8(A-H,O-Y),COMPLEX*16(Z)
ISN 0003      LOGICAL PNCHZA
ISN 0004      COMMON/TGINTG/N,NP1,NP2,N2,NB2,NB2P1,IP,IPMX,ITP,IWK,IMX,KJMX
ISN 0005      COMMON/TGCMPI/Z1,ZW2N,Z1,ZNN,ZA,ZCC,ZA1,ZA2
ISN 0006      COMMON/TGDBLE/PBN,OM,OMM,ANGERR,A,B,E,F,THT,PHI,X,Y
ISN 0007      DIMENSION ZS(80),ZP(80),ZOMS(80),ZOMP(80)
ISN 0008      DIMENSION RDS(80),RDP(80),THS(80),THP(80)
ISN 0009      DIMENSION RDSX(80),RDPX(80),THSX(80),THPX(80)
ISN 0010      DIMENSION X(160),Y(160),E(150),F(150),THT(160)
ISN 0011      DIMENSION PHI(130),A(65),B(65),ZCC(65),
*             ZA(165),ZA1(165),ZA2(165),
*             IWK(7)
ISN 0012      DIMENSION ITP(100),ID(36)
ISN 0013      DIMENSION XX(50,20),YY(50,20)
ISN 0014      DO 10 I = 1,7
ISN 0015      10 IWK(I) = 0
C
ISN 0016      C
*             NAMELIST/INPUTS/ ANGERR,EX,G,H,IGOT,ILE,ITE,KMX,
*             IMX, LMX, OM, PNCHZA, RTOL, ZC, ZLE, ZN, ZT, ZTE,
*             IPMX,ISHEAR,INR,KJS,KJP,KNR
C
C --- SET NAMELIST DEFAULT VALUES
C
ISN 0017      ANGERR=0.0100
ISN 0018      IGOT=0
ISN 0019      ILE = 1
ISN 0020      ITE=0
ISN 0021      KMX = 40
ISN 0022      LMX = 10
ISN 0023      KJS = 20
ISN 0024      KJP = 20
ISN 0025      IMX = 400
ISN 0026      ISHEAR = 1
ISN 0027      IPMX = 1
ISN 0028      KNR = 0
ISN 0029      INR = 0
ISN 0030      OM=0.100
ISN 0031      PNCHZA=.FALSE.
ISN 0032      RTOL=3.000
C
C NOTE: EX, G, H, ZC, ZLE, ZN, ZT, ZTE HAVE NO DEFAULT VALUES
C (ZC IS NOT NEEDED IF IGOT=0)

```

```

C
ISN 0033      READ(5,103)(ID(I),I=1,36)
ISN 0034      103 FORMAT(18A4)
C
ISN 0035      PI = 4.0D0*DATAN(1.0D0)
ISN 0036      TPI = 2.0D0*PI
ISN 0037      ZPI = DCMPLX(PI,0.0D0)
ISN 0038      Z1 = DCMPLX(1.0D0,0.0D0)
ISN 0039      ZERO = DCMPLX(0.0D0,0.0D0)
ISN 0040      ZMGA = DCMPLX(+1.0D0,0.0D0)
ISN 0041      ZMGB = DCMPLX(-1.0D0,0.0D0)
C *** READ PNCHZA=T IF (ZA(L),L=1,LMX) IS TO BE PUNCHED FOR ALL VALUES
C *** OF K, OTHERWISE READ PNCHZA=F
C IGOT = 1 IF THE VALUE OF ZC IS KNOWN. OTHERWISE, IGOT = 0.
C KMX AND LMX ARE THE GRID SIZES IN THE FINAL TRANSFORMED PLANE.
C KJS AND KJP ARE THE NUMBERS OF POINTS ON THE SUCTION AND PRESSURE
C SIDES AT WHICH PAIRS OF BLADE COORDINATES WILL BE INPUT.
C ILE = 0 OR 1 FOR A SHARP OR ROUNDED LEADING EDGE, RESPECTIVELY
C ITE = 0 OR 1 FOR A SHARP OR ROUNDED TRAILING EDGE, RESPECTIVELY.
C IMX IS THE MAXIMUM NUMBER OF ITERATIONS ALLOWED FOR THE PHI/THETA
C ITERATIONS
C INR AND KNR.NE.0 WILL CAUSE A DIAGNOSTIC OUTPUT FOR THE FIRST INR
C NEWTON-RAPHSON ITERATIONS (TO FIND Z, GIVEN ZBOMK), AT THE STATION
C K = KNR. THE PROGRAM WILL THEN STOP.
C IPMX.NE.1 CAN BE USED TO DISPLAY THE VALUES OF THETA DURING THE
C PHI/THETA ITERATIONS, AT THE ITERATION NUMBERS READ INTO THE
C ITP ARRAY BELOW.
C ISHEAR = 0 GIVES AN ORTHOGONAL GRID. THE LINES K=1 AND K=KMX, WHICH
C START AT THE TRAILING EDGE, DO NOT GO TO DOWNSTREAM INFINITY.
C ISHEAR = 1 SHEARS THE GRID UNIFORMLY; IN THIS CASE, THE K=1 AND KMX
C LINES DO GO TO DOWNSTREAM INFINITY.
C OM IS A RELAXATION FACTOR USED IN THE PHI/THETA MAPPING. USE 0.1,
C OR A SMALLER VALUE IF THE A AND B ITERATIONS FAIL TO CONVERGE.
C ANGERR IS THE ANGULAR TOLERANCE (IN RADIAN) FOR THE PHI/THETA
C TRANSFORMATION. A REASONABLE VALUE IS 0.01
C RTOL IS THE TOLERANCE FOR THE MAX/MIN RADIUS RATIO IN THE
C OMEGA PLANE
C
C --- FOR A SHARP LEADING EDGE (ILE=0) ZN MUST EQUAL ZLE
C --- FOR A SHARP TRAILING EDGE (ITE=0) ZT MUST EQUAL ZTE
C
C --- READ NAMELIST INPUT DATA
C
ISN 0042      READ(5,INPUTS)
ISN 0043      ITP(1) = 999
ISN 0044      IF(IPMX.NE.1) READ(5,102)(ITP(IP),IP=1,IPMX)
ISN 0045      102 FORMAT(20I4)
ISN 0046      IP = 1
C
C
ISN 0048      KMXH=(KMX+1)/2
ISN 0049      KMXL=KMXH
ISN 0050      IF(MOD(KMX,2).NE.0) KMXL=KMXH-1
ISN 0051      IOE = MOD(KMX,2)
ISN 0052      KAA = KMXH - 2 - IOE
ISN 0053      KBB = KMXH + 3
ISN 0054      KMXM4 = KMX - 4
ISN 0055      KOUNT = 0
ISN 0056

```

```

C
C
C INPUT VARIABLES EX, G, H, ZLE, ZN, ZT, ZTE,
C THE FORTRAN STATEMENTS IN SUBROUTINE SHAPE,
C AND THE VARIABLES LISTED IN COMMON BLOCK GEOM (IF ONE IS BEING USED)
C ARE ALL SPECIFIC TO THE BLADE SHAPE BEING USED.
C
C
C CALCULATION OF THE BLADE SHAPE
C
ISN 0057      D = CDABS(ZTE-ZLE)
ISN 0058      CALL SHAPE(D,H,G,EX,ZP,ZS,KJS,KJP)
ISN 0059      SG = DSQRT(H*H+G*G)
ISN 0060      SIZE = 5.000*SG
ISN 0061      ZDA = DCMPLX(G/SG,-H/SG)
ISN 0062      ZGAMMA=DCONJG(ZDA)
ISN 0063      WRITE(6,207)
ISN 0064 207 FORMAT(1H1)
ISN 0065      WRITE(6,206)(ID(I),I=1,36)
ISN 0066 206 FORMAT(30X,18A4)
ISN 0067      WRITE(6,240) D,H,G,EX,SG
ISN 0063 240 FORMAT(/10X,'BLADE-GEOMETRY PARAMETERS ARE:',
      * //5X,'ABS(ZTE-ZLE) = ',
      *      F10.5,' H = ',F10.5,' G = ',F10.5,' EX = ',
      *      F10.5,' SLANT GAP = ',F10.5,/)
ISN 0069      WRITE(6,INPUTS)
ISN 0070      KJLE = KJP + 2
ISN 0071      KJMX = KJLE + KJS + 1
ISN 0072      WRITE(6,209)
ISN 0073 209 FORMAT( 8X,'BLADE COORDINATES:',
      * //9X,'SUCTION SIDE',/3X,'KJ',7X,'S',13X,'N',13X,'X',13X,'Y',/)
ISN 0074      ZXY = ZGAMMA*ZLE
ISN 0075      WRITE(6,270) KJLE,ZLE,ZXY
ISN 0076 270 FORMAT(15,1P4E14.5)
ISN 0077      DO 64 K = 1,KJS
ISN 0078      KJ = KJLE + K
ISN 0079      ZXY = ZGAMMA*ZS(K)
ISN 0080      WRITE(6,270) KJ,ZS(K),ZXY
ISN 0081 64 CONTINUE
ISN 0082      ZXY = ZGAMMA*ZTE
ISN 0083      WRITE(6,270) KJMX,ZTE ,ZXY
ISN 0084      WRITE(6,271)
ISN 0085 271 FORMAT(/8X,'PRESSURE SIDE',/3X,'KJ',7X,'S',13X,'N',
      *      13X,'X',13X,'Y',/)
ISN 0086      ZXY = ZGAMMA*ZLE
ISN 0087      WRITE(6,270) KJLE,ZLE,ZXY
ISN 0088      DO 61 K = 1,KJP
ISN 0089      KJ = KJLE - K
ISN 0090      ZXY = ZGAMMA*ZP(K)
ISN 0091      WRITE(6,270) KJ,ZP(K),ZXY
ISN 0092 61 CONTINUE
ISN 0093      KJ = 1
ISN 0094      ZXY = ZGAMMA*ZTE
ISN 0095      WRITE(6,270) KJ,ZTE,ZXY
C
ISN 0096      ZDN = DCMPLX(H,G)
ISN 0097      ZZ = (ZT-ZN)/ZDN
ISN 0096      ZTN = ZPI*ZZ

```

```

ISN 0099      CHI = PI*EX*(G*DREAL(ZT-ZN)-H*DIMAG(ZT-ZN))/(SG*SG)
ISN 0100      XA = PI*EX*(H*DREAL(ZT-ZN)+G*DIMAG(ZT-ZN))/(SG*SG)
ISN 0101      R = DEXP(-CHI)
ISN 0102      ZPLUS = DCMPLX(R*DCOS(XA),-R*DSIN(XA))
ISN 0103      R = 1.000/R
ISN 0104      ZMINUS = DCMPLX(R*DCOS(XA),R*DSIN(XA))
ISN 0105      DO 20 K = 1,KJS
ISN 0106      ZZT = (ZS(K)-ZT)/ZDN
ISN 0107      ZETA1S = ZPI*ZZT
ISN 0108      ZETA2S = CDSIN(ZETA1S)
ISN 0109      ZZN = (ZS(K)-ZN)/ZDN
ISN 0110      ZETA3S = ZPI*ZZN
ISN 0111      ZETA4S = CDSIN(ZETA3S)
ISN 0112      ZFS = ZETA2S/ZETA4S
ISN 0113      RDS(K) = CDABS(ZFS)
ISN 0114      THS(K) = DATAN2(DIMAG(ZFS),DREAL(ZFS))
ISN 0115      20 CONTINUE
ISN 0116      DO 24 K = 1,KJP
ISN 0117      ZZT = (ZP(K)-ZT)/ZDN
ISN 0118      ZETA1P = ZPI*ZZT
ISN 0119      ZETA2P = CDSIN(ZETA1P)
ISN 0120      ZZN = (ZP(K)-ZN)/ZDN
ISN 0121      ZETA3P = ZPI*ZZN
ISN 0122      ZETA4P = CDSIN(ZETA3P)
ISN 0123      ZFP = ZETA2P/ZETA4P
ISN 0124      RDP(K) = CDABS(ZFP)
ISN 0125      THP(K) = DATAN2(DIMAG(ZFP),DREAL(ZFP))
ISN 0126      24 CONTINUE
C NOW ADD THE LEADING- AND TRAILING-EDGE POINTS, AND STORE THE G(Z)
C ARRAY AS E(KJ)*EXP(I*THT(KJ)), WHERE KJ=1,KJMX AS YOU GO FROM TE AROUND
C THE PRESSURE SIDE TO THE LE (KJ=KJLE) AND THEN ALONG THE SUCTION SIDE
C BACK TO THE TE AGAIN (KJ=KJMX).
ISN 0127      IF(ITE.EQ.1) GO TO 13
ISN 0128      E(1)=0.000
ISN 0129      THT(1)=0.000
ISN 0130      GO TO 14
ISN 0131      13 ZETA2=CDSIN(ZPI*(ZTE-ZT)/ZDN)
ISN 0132      ZETA4=CDSIN(ZPI*(ZTE-ZN)/ZDN)
ISN 0133      ZFP=ZETA2/ZETA4
ISN 0134      E(1)=CDABS(ZFP)
ISN 0135      THT(1)=DATAN2(DIMAG(ZFP),DREAL(ZFP))
ISN 0136      14 IF(ILE.EQ.1) GO TO 15
ISN 0137      E(KJLE) = 0.000
ISN 0138      THT(KJLE) = 0.000
ISN 0139      GO TO 16
ISN 0140      15 ZETA2=CDSIN(ZPI*(ZLE-ZT)/ZDN)
ISN 0141      ZETA4=CDSIN(ZPI*(ZLE-ZN)/ZDN)
ISN 0142      ZFP=ZETA2/ZETA4
ISN 0143      E(KJLE) = CDABS(ZFP)
ISN 0144      THT(KJLE) = DATAN2(DIMAG(ZFP),DREAL(ZFP))
ISN 0145      16 DO 17 K = 1,KJP
ISN 0146      KJ = KJLE + K
ISN 0147      E(KJ)=RDP(K)
ISN 0148      THT(KJ)=THP(K)
ISN 0149      17 CONTINUE
ISN 0150      DO 18 K = 1,KJS
ISN 0151      KJ = KJLE + K
ISN 0152      E(KJ)=RDS(K)
ISN 0153
ISN 0154

```

```

ISN 0155      13 THT(KJ)=THS(K)
ISN 0156      E(KJMX) = E(1)
ISN 0157      THT(KJMX) = THT(1)

C
C NOW ADJUST THE BRANCHES OF G(Z) SO AS TO BE CONTINUOUS ACROSS
C THE CUT (ALONG THE NEGATIVE REAL AXIS) OF THE DATAN2 FUNCTION.
C
ISN 0158      IF(H.LT.0.0D0) GO TO 251
ISN 0160      IF(ITE.EQ.1) GO TO 251

C
C FOR COMPRESSORS WITH A SHARP T. E. , THE CONVENTION IS TO FORCE
C ARG(THT(KJ=2)) TO BE NEGATIVE:
C
ISN 0162      BR = 0.0D0
ISN 0163      KA = 3
ISN 0164      PO = THT(2)
ISN 0165      IF(PO.LT.0.0D0) GO TO 252
ISN 0167      BR = -1.0D0
ISN 0168      THT(2) = THT(2) - TPI
ISN 0169      GO TO 252

C
C CONVENTION FOR ALL TURBINES, AND FOR COMPRESSORS
C WITH A ROUND T. E. , IS:
C
ISN 0170      251 BR=0.0D0
ISN 0171      KA=3
ISN 0172      IF(ITE.EQ.1) KA=2
ISN 0174      PO=THT(KA-1)
ISN 0175      252 DO 321 KJ=KA,KJMX
ISN 0176      CHG=THT(KJ)-PO
ISN 0177      PO=THT(KJ)
ISN 0178      IF(DABS(CHG).LE.PI) GO TO 321
ISN 0180      IF(CHG.GT.PI) BR=BR-1.0D0
ISN 0182      IF(CHG.LT.-PI) BR=BR+1.0D0
ISN 0184      321 THT(KJ)=THT(KJ)+BR*TPI
ISN 0185      DO 323 K=1,KJP
ISN 0186      THP(K) = THT(KJLE-K)
ISN 0187      323 CONTINUE
ISN 0188      DO 322 K = 1,KJS
ISN 0189      THS(K) = THT(KJLE+K)
ISN 0190      322 CONTINUE

C
ISN 0191      DO 25 K = 1,KJS
ISN 0192      ARS = EX*THS(K)
ISN 0193      RS = RDS(K)**EX
ISN 0194      RDSX(K) = RS
ISN 0195      THSX(K) = ARS
ISN 0196      25 CONTINUE
ISN 0197      DO 26 K = 1,KJP
ISN 0198      ARP = EX*THP(K)
ISN 0199      RP = RDP(K)**EX
ISN 0200      RDPX(K) = RP
ISN 0201      THPX(K) = ARP
ISN 0202      26 CONTINUE
ISN 0203      WRITE(6,241)
ISN 0204      241 FORMAT(//10X,'BLADE-SURFACE IMAGES IN THE G - PLANE (RATIO OF',
* ' SINES) AND CAP OMEGA PLANE(G**1/KAPPA) ,AND',
* '// 9X,' RADII AND ANGLES USED IN SELECTING THE PROPER',

```

```

* ' BRANCHES OF THE RATIO OF SINE FUNCTIONS ARE: ',
* //3X,'KJ',15X,'G',25X,'CAP OMEGA',13X,'R',11X,'THETA',
* 9X,'R**EX',7X,'THETA*EX' )
ISN 0205 XS=E(1)*DCOS(THT(1))
ISN 0206 VS=E(1)*DSIN(THT(1))
ISN 0207 RDTX = E(1)**EX
ISN 0208 THTX = EX*THT(1)
ISN 0209 US = RDTX*DCOS(THTX)
ISN 0210 VS = RDTX*DSIN(THTX)
ISN 0211 WRITE(6,325) XS,YS,US,VS,E(1),THT(1),RDTX,THTX
ISN 0212 232 FORMAT(I5,1P8E14.5)
ISN 0213 324 FORMAT(' LE ',1P8E14.5)
ISN 0214 325 FORMAT(' TE ',1P8E14.5)
ISN 0215 KJLM = KJLE - 1
ISN 0216 DO 65 KJ = 2,KJLM
ISN 0217 K = KJLE - KJ
ISN 0218 XP = RDP(K)*DCOS(THP(K))
ISN 0219 YP = RDP(K)*DSIN(THP(K))
ISN 0220 UP = RDPX(K)*DCOS(THPX(K))
ISN 0221 VP = RDPX(K)*DSIN(THPX(K))
ISN 0222 WRITE(6,232)KJ,XP,YP,UP,VP,RDP(K),THP(K),RDPX(K),THPX(K)
ISN 0223 65 CONTINUE
ISN 0224 XS = E(KJLE)*DCOS(THT(KJLE))
ISN 0225 YS = E(KJLE)*DSIN(THT(KJLE))
ISN 0226 RDLX = E(KJLE)**EX
ISN 0227 THLX = EX*THT(KJLE)
ISN 0228 US = RDLX*DCOS(THLX)
ISN 0229 VS = RDLX*DSIN(THLX)
ISN 0230 WRITE(6,324)XS,YS,US,VS,E(KJLE),THT(KJLE),RDLX,THLX
ISN 0231 KJMXM = KJMX - 1
ISN 0232 KJLP = KJLE + 1
ISN 0233 DO 31 KJ = KJLP,KJMXM
ISN 0234 K = KJ - KJLE
ISN 0235 XS = RDS(K)*DCOS(THS(K))
ISN 0236 YS = RDS(K)*DSIN(THS(K))
ISN 0237 US = RDSX(K)*DCOS(THSX(K))
ISN 0238 VS = RDSX(K)*DSIN(THSX(K))
ISN 0239 WRITE(6,232) KJ,XS,YS,US,VS,RDS(K),THS(K),RDSX(K),THSX(K)
ISN 0240 31 CONTINUE
ISN 0241 XP = E(KJMX)*DCOS(THT(KJMX))
ISN 0242 YP = E(KJMX)*DSIN(THT(KJMX))
ISN 0243 RDTX = E(KJMX)**EX
ISN 0244 THTX = EX*THT(KJMX)
ISN 0245 UP = RDTX*DCOS(THTX)
ISN 0246 VP = RDTX*DSIN(THTX)
ISN 0247 WRITE(6,325)XP,YP,UP,VP,E(KJMX),THT(KJMX),RDTX,THTX
ISN 0248 WRITE(6,242)ZPLUS,ZMINUS
ISN 0249 242 FORMAT('//10X,'POINTS AT INFINITY ARE LOCATED IN THE CAP OMEGA',
* ' PLANE AT:',
* //15X,'PLUS:',1P2E15.4,' MINUS:',2E15.4,/)

C
C DETERMINATION OF ZC SUCH AS TO MINIMIZE THE RATIO RMAX/RMIN IN THE
C L.C. OMEGA PLANE

ISN 0250 M=1
ISN 0251 ZE = (ZMGA-ZMGB)/(ZPLUS-ZMINUS)
ISN 0252 ZF = (ZMGA*ZMINUS-ZMGB*ZPLUS)/(ZPLUS-ZMINUS)
ISN 0253 ZG = (ZMGA*ZPLUS-ZMGB*ZMINUS)/(ZPLUS-ZMINUS)

```

```

ISN 0254      WRITE(6,601)
ISN 0255      601 FORMAT('1ITER',11X,'ZD',22X,'ZB',22X,'ZC',18X,'ZOMSTR',19X,'ZNTRD'
ISN 0256      1/13X,'RMIN',8X,'RMAX',7X,'RATIO'/' (ZA(KJ),KJ=1,KJMX)')
ISN 0257      250 ITER=1
ISN 0258      RATIO=0.000
                IF(IGOT.EQ.1) GO TO 60
C
C   FOR A FIRST GUESS, USE ZC=(-1.0,+1.0)
C
ISN 0260      ZC=DCMPLX(-1.000,1.000)
C
ISN 0261      60 ZB = (ZMGA*ZPLUS-ZMGB*ZMINUS-ZC*(ZMGA-ZMGB))/(ZPLUS-ZMINUS)
ISN 0262      ZD = (ZMGB*ZPLUS-ZMGA*ZMINUS+(ZMGA-ZMGB)/ZC)/(ZPLUS-ZMINUS)
ISN 0263      68 CONTINUE
ISN 0264      DO 75 K = 1,KJS
ISN 0265      RS = RDSX(K)
ISN 0266      ARS = THSX(K)
ISN 0267      ZOMS(K) = DCMPLX(RS*DCOS(ARS),RS*DSIN(ARS))
ISN 0268      ZOMS(K) = (ZD-ZB*ZOMS(K)/ZC)/(Z1-ZOMS(K)/ZC)
ISN 0269      75 CONTINUE
ISN 0270      DO 85 K = 1,KJP
ISN 0271      RP = RDPX(K)
ISN 0272      ARP = THPX(K)
ISN 0273      ZOMP(K) = DCMPLX(RP*DCOS(ARP),RP*DSIN(ARP))
ISN 0274      ZOMP(K) = (ZD-ZB*ZOMP(K)/ZC)/(Z1-ZOMP(K)/ZC)
ISN 0275      85 CONTINUE
ISN 0276      IF(ILE.EQ.1) GO TO 11
ISN 0277      ZOMLE = ZB
ISN 0278      GO TO 12
ISN 0279      11 RD = E(KJLE)
ISN 0280      TH = THT(KJLE)
ISN 0281      RS = RD**EX
ISN 0282      TH = EX*TH
ISN 0283      ZOMLE = DCMPLX(RS*DCOS(TH),RS*DSIN(TH))
ISN 0284      ZOMLE = (ZD-ZB*ZOMLE/ZC)/(Z1-ZOMLE/ZC)
ISN 0285      12 IF(ITE.EQ.1) GO TO 32
ISN 0286      ZOMTE = ZD
ISN 0287      ZA(1) = ZOMTE
ISN 0288      GO TO 33
ISN 0289      32 RD=E(1)
ISN 0290      TH=THT(1)
ISN 0291      RS=RD**EX
ISN 0292      TH=EX*TH
ISN 0293      ZOMTE=DCMPLX(RS*DCOS(TH),RS*DSIN(TH))
ISN 0294      ZOMTE=(ZD-ZB*ZOMTE/ZC)/(Z1-ZOMTE/ZC)
ISN 0295      ZA(1)=ZOMTE
ISN 0296      33 DO 76 KJ = 2,KJLM
ISN 0297      K = KJLE - KJ
ISN 0298      76 ZA(KJ) = ZOMP(K)
ISN 0299      ZA(KJLE) = ZOMLE
ISN 0300      DO 77 K = 1,KJS
ISN 0301      KJ = KJLE + K
ISN 0302      77 ZA(KJ) = ZOMS(K)
ISN 0303      ZA(KJMX) = ZA(1)
ISN 0304      ZNTRD = DCMPLX(0.000,0.000)
ISN 0305      AREA = 0.000
ISN 0306      RMIN=CDABS(ZA(1))
ISN 0307      RMAX=RMIN
ISN 0308
ISN 0309

```

```

ISN 0310      ZMAX=ZA(1)
ISN 0311      ZMIN=ZA(1)
ISN 0312      KJMX = 1
ISN 0313      KJMN=1
ISN 0314      DO 78 KJ = 2,KJMX
ISN 0315      DAREA = DABS(DREAL(ZA(KJ-1))*DIMAG(ZA(KJ))-DREAL(ZA(KJ)))*
*      DIMAG(ZA(KJ-1)))/2.0D0
ISN 0316      ZBR = (ZA(KJ-1)+ZA(KJ))/3.0D0
ISN 0317      ZNTRD = ZNTRD + ZBR*DAREA
ISN 0318      RABS=CDABS(ZA(KJ))
ISN 0319      IF(RABS.GE.RMIN) GO TO 79
ISN 0321      RMIN=RABS
ISN 0322      ZMIN=ZA(KJ)
ISN 0323      KJMN=KJ
ISN 0324      GO TO 78
ISN 0325      79 IF(RABS.LE.RMAX) GO TO 78
ISN 0327      RMAX=RABS
ISN 0328      ZMAX=ZA(KJ)
ISN 0329      KJMX = KJ
ISN 0330      73 AREA = AREA + DAREA
ISN 0331      RATIO=RMAX/RMIN
ISN 0332      ZNTRD = ZNTRD/AREA
ISN 0333      ZOMSTR= ZC*(ZNTRD-ZD)/(ZNTRD-ZB)
ISN 0334      WRITE(6,602) ITER,ZD,ZB,ZC,ZOMSTR,ZNTRD,RMIN,RMAX,RATIO,
1      (ZA(KJ),KJ=1,KJMX)
ISN 0335      602 FORMAT(/I5,1P10E12.4/E17.4,2E12.4/(10E13.5))
ISN 0336      IF(RATIO.LT.RTOL) GO TO 63
ISN 0338      IF(IGOT.EQ.1) GO TO 63
ISN 0340      ITER = ITER + 1
ISN 0341      IF(ITER.LE.30) GO TO 62
ISN 0343      WRITE(6,204)
ISN 0344      204 FORMAT(///10X,'TOLERANCE SPECIFIED FOR RMAX/RMIN NOT MET IN',
*      ' 30 ITERATIONS')
ISN 0345      STOP
ISN 0346      62 CONTINUE
ISN 0347      IF(M.EQ.2) GO TO 66
ISN 0349      ZDS=1.1D0*ZMIN
ISN 0350      KJ=KJMN
ISN 0351      67 RD=E(KJ)**EX
ISN 0352      TH=EX*THT(KJ)
ISN 0353      ZOM=DCPLX(RD*DCOS(TH),RD*DSIN(TH))
ISN 0354      ZC=(ZOM*(ZDS-ZG)+ZE)/(ZDS+ZF-ZE*ZOM)
ISN 0355      GO TO 60
ISN 0356      66 M=1
ISN 0357      ZDS=0.9D0*ZMAX
ISN 0358      KJ=KJMX
ISN 0359      GO TO 67
ISN 0360      63 IGOT = 1
ISN 0361      WRITE(6,208) ZD,ZB,ZC,ZNTRD,ZOMSTR
ISN 0362      208 FORMAT(//10X, 'CONSTANTS FOR MAPPING FROM ',
*      'Z - PLANE TO OMEGA - PLANE ARE',
*      //20X,'A = ',1P2E20.5,/20X,'B = ',1P2E20.5,/20X,'C = ',
*      1P2E20.5,/20X,'ZNTRD = ',1P2E20.5,/20X,'ZOMSTR = ',1P2E20.5)
C
C      SET UP THE ARRAYS OF THETA AND LN(R)
C
ISN 0363      DO 41 KJ = 1,KJMX
ISN 0364      X(KJ) = DATAN2(DIMAG(ZA(KJ)),DREAL(ZA(KJ)))

```

```

ISN 0365      41 Y(KJ) = CDABS(ZA(KJ))
ISN 0366      X(KJMX) = X(1) + TPI
ISN 0367      Y(KJMX) = Y(1)
C
C NOW ADJUST THE ARGUMENTS OF THE THETA ARRAY, SO AS TO BE CONTINUOUS
C ACROSS THE BRANCH CUT (ALONG THE NEGATIVE REAL AXIS) OF THE
C DATAN2 FUNCTION. THIS ADJUSTMENT ASSUMES THAT THE CONTOUR IS
C TRAVERSED IN A COUNTERCLOCKWISE DIRECTION.
C
ISN 0368      BR = 0.000
ISN 0369      PO = X(1)
ISN 0370      DO 410 KJ = 2,KJMX
ISN 0371      IF(DABS(X(KJ)-PO).GT.PI) BR = 1.000
ISN 0373      PO = X(KJ)
ISN 0374      X(KJ) = X(KJ) + BR*TPI
ISN 0375      410 CONTINUE
C
ISN 0376      RMIN = 10.000
ISN 0377      RMAX = 0.000
ISN 0378      DO 49 K = 1,KJMX
ISN 0379      IF(Y(K).LT.RMIN) RMIN = Y(K)
ISN 0381      49 IF(Y(K).GT.RMAX) RMAX = Y(K)
ISN 0383      WARSCH = DSQRT(RMAX/RMIN) - 1.000
ISN 0384      IF(WARSCH.LT.0.300) OM = 1.000
ISN 0386      WRITE(6,202)
ISN 0387      WRITE(6,243)
ISN 0388      243 FORMAT(3X, 'BLADE-SURFACE IMAGE IN THE OMEGA PLANE:',
*      /3X, 'KJ', 6X, 'REAL', 10X, 'IMAG', 12X, 'R', 9X, 'THETA', /)
ISN 0389      DO 51 KJ = 1,KJMX
ISN 0390      WRITE(6,270) KJ,ZA(KJ),Y(KJ),X(KJ)
ISN 0391      51 CONTINUE
ISN 0392      DO 43 KJ = 1,KJMX
ISN 0393      43 Y(KJ) = DLOG(Y(KJ))
C
C USE FFT TO FIND A(N) AND B(N)
C
C FIRST SET UP THE CONSTANTS FOR THE LN(R), THETA SPLINE FIT
C
ISN 0394      CALL CISPLN(Y,X,E,F,KJMX,1,128,1)
C
ISN 0395      ZI = DCMPLX(0.000,1.000)
ISN 0396      N = 64
ISN 0397      N2 = 128
ISN 0398      NP1 = N + 1
ISN 0399      NP2 = N + 2
ISN 0400      PBN = PI/64.000
ISN 0401      NB2 = N/2
ISN 0402      NB2P1 = NB2 + 1
ISN 0403      ZNN = DCMPLX(DFLOAT(N),0.000)
ISN 0404      ZW2N = DCMPLX(DCOS(PBN),DSIN(PBN))
ISN 0405      OMM = 1.000 - OM
C
ISN 0406      CALL THDGRK
C
C FIND ZETAA AND ZETAB
C
ISN 0407      ZABST = ZMGA
ISN 0408      ZBBST = ZMGB

```

```

ISN 0409      RABST = 1.000
ISN 0410      RBBST = 1.000
ISN 0411      DR = 0.100
ISN 0412      DTH = 10.000*PI/180.000
ISN 0413      THA = -6.000*DTH
ISN 0414      DO 21 I = 1,6
ISN 0415      R = .4900 + DR*DFLOAT(I-1)
ISN 0416      DO 22 J = 1,13
ISN 0417      TH = DTH*DFLOAT(J-1) + THA
ISN 0418      ZTAGS = DCMPLX(R*DCOS(TH),R*DSIN(TH))
ISN 0419      CALL OMETA(A,B,ZMG,ZTAGS,ZTANSR,65,1.00-00,1)
ISN 0420      RA = CDABS(ZMG-ZMGA)
ISN 0421      IF(RA.GT.RABST) GO TO 23
ISN 0422      RABST = RA
ISN 0423      ZABST = ZTAGS
ISN 0424      23 ZTAGS = -ZTAGS
ISN 0425      CALL OMETA(A,B,ZMG,ZTAGS,ZTANSR,65,1.00-00,1)
ISN 0426      RB = CDABS(ZMG-ZMGB)
ISN 0427      IF(RB.GT.RBBST) GO TO 22
ISN 0428      RBBST = RB
ISN 0429      ZBBST = ZTAGS
ISN 0430      22 CONTINUE
ISN 0431      21 CONTINUE
ISN 0432      ZTAGS = ZABST
ISN 0433      M = 0
ISN 0434      CALL OMETA(A,B,ZMGA,ZTAGS,ZTANSR,65,1.00-05,M)
ISN 0435      IF(M.EQ.0) GO TO 260
ISN 0436      WRITE(6,261) ZTAGS,RABST
ISN 0437      261 FORMAT(/5X,'OMETA FAILED TO CONVERGE FOR ZETA A:',
* /10X,'ZTAGS = ',1P2E13.5,' RABST = ',E13.5)
ISN 0438      STOP
ISN 0439      260 CONTINUE
ISN 0440      ZETAA = ZTANSR
ISN 0441      ZTAGS = ZBBST
ISN 0442      M = 0
ISN 0443      CALL OMETA(A,B,ZMGB,ZTAGS,ZTANSR,65,1.00-05,M)
ISN 0444      IF(M.EQ.0) GO TO 262
ISN 0445      WRITE(6,263) ZTAGS,RBBST
ISN 0446      263 FORMAT(/5X,'OMETA FAILED TO CONVERGE FOR ZETA B:',
* /10X,'ZTAGS = ',1P2E13.5,' RBBST = ',E13.5)
ISN 0447      STOP
ISN 0448      262 CONTINUE
ISN 0449      ZETAB = ZTANSR
ISN 0450
C
C FIND GAMMA, ALPHA, BETA, AND S FOR MAPPING TO ETA - PLANE
C
ISN 0454      AP = CDABS(ZETAA + ZETAB)
ISN 0455      AM = CDABS(ZETAA - ZETAB)
ISN 0456      AB = CDABS(ZETAA*ZETAB)
ISN 0457      CHY = (2.000-AP*AP+2.000*AB*AB)/AM/AM
ISN 0458      RT = DSQRT(CHY*CHY-1.000)
ISN 0459      CA = DSQRT(DABS(CHY+RT))
ISN 0460      CB = DSQRT(DABS(CHY-RT))
ISN 0461      SS = DMIN1(CA,CB)
ISN 0462      ZAL = (2.000*ZETAA*ZETAB+(SS*SS*(ZETAA-ZETAB)-ZETAA-ZETAB)
* /DCONJG(ZETAA))/((SS*SS*(ZETAA-ZETAB)+ZETAA+ZETAB-2.000/
* DCONJG(ZETAA))
ISN 0463      ZBT = (2.000*ZETAA*ZETAB-ZAL*(ZETAA+ZETAB))/(ZETAA+ZETAB-2.000

```

```

      * *ZAL)
      ZGM = SS*(ZETAA-ZBT)/(ZETAA-ZAL)
C
415 WRITE(6,245) ZABST
245 FORMAT(/10X,'BEST GUESS FOR ZETA A IS ZABST = ',1P2E12.3)
      WRITE(6,246) ZBBST
246 FORMAT(/10X,'BEST GUESS FOR ZETA B IS ZBBST = ',1P2E12.3)
      WRITE(6,215) ZETAA,ZETAB
215 FORMAT(/ 5X,'ZETAA = ',1P2E13.5,' ZETAB = ',2E13.5)
      WRITE(6,216) ZAL,ZBT,ZGM,SS
216 FORMAT(/ 5X,'ALPHA = ',1P2E11.3,' BETA = ',2E11.3,
      * ' GAMMA = ',2E11.3,' S = ',2E11.3)
C
C FINDING THE LOCATION OF BLADE-SURFACE POINTS IN THE ZETA PLANE ONLY
C INVOLVES PHI(THETA), SINCE R = 1. USE SPLINE INTERPOLATION
C
      PHI(129) = PHI(1) + TPI
      E(129) = E(1) + TPI
      CALL CISPLN(PHI,E,THT,F,129,1,1,2)
      CALL CISPLN(PHI,E,X,F,129,2,KJMX,2)
C
C WHEN THE THETA VS. PHI CURVE IS VERY STEEP, IT MAY HAPPEN THAT THE
C SPLINE-FITTED PHI VS. THETA CURVE IS NOT MONOTONIC: CHECK NOW WHETHER
C THIS HAS HAPPENED, AND REPLACE THE SPLINE-FITTED DATA WITH LINEAR
C INTERPOLATES WHEREVER IT HAS.
C
      DO 810 K = 2,KJMX
      IF(F(K).GE.F(K-1)) GO TO 810
      DO 811 I = 2,129
      IF(E(I).GT.X(K)) GO TO 812
811 CONTINUE
      I = 129
812 IP = I
      IM = I-1
      CON = (PHI(IP)-PHI(IM))/(E(IP)-E(IM))
      J = K
      DO 815 JJ = 1,20
      IF(X(J+JJ).GT.E(IP)) GO TO 816
815 CONTINUE
816 KP = J + JJ - 1
      DO 817 JJ = 1,20
      IF(X(J-JJ).LT.E(IM)) GO TO 818
817 CONTINUE
818 KM = J - JJ + 1
      DO 819 JJ = KM,KP
      TMP = F(JJ)
      F(JJ) = PHI(IM) + CON*(X(JJ)-E(IM))
      WRITE(6,813) JJ,F(JJ),TMP
813 FORMAT(5X,'NOTE: SPLINE FIT REPLACED BY LINEAR INTERPOLATION IN',
      * ' FINDING PHI(',13,') = ',F10.5,' OLD PHI = ',F10.5)
819 CONTINUE
      K = KP
810 CONTINUE
C
      WRITE(6,217)
217 FORMAT(/10X,'MAPPING FROM OMEGA - PLANE TO ZETA - PLANE:',
      * // 4X,'K',14X,'OMEGA',26X,'ZETA',//)
      DO 54 K = 1,KJMX

```

```

ISN 0510      R = DEXP(Y(K))
ISN 0511      ZX = DCMPLX(R*DCOS(X(K)),R*DSIN(X(K)))
ISN 0512      ZYG = DCMPLX(DCOS(F(K)),DSIN(F(K)))
ISN 0513      IF(K.EQ.1.OR.K.EQ.KJMX) ZYG = Z1
ISN 0515      ZCC(K) = ZYG
ISN 0516      54 WRITE(6,218) K,ZX,ZYG
ISN 0517      218 FORMAT(15,1P4E15.5)
C
C ZCC NOW CONTAINS ZETA ON THE BLADE SURFACES
C
ISN 0518      WRITE(6,202)
C
C NOW DO THE MAPPING FROM THE ETA - PLANE TO THE KSI TILDE - PLANE,
C WHERE ETA/S = SN(KSI TILDE)
C
ISN 0519      AK = SS*SS
ISN 0520      AKQ = AK*AK
ISN 0521      AKP = DSQRT(1.000-AKQ)
ISN 0522      AKM = AKP*AKP
ISN 0523      CALL ELLPT(PI,AK,RL,1)
ISN 0524      TBK = 2.000*RL
ISN 0525      CTR = -RL
ISN 0526      CAPK = RL
ISN 0527      CALL ELLPT(PI,AKP,RL,1)
ISN 0528      CAPKPM = RL
ISN 0529      WRITE(6,222) AK,CAPK,AKP,CAPKPM
ISN 0530      222 FORMAT(/10X, 'COMPLETE ELLIPTIC INTEGRALS OF K AND K PRIME',
* ' ARE AS FOLLOWS:',
* //10X, 'K(',F10.6,') = ',F10.6,5X, 'K(',F10.6,') = ',F10.6)
* IFIND=0
ISN 0531      DO 55 I = 1,KJMX
ISN 0532      DO 55 I = 1,KJMX
ISN 0533      ZA(I) = ZGM*(ZCC(I)-ZAL)/(ZCC(I)-ZBT)
C
C ZA(I) NOW HOLDS ETA
C
ISN 0534      ZTD = ZA(I)/SS
ISN 0535      TAU = DREAL(ZTD)
ISN 0536      DLT = DIMAG(ZTD)
ISN 0537      TSQ = TAU*TAU
ISN 0538      DSQ = DLT*DLT
ISN 0539      ART = 1.000 + AKQ*(TSQ + DSQ)
ISN 0540      RT = DSQRT((1.000-AKQ*TSQ)*(1.000-AKQ*TSQ) + AKQ*DSQ*(2.000*
* (1.000+AKQ*TSQ)
* +AKQ*DSQ))
ISN 0541      BRQ = 1.000 + TSQ + DSQ
ISN 0542      BRT = DSQRT((1.000-TSQ)*(1.000-TSQ) + DSQ*(DSQ+2.000+2.000*TSQ))
ISN 0543      ALM = (BRQ-BRT)*(ART-RT)/4.000/AKQ/TSQ
ISN 0544      SGA = (TSQ + DSQ -ALM)
ISN 0545      SGA = SGA/(SGA+1.000-ALM*AKQ*(TSQ+DSQ))
ISN 0546      IF(TAU.EQ.0.000) GO TO 403
ISN 0548      RTALM = DSQRT(ALM)*TAU/DABS(TAU)
ISN 0549      GO TO 404
ISN 0550      403 RTALM = 0.000
ISN 0551      404 SGN = 1.000
ISN 0552      IF(DLT.LT.0.000) SGN = -1.000
ISN 0554      RTSGA = SGN*DSQRT(SGA)
ISN 0555      RTALM = DARSIN(RTALM)
ISN 0556      RTSGA = DARSIN(RTSGA)

```

```

ISN 0557      CALL ELLPT(RTALM,AK,RL,0)
ISN 0558      CALL ELLPT(RTSGA,AKP,AG,0)
ISN 0559      IF(AG.GE.0.0D0) GO TO 57
ISN 0561      AG = -AG
ISN 0562      RL = -RL - TBK
ISN 0563      57 ZA1(I) = DCMLX(RL,AG)
ISN 0564      IF(IFIND.EQ.1) GO TO 55
ISN 0566      IF(I.EQ.1) GO TO 55
ISN 0568      IF((RL-DREAL(ZA1(I-1))).LT.0.0D0) GO TO 55
ISN 0570      KEDGE = I
ISN 0571      KGM = I - 1
ISN 0572      IFIND = 1
ISN 0573      55 CONTINUE

C
C      ZA1(I) NOW HOLDS KSI HAT
C
ISN 0574      WRITE(6,219)
ISN 0575      219 FORMAT(///10X,'MAPPING FROM THE ETA - PLANE TO THE KSI HAT - ',
* ' PLANE',
* ' /// 4X,'K',15X,'ETA',25X,'KSI HAT',//)
ISN 0576      DO 56 K = 1,KJMX
ISN 0577      WRITE(6,218) K,ZA(K),ZA1(K)
ISN 0578      56 CONTINUE

C
C      NOW SET UP A GRID IN THE KSI-HAT PLANE, AND MAP IT BACK
C      TO THE Z - PLANE:
C
ISN 0579      WRITE(6,202)
ISN 0580      WRITE(6,205)
ISN 0581      205 FORMAT(/5X,'MAPPING OF A GRID IN THE KSI-HAT PLANE',
* ' //3X,'K' L', 8X,'KSI HAT',
* ' 15X,'ETA',16X,'ZETA',16X,'OMEGA',
* ' 13X,'Z MAPPED',14X,'ZXY',//)
ISN 0582      ZA2(1) = ZTE
ISN 0583      ZA2(KJMX) = ZTE
ISN 0584      ZA2(KJLE) = ZLE
ISN 0585      DO 91 KJ = 2,KJLM
ISN 0586      K = KJLE - KJ
ISN 0587      91 ZA2(KJ) = ZP(K)
ISN 0588      DO 92 K = 1,KJS
ISN 0589      KJ = KJLE + K
ISN 0590      92 ZA2(KJ) = ZS(K)

C
C      THE ZA2 ARRAY NOW HOLDS THE BLADE-SURFACE COORDINATES, IN THE ORDER:
C      KJ = 1: TE
C      KJ = 2,KJLM: PRESSURE SIDE, FROM TE TO LE
C      KJ = KJLE: LE
C      KJ = KJLP,KJMXM: SUCTION SIDE, FROM LE TO TE
C      KJ = KJMX: TE AGAIN
C
ISN 0591      KMXM1 = KMX - 1
ISN 0592      LMXM1 = LMX - 1
ISN 0593      HCPKPM = CAPKPM/2.0D0
ISN 0594      TWOCPPK = 2.0D0*CAPK
ISN 0595      THCPK = 3.0D0*CAPK
ISN 0596      FCPK = 4.0D0*CAPK
ISN 0597      ZAG = ZAL*ZGM
ISN 0598      EXINV = 1.0D0/EX

```

```

ISN 0599      DXIR = FCPK/DFLOAT(KMXM1)
ISN 0600      SHXTE = DREAL(ZA1(1))
ISN 0601      IF(SHXTE.GT.CTR) SHXTE = SHXTE-FCPK
ISN 0603      SHK = -CAPKPM*CAPKPM/4.000/(THCPK      +SHXTE)
ISN 0604      SHKINV = 1.000/SHK
ISN 0605      IF(ISHEAR.EQ.0) SHKINV = 0.000
ISN 0607      DSHX = 2.000/DFLOAT(KMXM1)
ISN 0608      DSHY = 1.000/DFLOAT(LMXM1)
ISN 0609      ZEETE = ZA(1)
ISN 0610      IO = 0
ISN 0611      K = 1
ISN 0612      L = 1

C
C K - LOOP STARTS HERE
C
ISN 0613      760 CONTINUE
C
C USE LINEAR INTERPOLATION TO GIVE A FIRST GUESS AT Z ON THE
C BLADE SURFACE
C
ISN 0614      XIR = SHXTE+DFLOAT(K-1)*DXIR
ISN 0615      IF(XIR.LT.-THCPK) XIR = XIR + FCPK
ISN 0617      IF(XIR.GT.CAPK) XIR = XIR - FCPK
ISN 0619      DO 371 I = KEDGE,KJMX
ISN 0620      IF(DREAL(ZA1(I)).LT.XIR) GO TO 372
ISN 0622      371 CONTINUE
ISN 0623      DO 373 I = 2,KGM
ISN 0624      IF(DREAL(ZA1(I)).LT.XIR) GO TO 372
ISN 0626      373 CONTINUE
ISN 0627      I = KEDGE
ISN 0628      372 KA = I
ISN 0629      KB = I - 1
ISN 0630      XIA = DREAL(ZA1(KA))
ISN 0631      XIB = DREAL(ZA1(KB))
ISN 0632      IF(KA.NE.KEDGE) GO TO 380
ISN 0634      IF(XIA.LT.0.000)XIA = XIA + FCPK
ISN 0636      IF(XIB.LT.0.000)XIB = XIB + FCPK
ISN 0638      IF(XIR.LT.0.000)XIR = XIR + FCPK
ISN 0640      380 ZGSA = ((XIR-XIA)*ZA2(KB)+(XIB-XIR)*ZA2(KA))/(XIB-XIA)

C
ISN 0641      381 ZGS=ZGSA
ISN 0642      SHX = -1.000+DSHX*DFLOAT(K-1)

C
C L - LOOP STARTS HERE
C
ISN 0643      770 CONTINUE
ISN 0644      SHY=DSHY*DFLOAT(L-1)
ISN 0645      XIM=HCPKPM*(1.000-SHY)
ISN 0646      XIR = (XIM-HCPKPM)**2
ISN 0647      XIR = SHXTE+      TWOPK*(1.000+SHX)+XIR*SHKINV
ISN 0648      ZXI = DCMPLX(XIR,XIM)

C
C BYPASS IMAGE CALCULATIONS FOR POINTS THAT FALL ON THE IMAGES OF
C PLUS OR MINUS INFINITY OR THE T.E.
C
ISN 0649      IF(ISHEAR.EQ.0) GO TO 84
ISN 0651      IF(L.EQ.1.AND.K.EQ.1) GO TO 73
ISN 0653      IF(L.EQ.1.AND.K.EQ.KMX) GO TO 73

```

```

ISN 0655      84 CONTINUE
ISN 0656      IF(L.EQ.LMX.AND.K.EQ.1) GO TO 74
ISN 0658      IF(L.EQ.LMX.AND.K.EQ.KMX) GO TO 74
ISN 0660      IF(10E.EQ.0) GO TO 80
ISN 0662      IF(L.EQ.LMX.AND.K.EQ.KMXH) GO TO 81
ISN 0664      GO TO 80

```

C
C
C

```

ISN 0665      73 ZEETA = ZEETE
ISN 0666      ZETA = (ZBT*ZEETA-ZAG)/(ZEETA-ZGM)
ISN 0667      ZOMA = ZOMTE
ISN 0668      ZFNL = DCMPLX(SHX,SHY)
ISN 0669      ZXY = ZTE*ZGAMMA
ISN 0670      GO TO 710
ISN 0671      74 ZEETA = DCMPLX(SS,0.0D0)
ISN 0672      ZETA = ZETAA
ISN 0673      ZOMA = ZMGA
ISN 0674      ZFNL = DCMPLX(SHX,SHY)
ISN 0675      ZXY = 2.0D0*ZA(LMX-1)-ZA(LMX-2)
ISN 0676      GO TO 710
ISN 0677      81 ZEETA = DCMPLX(-SS,0.0D0)
ISN 0678      ZETA = ZETAB
ISN 0679      ZOMA = ZMGB
ISN 0680      ZXY = 2.0D0*ZA(LMX-1)-ZA(LMX-2)
ISN 0681      ZFNL = DCMPLX(SHX,SHY)
ISN 0682      GO TO 710

```

C
C
C

```

ISN 0683      80 CONTINUE
ISN 0684      CALL JCSELF(XIR,XIM,AKQ,AKM,RLS,AGS,1)
ISN 0685      ZEETA = SS*DCMPLX(RLS,AGS)
ISN 0686      ZETA = (ZBT*ZEETA-ZAG)/(ZEETA-ZGM)
ISN 0687      CALL OMETA(A,B,ZOMA,ZETA,ZTANSR,65,1.0D-00,1)

```

C
C
C

```

NOW DO THE NEWTON-RAPHSON ITERATION TO FIND Z, GIVEN ZBOMK

ISN 0688      ZBOM=ZC*(ZOMA-ZD)/(ZOMA-ZB)
ISN 0689      RAD = CDABS(ZBOM)
ISN 0690      ARG = DATAN2(DIMAG(ZBOM),DREAL(ZBOM))
ISN 0691      RADO = RAD**EXINV
ISN 0692      ARGO = EXINV*ARG
ISN 0693      ZBOMK = DCMPLX(RADO*DCOS(ARGO),RADO*DSIN(ARGO))
ISN 0694      IT=1
ISN 0695      90 Z = ZGS
ISN 0696      ZZT = (Z-ZT)/ZDN
ISN 0697      ZT1 = ZPI*ZZT
ISN 0698      ZT2 = CDSIN(ZT1)
ISN 0699      ZZN = (Z-ZN)/ZDN
ISN 0700      ZT3 = ZPI*ZZN
ISN 0701      ZT4 = CDSIN(ZT3)
ISN 0702      ZF=ZT2/ZT4-ZBOMK
ISN 0703      87 ZFPM = ZPI*CDSIN(ZTN)/ZDN/ZT4/ZT4
ISN 0704      ZNEW = Z - ZF/ZFPM
ISN 0705      IF(KNR.EQ.0) GO TO 83
ISN 0707      IF(KOUNT.EQ.0) WRITE(6,211)
ISN 0709      211 FORMAT(/5X,'DIAGNOSTIC OUTPUT OF THE NEWTON-RAPHSON ITERATIONS:',

```

```

      * /5X,'VARIABLES PRINTED ARE IT,ZBOMK,Z,ZNEW,ZF,ZFPM',/)
      WRITE(6,212) IT,ZBOMK,Z,ZNEW,ZF,ZFPM
212  FORMAT(15,1P10E12.3)
      KOUNT = KOUNT + 1
      IF(KOUNT.GT.INR) STOP
      83 CONTINUE
      IT = IT + 1
      ZGS = ZNEW
      IF(CDABS(ZNEW-Z).LT.1.0D-06) GO TO 72
      IF(CDABS(ZGS).GT.SIZE ) ZGS = ZGSA
      IF(IT.LE.50) GO TO 90
      82 ZGS = ZGSA
      ZNEW = ZERO
      IF(L.EQ.1) ZNEW=ZGSA
      72 CONTINUE
C
      ZFNL = DCMPLX(SHX,SHY)
      ZXY=ZGAMMA*ZNEW
C
C CHECK POINTS ON THE PERIODIC BOUNDARY, NEAR K=1, K=KMX, AND K=KMX/2
C
      IF(ISHEAR.EQ.0) GO TO 710
      IF(L.NE.LMX) GO TO 710
      IF(K.EQ.1.OR.K.EQ.KMX) GO TO 710
      IF(K.GT.4) GO TO 711
214  DY = DIMAG(ZXY-ZT)
      IF(DY.GE.0.0D0) GO TO 717
      ZXY = ZXY + ZI*SG
      GO TO 714
217  IF(DY.LE.SG) GO TO 710
      ZXY = ZXY - ZI*SG
      GO TO 714
211  IF(K.GE.KAA) GO TO 712
      GO TO 710
212  IF(K.GT.KMXH) GO TO 713
      IF(K.EQ.KMXH.AND.10E.EQ.1) GO TO 710
218  DY = DIMAG(ZXY-ZN)
      IF(DY.GE.0.0D0) GO TO 719
      ZXY = ZXY + ZI*SG
      GO TO 718
219  IF(DY.LE.SG) GO TO 710
      ZXY = ZXY - ZI*SG
      GO TO 718
213  IF(K.GT.KBB) GO TO 715
220  DY = DIMAG(ZN-ZXY)
      IF(DY.GE.0.0D0) GO TO 721
      ZXY = ZXY - ZI*SG
      GO TO 720
221  IF(DY.LE.SG) GO TO 710
      ZXY = ZXY + ZI*SG
      GO TO 720
215  IF(K.LT.KMXM4) GO TO 710
222  DY = DIMAG(ZT-ZXY)
      IF(DY.GE.0.0D0) GO TO 723
      ZXY = ZXY - ZI*SG
      GO TO 722
223  IF(DY.LE.SG) GO TO 710
      ZXY = ZXY - ZI*SG

```

ISN 0785

GO TO 722

C
C
C

ISN 0786
ISN 0787
ISN 0788
ISN 0789
ISN 0790
ISN 0791
ISN 0793
ISN 0794
ISN 0795
ISN 0797
ISN 0799
ISN 0800
ISN 0801
ISN 0803

710 WRITE(6,224) K,L,ZXI,ZEETA,ZETA,ZOMA,ZFNL,ZXY
ZA(L) = ZXY
XX(L,L) = DREAL(ZXY)
YY(K,L) = DIMAG(ZXY)
71 CONTINUE
IF(I0.EQ.1) GO TO 790
L = L + 1
IF(L.LE.LMX) GO TO 770
WRITE(6,202)
IF(PNCHZA) WRITE(7,210) (ZA(L),L=1,LMX)
K = K + 1
L = 1
IF(K.LE.KMX) GO TO 760
70 CONTINUE

C
C
C
C

NOW LOOK FOR CASES WHERE ZXY = ZERO, AND TRY AGAIN, USING
INTERPOLATION FROM ALL NEIGHBORING POINTS

ISN 0804
ISN 0805
ISN 0806
ISN 0807
ISN 0808
ISN 0809
ISN 0811
ISN 0812
ISN 0814
ISN 0815
ISN 0816
ISN 0817
ISN 0818
ISN 0820
ISN 0822
ISN 0823
ISN 0824
ISN 0825
ISN 0826
ISN 0827
ISN 0828
ISN 0829
ISN 0831
ISN 0832
ISN 0833
ISN 0834
ISN 0835
ISN 0836
ISN 0837
ISN 0838
ISN 0839
ISN 0840
ISN 0841
ISN 0842
ISN 0843
ISN 0844
ISN 0845
ISN 0846

WRITE(6,214)
214 FORMAT(//5X,'SECOND ATTEMPT TO FIND NON-CONVERGENT CASES',//)
WRITE(6,205)
DO 750 K = 1,KMX
DO 755 L = 2,LMX
IF(XX(K,L).NE.0.0D0) GO TO 755
IF(YY(K,L).NE.0.0D0) GO TO 755
I0 = 1
L1 = L
K1 = K
KM = K - 1
KP = K + 1
IF(K1.EQ.1) KM = KMX - 1
IF(K1.EQ.KMX) KP = 2
LM = L - 1
LP = L + 1
Z11 = DCNPLX(XX(KM,LM),YY(KM,LM))
Z22 = DCNPLX(XX(K1,LM),YY(K1,LM))
Z33 = DCNPLX(XX(KP,LM),YY(KP,LM))
Z44 = DCNPLX(XX(KM,L1),YY(KM,L1))
Z55 = DCNPLX(XX(KP,L1),YY(KP,L1))
IF(L1.EQ.LMX) GO TO 756
Z66 = DCNPLX(XX(KM,LP),YY(KM,LP))
Z77 = DCNPLX(XX(K1,LP),YY(K1,LP))
Z88 = DCNPLX(XX(KP,LP),YY(KP,LP))
ZGSA = (Z11+Z22+Z33+Z44+Z55+Z66+Z77+Z88)/8.000
GO TO 381
756 ZGSA = (Z11+Z22+Z33+Z44+Z55)/5.000
GO TO 381
790 CONTINUE
755 CONTINUE
750 CONTINUE
STOP
100 FORMAT(8F10.4)
202 FORMAT(///)
210 FORMAT(1P4E20.13)
224 FORMAT(2I4,12F10.5)
END

LEVEL 21.7 (DEC 72)

OS/360 FORTRAN H

```

COMPILER OPTIONS - NAME= MAIN,OPT=02,LINECNT=60,SIZE=0000K,
                  SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,ID,XREF
ISN 0002          SUBROUTINE CISPLN(Y,X,E,F,NP,IRTN,NRTN,NPD)
C
C THIS SUBROUTINE FITS A CUBIC SPLINE TO A FUNCTION Y(X), DEFINED BY
C NP PAIRS OF POINTS. THE SECOND DERIVATIVE OF THE FUNCTION
C IS PERIODIC, WITH PERIOD 2*PI. IF NPD = 1, THE FUNCTION
C ITSELF IS ALSO PERIODIC, WHILE IF NPD = 2, THE FUNCTION INCREASES
C BY 2*PI EVERY PERIOD: Y(X+2*PI) = Y(X) + 2*PI. SOLUTIONS FOLLOW
C PAGES 9 - 15 OF
C THE THEORY OF SPLINES AND THEIR APPLICATIONS, BY J. H. AHLBERG,
C E. N. NILSON, AND J. L. WALSH, ACADEMIC PRESS, 1967
C
C NOTE THAT Y,X,E,AND F ARE, RESPECTIVELY, ORDINATE, ABSCISSA,ABSCISSA,
C AND ORDINATE.
C
ISN 0003          IMPLICIT REAL*8(A-H,O-Y),COMPLEX*16(Z)
ISN 0004          DIMENSION Y(160),X(160),E(160),F(160),BDA(160),EM(160),H(160)
ISN 0005          DIMENSION S(160),T(160),V(160),D(160)
ISN 0006          DATA TPI/6.283185307179586/
C
C THIS SECTION (ENTERED WHEN IRTN = 1) USES THE NP PAIRS OF INPUT
C COORDINATES X AND Y TO FIND THE COEFFICIENTS OF THE SPLINE FIT.
C
C THESE COEFFICIENTS - HERE CALLED EM(KJ) - ARE THE SECOND DERIVATIVES
C OF THE FUNCTION.
C
ISN 0007          IF(IRTN.EQ.2) GO TO 20
ISN 0008          NPM = NP - 1
ISN 0009          N = NP + 1
ISN 0010          DO 1 KJ = 2,NP
ISN 0011          1 H(KJ) = X(KJ) - X(KJ-1)
ISN 0012          H(N) = H(2)
ISN 0013          DO 2 KJ = 2,NP
ISN 0014          2 BDA(KJ) = H(KJ+1)/(H(KJ)+H(KJ+1))
ISN 0015          E(1) = 0.000
ISN 0016          F(1) = 0.000
ISN 0017          S(1) = 1.000
ISN 0018          DO 3 KJ = 2,NPM
ISN 0019          DN = 2.000 + (1.000 - BDA(KJ))*E(KJ-1)
ISN 0020          E(KJ) = -BDA(KJ)/DN
ISN 0021          D(KJ) =
ISN 0022          * 6.000*((Y(KJ+1)-Y(KJ))/H(KJ+1)
ISN 0023          * -(Y(KJ)-Y(KJ-1))/H(KJ))/(H(KJ)+H(KJ+1))
ISN 0024          S(KJ) = -(1.000-BDA(KJ))*S(KJ-1)/DN
ISN 0025          3 F(KJ) = (D(KJ)-(1.000-BDA(KJ))*F(KJ-1))/DN
ISN 0026          Y(N) = Y(2)
ISN 0027          IF(NPD.EQ.2) Y(N) = Y(N) + TPI
ISN 0028          D(NP) = 6.000*((Y(N)-Y(NP))/H(2)
ISN 0029          * -(Y(NP)-Y(NPM))/H(NP))/(H(NP)+H(2))
ISN 0030          NPM = NP - 2
ISN 0031          T(NP) = 1.000
ISN 0032          V(NP) = 1.000
ISN 0033          DO 5 I = 1,NPM
ISN 0034          KJ = NP - I
ISN 0035          T(KJ) = E(KJ)*T(KJ+1) + S(KJ)

```

```

ISN 0035      V(KJ) = E(KJ)*V(KJ+1) + F(KJ)
ISN 0035      6 CONTINUE
ISN 0037      EM(NP) = (D(NP)-BDA(NP)*V(2)-(1.000-BDA(NP))*V(NPM))/
*      (BDA(NP)*T(2)+(1.000-BDA(NP))*T(NPM) + 2.000)
ISN 0038      DO 4 I = 1,NPM
ISN 0039      KJ = NP - I
ISN 0040      4 EM(KJ) = E(KJ)*EM(KJ+1) + F(KJ) + S(KJ)*EM(NP)
ISN 0041      SIZE = DABS(X(NP)-X(1))/10.000
ISN 0042      RETURN

C
C THIS SECTION (ENTERED WHEN IRTN = 2) RETURNS NRTN INTERPOLATED
C VALUES OF THE ORDINATE F AT ASSIGNED VALUES OF THE ABSCISSA E.
C

ISN 0043      20 KJ = 2
ISN 0044      DO 21 J = 1,NRTN
ISN 0045      A = E(J)
ISN 0046      24 IF(A.LE.X(KJ)) GO TO 23
ISN 0048      KJ = KJ + 1
ISN 0049      IF(KJ.LE.NP ) GO TO 24
ISN 0051      DF = A - X(NP)
ISN 0052      IF(DF.GT.SIZE) WRITE(6,200) J,E(J),NP,X(NP)
ISN 0054      200 FORMAT(/10X,'WARNING - ENTRY IN CISPLN EXCEEDS END OF BASE ',
*      'ARRAY',/5X,'E(',I3,') = ',1PE16.8,' EXCEEDS X(',I3,') = ',
*      E16.8)
ISN 0055      DF = A - X(1)
ISN 0056      IF(DF.LT.(-SIZE)) WRITE(6,201)J,E(J),X(1)
ISN 0058      201 FORMAT(/10X,'WARNING - ENTRY IN CISPLN IS LESS THAN THE FIRST',
*      ' BASE POINT',
*      /5X,'E(',I3,') = ',1PE16.8,' IS LESS THAN X(1) = ',E16.8)

ISN 0053      KJ = NP
ISN 0060      23 DXA = X(KJ) - A
ISN 0061      DXB = A - X(KJ-1)
ISN 0062      CBA = DXA*DXA*DXA
ISN 0063      CBB = DXB*DXB*DXB
ISN 0064      F(J) = (EM(KJ-1)*CBA+EM(KJ)*CBB)/6.000/H(KJ)
*      + DXA*(V(KJ-1)-EM(KJ-1)*H(KJ)*H(KJ)/6.000)/H(KJ)
*      + DXB*(V(KJ)-EM(KJ)*H(KJ)*H(KJ)/6.000)/H(KJ)

ISN 0065      21 CONTINUE
ISN 0066      RETURN
ISN 0067      END

```

LEVEL 21.7 (DEC 72)

OS/360 FORTRAN H

COMPILER OPTIONS - NAME= MAIN,OPT=02,LINECNT=60,SIZE=0000K,
SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,ID,XREF

ISN 0002 SUBROUTINE ELLPT(AR,AK,ANS,KOMP)
ISN 0003 IMPLICIT REAL*8(A-H,O-Y),COMPLEX*16(Z)
ISN 0004 DIMENSION SQ(12)
ISN 0005 DATA K/1/
ISN 0006 DATA PI/3.141592653589793/

C
C THIS SUBROUTINE EVALUATES THE ELLIPTIC INTEGRAL OF THE FIRST KIND,
C WITH ARGUMENT AR (AN ANGLE IN RADIANS), AND PARAMETER AK (A REAL
C NUMBER).
C THIS EVALUATION USES EQ.(14) OF: Y. L. LUKE, 'APPROXIMATIONS
C FOR ELLIPTIC INTEGRALS', MATH. COMP., VOL. 22 (JULY 1958), PP 627-
C 634, WITH N = 12.
C KOMP = 0.1 FOR THE INCOMPLETE, COMPLETE INTEGRAL, RESPECTIVELY.
C

ISN 0007 IF(K.GT.1) GO TO 11
ISN 0009 K = 2
ISN 0010 TNP = 25.000
ISN 0011 DO 10 M = 1,12
ISN 0012 THM = PI*FLOAT(M)/TNP
ISN 0013 S = DSIN(THM)
ISN 0014 10 SQ(M) = S*S
ISN 0015 11 AKK = AK*AK
ISN 0016 SM = 0.000
ISN 0017 IF(KOMP.EQ.1) GO TO 40
ISN 0019 TN = DTAN(AR)
ISN 0020 DO 20 M = 1,12
ISN 0021 SG = DSQRT(1.000-AKK*SQ(M))
ISN 0022 T = DATAN(SG*TN)
ISN 0023 20 SM = SM + T/SG
ISN 0024 ANS = (AR + 2.000*SM)/TNP
ISN 0025 RETURN
ISN 0026 40 DO 41 M = 1,12
ISN 0027 SG = DSQRT(1.000-AKK*SQ(M))
ISN 0028 41 SM = SM + 1.000/SG
ISN 0029 ANS = PI*(1.000+2.000*SM)/2.000/TNP
ISN 0030 RETURN
ISN 0031 END

COMPILER OPTIONS - NAME= MAIN.OPT=02.LINECNT=60.SIZE=0000K.
SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,ID,XREF
SUBROUTINE JCSELFN(RL,AG,AKQ,AKM,RLS,AGS,M)

ISN 0002

```

C
C WHEN M.EQ.1,
C THIS SUBROUTINE RETURNS THE JACOBIAN ELLIPTIC SINE
C OF A COMPLEX ARGUMENT:
C   RLS + I*AGS = SN(RL + I*AG,K)
C USING THE ARITHMETIC - GEOMETRIC MEAN FORMULA (SEE P.571, HANDBOOK
C OF MATHEMATICAL FUNCTIONS, ED. BY M. ABRAMOWITZ AND I. A. STEGUN,
C U.S. NATIONAL BUREAU OF STANDARDS, APPLIED MATHEMATICS SERIES, 55,
C JUNE 1964) AND THE ADDITION FORMULA FOR THE SN (SEE EQUATION 125.01
C P. 24, OF HANDBOOK OF ELLIPTIC INTEGRALS FOR ENGINEERS AND
C PHYSICISTS, BY P. F. BYRD AND M. D. FRIEDMAN, SPRINGER VERLAG, 1954).
C AKQ = K**2, AKM = 1. - K**2
C
C WHEN M.EQ.2, THE QUANTITIES RETURNED ARE THE REAL AND IMAGINARY PARTS
C OF THE PRODUCT CN(...)*DN(...), WHICH IS THE DERIVATIVE OF THE SN(...)
C

```

```

ISN 0003      IMPLICIT REAL*8(A-H,O-Y),COMPLEX*16(Z)
ISN 0004      DIMENSION A(20),B(20),C(20),PH(20)
ISN 0005      K = 1
ISN 0006      A(1) = 1.0D0
ISN 0007      B(1) = DSQRT(AKM)
ISN 0008      C(1) = DSQRT(AKQ)
ISN 0009      5 DO 6 I = 2,20
ISN 0010          A(I) = (A(I-1)+B(I-1))/2.0D0
ISN 0011          B(I) = DSQRT(A(I-1)*B(I-1))
ISN 0012          C(I) = (A(I-1)-B(I-1))/2.0D0
ISN 0013          IF(DABS(C(I)).LT.1.0D-09) GO TO 7
ISN 0015      6 CONTINUE
ISN 0016      WRITE(6,200) RL,AG
ISN 0017      200 FORMAT(///10X,'JCSELFN FAILED TO CONVERGE FOR Z = ',1P2E15.4)
ISN 0018      STOP
ISN 0019      7 NM = I-1
ISN 0020      N = I
ISN 0021      IF(K.EQ.2) GO TO 20
ISN 0023      PH(N) = A(N)*RL*2**NM
ISN 0024      15 DO 11 L = 1,NM
ISN 0025          J = N - L
ISN 0026      11 PH(J) = (PH(J+1)+DARSIN(C(J+1)*DSIN(PH(J+1))/A(J+1)))/2.0D0
ISN 0027      IF(K.EQ.2) GO TO 40
ISN 0029      SNK = DSIN(PH(1))
ISN 0030      CNK = DCOS(PH(1))
ISN 0031      DNK = CNK/DCOS(PH(2)-PH(1))
ISN 0032      K = 2
ISN 0033      TMP = B(1)
ISN 0034      B(1) = C(1)
ISN 0035      C(1) = TMP
ISN 0036      GO TO 5
ISN 0037      20 PH(N) = A(N)*AG*2**NM
ISN 0038      GO TO 15
ISN 0039      40 SNP = DSIN(PH(1))
ISN 0040      CNP = DCOS(PH(1))
ISN 0041      DNP = CNP/DCOS(PH(2)-PH(1))
ISN 0042      DNM = 1.0D0 - SNP*SNP*DNK*DNK
ISN 0043      IF(M.EQ.2) GO TO 50
ISN 0045      RLS = SNK*DNP/DNM

```

ISN 0046
ISN 0047
ISN 0048
ISN 0049
ISN 0050
ISN 0051
ISN 0052
ISN 0053
ISN 0054
ISN 0055

AGS = CNK*DNK*SNP*CNP/DNM
RETURN
50 RLA = CNK*CNP
AGA = SNK*DNK*SNP*DNP
RLB = DNK*CNP*DNP
AGB = AKQ*SNK*CNK*SNP
RLS = (RLA*RLB-AGA*AGB)/DNM/DNM
AGS = (-RLA*AGB-RLB*AGA)/DNM/DNM
RETURN
END

```

COMPILER OPTIONS - NAME= MAIN,OPT=02,LINECNT=60,SIZE=0000K,
SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,ID,XREF
ISN 0002 SUBROUTINE OMETA(A,B,ZMGA,ZTAGS,ZTANSR,N,EPS,M)
ISN 0003 IMPLICIT REAL*8(A-H,O-Y),COMPLEX*16(Z)
ISN 0004 DIMENSION A(65),B(65),ZC(65)

C
C IF M.EQ.0,
C THIS SUBROUTINE USES NEWTON - RAPHSON TO FIND ZETA(OMEGA): ZMGA IS A
C KNOWN VALUE OF OMEGA, ZTAGS IS THE INITIAL GUESS AT ZETA, ZTANSR IS
C THE SOLUTION, AND EPS IS THE TOLERANCE ON THE ANSWER.
C
C IF M.EQ.1., THIS SUBROUTINE RETURNS THE VALUE OF OMEGA (IN ZMGA)
C FOR A GIVEN VALUE OF ZETA (IN ZTAGS).
C
C IF M.EQ.2, THE QUANTITY RETURNED (IN ZTANSR) IS D OMEGA/D ZETA, FOR
C GIVEN VALUES OF OMEGA (IN ZMGA) AND ZETA (IN ZTAGS)
C
ISN 0005 Z1 = DCMPLX(1.0D0,0.0D0)
ISN 0006 ZETA = ZTAGS
ISN 0007 NM = N - 1
ISN 0008 IT = 1
ISN 0009 5 CONTINUE
ISN 0010 ZSMA = DCMPLX(A(N),B(N))
ISN 0011 IF(M.EQ.1) GO TO 22
ISN 0012 ZSMB = ZSMA*DFLOAT(NM)
ISN 0013 IF(M.EQ.2) GO TO 40
ISN 0014 DO 10 J = 1,NM
ISN 0015 ZSMA = ZETA*ZSMA + DCMPLX(A(N-J),B(N-J))
ISN 0016 ZSMB = ZETA*ZSMB + DCMPLX(A(N-J),B(N-J))*DFLOAT(NM-J)
ISN 0017 10 CONTINUE
ISN 0018 ZEXP = CDEXP(ZSMA)
ISN 0019 ZG = ZETA*ZEXP - ZMGA
ISN 0020 ZDG = ZEXP*(Z1 + ZSMB)
ISN 0021 ZETOLD = ZETA
ISN 0022 ZETA = ZETOLD - ZG/ZDG
ISN 0023 IF(CDABS(ZETA-ZETOLD).LT.EPS) GO TO 20
ISN 0024 IT = IT + 1
ISN 0025 IF(CDABS(ZETA).GT.1.0D0) ZETA = 0.9D0*ZETA/CDABS(ZETA)
ISN 0026 IF(IT.LE.50) GO TO 5
ISN 0027 M = 5
ISN 0028 RETURN
ISN 0029 20 ZTANSR = ZETA
ISN 0030 RETURN
ISN 0031 22 DO 21 J = 1,NM
ISN 0032 ZSMA = ZETA*ZSMA + DCMPLX(A(N-J),B(N-J))
ISN 0033 21 CONTINUE
ISN 0034 ZEXP = CDEXP(ZSMA)
ISN 0035 ZMGA = ZETA*ZEXP
ISN 0036 RETURN
ISN 0037 40 DO 41 J = 1,NM
ISN 0038 41 ZSMB = ZETA*ZSMB + DCMPLX(A(N-J),B(N-J))*DFLOAT(NM-J)
ISN 0039 ZTANSR = ZMGA*(Z1+ZSMB)/ZTAGS
ISN 0040 RETURN
ISN 0041 END
ISN 0042
ISN 0043
ISN 0044
ISN 0045
ISN 0046

```

LEVEL 21.7 (DEC 72)

OS/360 FORTRAN H

```

COMPILER OPTIONS - NAME= MAIN,OPT=02,LINECNT=60,SIZE=0000K,
                  SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,ID,XREF
ISN 0002          SUBROUTINE SHAPE(C,H,G,EX,ZP,ZS,KJS,KJP)
ISN 0003          IMPLICIT REAL*8(A-H,O-Y),COMPLEX*16(Z)
ISN 0004          DIMENSION ZS(80),ZP(80)
ISN 0005          DO 10 K = 1,KJS
ISN 0006          10 READ(5,100) ZS(K)
ISN 0007          DO 20 K = 1,KJP
ISN 0008          20 READ(5,100) ZP(K)
ISN 0009          100 FORMAT(8F10.0)
ISN 0010          RETURN
ISN 0011          END

```

LEVEL 21.7 (DEC 72)

OS/360 FORTRAN H

```

COMPILER OPTIONS - NAME= MAIN,OPT=02,LINECNT=60,SIZE=0000K,
                  SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,ID,XREF
ISN 0002          SUBROUTINE SHUFL(N,ZA,ZCC)
C
C THIS SUBROUTINE TAKES THE N COMPLEX VALUES IN ARRAY ZA, WHICH WERE
C COMPUTED AND STORED IN REVERSE BINARY ORDER BY FFT2 AND "SHUFFLES"
C THEM INTO PROPER ORDER USING ARRAY ZCC FOR INTERMEDIATE STORAGE.
C N IS ASSUMED TO HAVE THE FORM 2**M.
C
ISN 0003          IMPLICIT REAL*8(A-H,O-Y),COMPLEX*16(Z)
ISN 0004          DIMENSION ZA(65),ZCC(65),IAL(6),KR(64)
ISN 0005          DATA KALL/0/
ISN 0006          DATA IAL/6*0/
ISN 0007          IF (KALL.EQ.1) GO TO 10
ISN 0008          KALL = 1
ISN 0009          DO 341 JP = 1,N
ISN 0010          J = JP - 1
ISN 0011          IAL(6) = J/32
ISN 0012          J = J - 32*IAL(6)
ISN 0013          IAL(5) = J/16
ISN 0014          J = J - 16*IAL(5)
ISN 0015          IAL(4) = J/8
ISN 0016          J = J - 8*IAL(4)
ISN 0017          IAL(3) = J/4
ISN 0018          J = J - 4*IAL(3)
ISN 0019          IAL(2) = J/2
ISN 0020          J = J - 2*IAL(2)
ISN 0021          IAL(1) = J
ISN 0022          341 KR(JP) =
ISN 0023          * 32*IAL(1)+16*IAL(2)+8*IAL(3)+4*IAL(4)+2*IAL(5)+IAL(6)
ISN 0024          10 DO 342 J = 1,N
ISN 0025          342 ZCC(J) = ZA(KR(J)+1)
ISN 0026          DO 360 JP = 1,N
ISN 0027          360 ZA(JP) = ZCC(JP)
ISN 0028          RETURN
ISN 0029          END

```

LEVEL 21.7 (DEC 72)

OS/360 FORTRAN H

COMPILER OPTIONS - NAME= MAIN,OPT=02,LINECNT=60,SIZE=0000K,
SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,ID,XREF

```

ISN 0002      SUBROUTINE THDGRK
C
C THIS SUBROUTINE MAPS AN OVAL TO A UNIT CIRCLE, USING A VARIANT OF
C THE THEODORSEN-GARRICK TRANSFORMATION AND FAST FOURIER TRANSFORM
C TECHNIQUES. (SEE REFERENCES AT BEGINNING OF MAIN PROGRAM).
C
ISN 0003      IMPLICIT REAL*8(A-H,O-Y),COMPLEX*16(Z)
ISN 0004      COMMON/TGINTG/N,NP1,NP2,N2,NB2,NB2P1,IP,IPMX,ITP,IWK,IMX,KJMX
ISN 0005      COMMON/TGCMPLX/Z1,ZW2N,ZI,ZNN,ZA,ZCC,ZA1,ZA2
ISN 0006      COMMON/TGDBLE/PBN,OM,OMM,ANGERR,A,B,E,F,THT,PHI,X,Y
ISN 0007      DIMENSION X(160),Y(160),E(150),F(150),THT(160)
ISN 0008      DIMENSION PHI(130),A(65),B(65),ZCC(65),
*             ZA(165),ZA1(165),ZA2(165),
*             IWK(7)
ISN 0009      DIMENSION ITP(100)
C
ISN 0010      301 DO 300 I = 1,N2
ISN 0011      300 PHI(I)=PBN*DFLOAT(I-1)
C
C THE FOLLOWING CALSPAN LIBRARY ROUTINE PLACES A ZERO IN THE LOCATIONS
C FROM THE FIRST TO THE LAST ARGUMENT OF THE CALL. THE THIRD ARGUMENT
C GIVES THE LENGTH SPECIFICATION OF EACH ENTRY.
C
ISN 0012      CALL CLEAR(A(1),A(65),8)
ISN 0013      CALL CLEAR(B(1),B(65),8)
ISN 0014      CALL CLEAR(E(1),E(150),8)
ISN 0015      CALL CLEAR(ZCC(1),ZCC(65),16)
C
ISN 0016      WRITE(6,401)
ISN 0017      401 FORMAT(/10X,'PROGRESS OF PHI / THETA ITERATIONS IS AS FOLLOWS: '//
1 3X,'IT',4X,'DEMX',4X,'NO. OF THETA REVERSALS')
C
ISN 0018      IT = 1
C
C FIRST GUESS IS THETA - THETA(TRAILING EDGE) = PHI
C
ISN 0019      DO 315 K = 1,N2
ISN 0020      315 E(K) = X(1) + PHI(K)
ISN 0021      305 DEMX = 0.000
ISN 0022      B(1) = 0.000
C
C USE AJ AND BJ TO GET NEXT APPROXIMATION TO THETA
C
ISN 0023      ZCC(1) = DCMLPX(B(1),0.000)
ISN 0024      DO 302 JP = 2,N
ISN 0025      302 ZCC(JP) = DCMLPX(B(JP)/2.000,-A(JP)/2.000)
ISN 0026      ZCC(NP1) = DCMLPX(0.000,0.000)
ISN 0027      ZW = Z1/ZW2N
ISN 0028      DO 303 NP = 1,NB2P1
ISN 0029      ZA1(NP) = ZCC(NP) + DCONJG(ZCC(NP2-NP))
ISN 0030      ZW = ZW*ZW2N
ISN 0031      303 ZA2(NP) = ZW*(ZCC(NP)-DCONJG(ZCC(NP2-NP)))
ISN 0032      DO 304 NP = 1,NB2P1
ISN 0033      304 ZA(NP) = ZA1(NP) + ZI*ZA2(NP)
ISN 0034      DO 309 NP = 2,NB2
ISN 0035      309 ZA(NP2-NP) =

```

```

      * DCONJG(ZA1(NP)) + ZI*DCONJG(ZA2(NP))
ISN 0036 CALL FFT2(ZA,6,IWK)
ISN 0037 CALL SHUFL(N,ZA,ZCC)
ISN 0038 B(1) = X(1) - PHI(1) - DREAL(ZA(1))
ISN 0039 DO 306 K = 1,64
ISN 0040 TMP = E(2*K-1)
ISN 0041 E(2*K-1) = DREAL(ZA(K)) + PHI(2*K-1) + B(1)
ISN 0042 THT(2*K-1) = E(2*K-1)
ISN 0043 DEM = DABS(TMP-E(2*K-1))
ISN 0044 IF(DEM.GT.DEMX) DEMX = DEM
ISN 0046 E(2*K-1) = OM*E(2*K-1) + OMM*TMP
ISN 0047 TMP = E(2*K)
ISN 0048 E(2*K) = DIMAG(ZA(K)) + PHI(2*K) + B(1)
ISN 0049 THT(2*K) = E(2*K)
ISN 0050 DEM = DABS(TMP-E(2*K))
ISN 0051 IF(DEM.GT.DEMX) DEMX = DEM
ISN 0053 E(2*K) = OM*E(2*K) + OMM*TMP
ISN 0054 306 CONTINUE
ISN 0055 IF(IT.NE.ITP(IP)) GO TO 840
ISN 0057 IP = IP + 1
ISN 0058 WRITE(6,841) IT
ISN 0059 841 FORMAT(/3X,
      * THETA BEFORE AND AFTER RELAXATION AT IT = ',I4)
ISN 0060 WRITE(6,832)(THT(I),I=1,128)
ISN 0061 WRITE(6,832)(E(I),I=1,128)
ISN 0062 832 FORMAT(1P10E13.5)
ISN 0063 840 CONTINUE
ISN 0064 IF(IT.GE.ITP(IPMX)) STOP

C
C NOW USE THESE THETAS TO GET THE NEXT APPROXIMATION TO LN R(K)
C
ISN 0066 CALL CISPLN(Y,X,E,F,KJMX,2,128,1)

C
C NOW FIND THE AJ AND BJ COEFFICIENTS CORRESPONDING TO THE LN R(K) DATA
C
ISN 0067 DO 310 JP = 1,N
ISN 0068 310 ZA(JP) = DCMPLX(F(2*JP-1),F(2*JP))
ISN 0069 DO 307 NP = 1,N
ISN 0070 307 ZA(NP) = DCONJG(ZA(NP))
ISN 0071 CALL FFT2(ZA,6,IWK)
ISN 0072 DO 308 JP = 1,N
ISN 0073 308 ZA(JP) = DCONJG(ZA(JP))/ZNN
ISN 0074 CALL SHUFL(N,ZA,ZCC)
ISN 0075 ZA(65) = ZA(1)
ISN 0076 DO 311 NP = 1,NB2P1
ISN 0077 ZA1(NP) = (DCONJG(ZA(NP2-NP)) + ZA(NP))/2.000
ISN 0078 311 ZA2(NP) = ZI*(DCONJG(ZA(NP2-NP)) - ZA(NP))/2.000
ISN 0079 ZW = ZW2N
ISN 0080 DO 312 NP = 1,NB2P1
ISN 0081 ZW = ZW/ZW2N
ISN 0082 312 ZCC(NP) = (ZA1(NP)+ZA2(NP)*ZW)/2.000
ISN 0083 ZW = Z1/ZW
ISN 0084 DO 313 I = 2,NB2
ISN 0085 ZW = ZW*ZW2N
ISN 0086 NP = NB2 + I
ISN 0087 ZBB = (ZA1(NP2-NP)+ZA2(NP2-NP)*ZW)/2.000
ISN 0088 313 ZCC(NP) = DCONJG(ZBB)
ISN 0089 A(1) = DREAL(ZCC(1))

```

```

ISN 0090      ZCC(NP1)=0.5D0*DCONJG(ZA1(1)-ZA2(1))
ISN 0091      A(65) = DREAL(ZCC(65))
ISN 0092      DO 314 NP = 2,N
ISN 0093      A(NP) = 2.0D0*DREAL(ZCC(NP))
ISN 0094      314 B(NP) = 2.0D0*DIMAG(ZCC(NP))

C
C CHECK FOR CONVERGENCE
C
ISN 0095      IF(IT.EQ.1) DEMX = 1.0D0
ISN 0097      NRV = 0
ISN 0098      DO 809 LL = 2,128
ISN 0099      809 IF(E(LL).LE.E(LL-1)) NRV = NRV + 1
ISN 0101      WRITE(6,400) IT,DEMX,NRV
ISN 0102      400 FORMAT(15,1PE12.3,18)
ISN 0103      IF(DEMX.LT.ANGERR) GO TO 316
ISN 0105      IT = IT + 1
ISN 0106      IF(IT.LE.IMX) GO TO 305
ISN 0108      WRITE(6,213)
ISN 0109      213 FORMAT(//5X,'ITERATIONS FOR A AND B DID NOT CONVERGE')
ISN 0110      STOP

C
ISN 0111      316 WRITE(6,203) IT,OM,DEMX
ISN 0112      203 FORMAT(//10X,'A AND B ITERATIONS CONVERGED AT IT = ',I3,
*          ' USING OM = ',F6.3,' . MAXIMUM ANGULAR ERROR = ',
*          1PE12.3,' RADIANS')
ISN 0113      WRITE(6,220)
ISN 0114      220 FORMAT(// 4X,'K',11X,'THETA',15X,'PHI',//)
ISN 0115      DO 69 K = 1,128
ISN 0116      69 WRITE(6,221) K,E(K),PHI(K)
ISN 0117      221 FORMAT(15,1P2E20.6)

C
ISN 0118      RETURN
ISN 0119      END

```

Appendix C
DICTIONARY OF VARIABLES

FORTTRAN SYMBOL	ALGEBRAIC EQUIVALENT	DEFINITION, USE, COMMENTS
A(J), B(J)	A_j, B_j	Eq. 2-20, 2-21
AK	$k = S^2$	Eq. 2-23
AKP	$k' = \sqrt{1 - k^2}$	Eq. 2-30
AKQ	k^2	-
ALM	λ	Eq. 2-31
CAPK	$K(k)$	Eq. 2-34
CAPKPM	$K'(k')$	Eq. 2-34
DLT	δ	Eq. 2-28
DXIM, DXIR	$Re(\Delta \hat{\xi}), Im(\Delta \hat{\xi})$	Eq. 2-38, 4-4
EX	$1/(2 - \frac{\tau}{\pi})$	See Figure 2
EXINV	$2 - \frac{\tau}{\pi}$	-
G, H	G, H	See Figure 1
HCPKPM	$\frac{1}{2} K'(k')$	-
OM	-	Relaxation factor used in ϕ, θ mapping
PBN	$\frac{\pi}{N}$	-
PHI	ϕ	-
SGA	σ	Eq. 2-31
SS	S	Eqs. 2-23
TAU	τ	Eq. 2-28

FORTTRAN SYMBOL	ALGEBRAIC EQUIVALENT	DEFINITION, USE, COMMENTS
TPI	2π	-
XIR, XIM	$Re(\hat{\xi}), Im(\hat{\xi})$	-
ZA(N)	$A(n)$	Eq. A-9; used elsewhere for temporary storage
ZB	b	Eq. 2-9
ZC	c	Eq. 2-9
ZD	a	Eq. 2-9
ZI	$\sqrt{-1}$	-
ZN, ZT, ZLE, ZTE	$\bar{z}_N, \bar{z}_T$	See Figure 1
ZAL, ZBT, ZGM	α, β, δ	Eq. 2-22
ZA1(N), ZA2(N), ZCC(N)	$A_1(n), A_2(n), C(n)$	See, for example, Eq. A-9. Also used for temporary storage
ZBOM	Ω	-
ZBOMK	Ω^K	-
ZEETA	η	-
ZETA	ζ	-
ZNTRD		centroid of the ω -plane
ZOMSTR		Ω - plane image of ZNTRO
ZOMS(K), ZOMP(K)	ω_s, ω_p	-
ZPLUS, ZMINUS	Ω^+, Ω^-	Eq. 1-6
ZXI	$\hat{\xi}$	-
ZXY		$x + iy$

Appendix D LISTING OF METRIC GENERATOR PROGRAM

LEVEL 21.7 (DEC 72)

OS/360 FORTRAN H

COMPILER OPTIONS - NAME= MAIN,OPT=02,LINECNT=60,SIZE=0000K,
SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,ID,XREF

```

C
C PROGRAM CIMTVL - A PROGRAM TO CALCULATE THE METRICS OF A COORDINATE
C TRANSFORMATION FOR TURBOMACHINERY CASCADES. FOR DOCUMENTATION,SEE:
C
C J. P. NENNI AND W. J. RAE, EXPERIENCE WITH THE DEVELOPMENT OF
C AN EULER CODE FOR ROTOR ROWS, ASME PAPER 83-GT-36, MARCH 1983
C W. J. RAE, A COMPUTER PROGRAM FOR THE IVES TRANSFORMATION IN
C TURBOMACHINERY CASCADES, AFOSR TR-81-0154, ADA096416, NOV 1980
C W. J. RAE, MODIFICATIONS OF THE IVES-LIUTERMOZA CONFORMAL-
C MAPPING PROCEDURE FOR TURBOMACHINERY CASCADES, ASME PAPER
C 83-GT-116, MARCH 1983
C W.J. RAE, REVISED COMPUTER PROGRAM FOR EVALUATING THE IVES
C TRANSFORMATION IN TURBOMACHINERY CASCADES, CALSPAN CORPORATION
C REPORT 7177-A-1, JULY 1983
C
C THE PROGRAM READS A CARD DECK CONTAINING THE COORDINATES X(K,L) ,
C Y(K,L) ; K = 1,KMX ; L = 1,LMX, AND FINDS BY FINITE DIFFERENCES
C THE METRICS OF A TRANSFORMATION TO A RECTANGLE(KSI,ETA). IN THIS
C RECTANGLE, THE IMAGE OF THE BLADE SURFACE LIES ALONG ONE SIDE, AND
C THE IMAGES OF THE POINTS AT INFINITY LIE AT CORNERS OF THE RECTANGLE
C (SEE THE REFERENCES ABOVE). THE METRICS ARE WRITTEN ON TAPE 1.
C
ISN 0002      IMPLICIT REAL*8(A-H,O-Z)
ISN 0003      DIMENSION Q(410,6)
ISN 0004      READ(5,100) KMX,LMX
ISN 0005      100 FORMAT(20I4)
ISN 0006      READ(5,101) SG
ISN 0007      101 FORMAT(8F10.4)
C
C SG IS THE SLANT GAP BETWEEN BLADES
C
ISN 0008      DO 10 K = 1,KMX
ISN 0009      KMILMX = (K-1)*LMX
ISN 0010      LKA = KMILMX + 1
ISN 0011      LKB = KMILMX + LMX
ISN 0012      10 READ(5,200)((Q(LK,5),Q(LK,6)),LK=LKA,LKB)
ISN 0013      200 FORMAT(1P4E20.13)
C
C Q(LK,5) CONTAINS X(K,L), Q(LK,6) CONTAINS Y(K,L)
C
ISN 0014      KM=KMX-1
ISN 0015      LM=LMX-1
C
C --- SET X AND Y AT K=KMX AND L=1
C
ISN 0016      LK=KM*LMX+1
ISN 0017      Q(LK,5)=Q(1,5)
ISN 0018      Q(LK,6)=Q(1,6)
C
C --- FIND X AND Y AT K=1 AND L=LMX
C
ISN 0019      Q(LMX,5)=2D0*Q(LM,5)-Q(LMX-2,5)
ISN 0020      Q(LMX,6)=2D0*Q(LM,6)-Q(LMX-2,6)
C
C --- FIND X AND Y AT K=KMX AND L=LMX
C

```

```

ISN 0021      LK=KMX*LMX
ISN 0022      Q(LK,5)=2D0*Q(LK-1,5)-Q(LK-2,5)
ISN 0023      Q(LK,6)=2D0*Q(LK-1,6)-Q(LK-2,6)
ISN 0024      TDELXI=4D0/DFLOAT(KM)
ISN 0025      TDELZT=2D0/DFLOAT(LM)

C
ISN 0026      DO 520 K=1,KMX
ISN 0027      KM1LMX=(K-1)*LMX

C
ISN 0028      DO 510 L=1,LMX
ISN 0029      LK=KM1LMX+L
ISN 0030      LP1=LK+1
ISN 0031      LM1=LK-1
ISN 0032      KP1=LK+LMX
ISN 0033      KM1=LK-LMX

C
ISN 0034      IF(K.EQ.1) GO TO 501
ISN 0036      IF(K.EQ.KMX) GO TO 501
ISN 0038      IF(L.EQ.1) GO TO 504
ISN 0040      IF(L.EQ.LMX) GO TO 505

C
C --- K=2 TO KM AND L=2 TO LM      (CENTERED DIFFERENCES)
C
ISN 0042      DXDXI=(Q(KP1,5)-Q(KM1,5))/TDELXI
ISN 0043      DYDXI=(Q(KP1,6)-Q(KM1,6))/TDELXI
ISN 0044      DXDZT=(Q(LP1,5)-Q(LM1,5))/TDELZT
ISN 0045      DYDZT=(Q(LP1,6)-Q(LM1,6))/TDELZT
ISN 0046      GO TO 509

C
ISN 0047      501 IF(L.EQ.1) GO TO 510
ISN 0049      IF(L.EQ.LMX) GO TO 510

C
C --- K=1 OR KMX AND L=2 TO LM
C
ISN 0051      LK2=LMX+L
ISN 0052      LKKM=(KM-1)*LMX+L
ISN 0053      DXDXI=(Q(LK2,5)-Q(LKKM,5))/TDELXI
ISN 0054      DYDXI=(Q(LK2,6)-Q(LKKM,6))/TDELXI
ISN 0055      DXDZT=(Q(LP1,5)-Q(LM1,5))/TDELZT
ISN 0056      DYDZT=(Q(LP1,6)-Q(LM1,6))/TDELZT
ISN 0057      GO TO 509

C
C --- L=1 AND K=2 TO KM
C
ISN 0058      504 LK1=KM1LMX+1
ISN 0059      LK2=LK1+1
ISN 0060      LK3=LK2+1
ISN 0061      DXDZT=(-3D0*Q(LK1,5)+4D0*Q(LK2,5)-Q(LK3,5))/TDELZT
ISN 0062      DYDZT=(-3D0*Q(LK1,6)+4D0*Q(LK2,6)-Q(LK3,6))/TDELZT
ISN 0063      GO TO 506

C
C --- L=LMX AND K=2 TO KM
C
ISN 0064      505 LKS=(KMX-K)*LMX+LM
ISN 0065      LKB=KM1LMX+LM
ISN 0066      DXDZT=(Q(LKS,5)-Q(LKB,5))/TDELZT
ISN 0067      DQ=SG
ISN 0068      IF(K.GT.KMX) DQ=-SG

```

```

ISN 0070      DYDZT=(Q(LKS,6)+DQ-Q(LKB,6))/TDELZT
C
ISN 0071      506 DXDXI=(Q(KP1,5)-Q(KM1,5))/TDELXI
ISN 0072      DYDXI=(Q(KP1,6)-Q(KM1,6))/TDELXI
C
ISN 0073      509 RDM1=(DXDXI*DYDZT-DXDZT*DYDXI)
C
C      STORE DERIVATIVES IN THE ORDER DKSI/DX,DKSI/DY,DETA/DX,DETA/DY
C
ISN 0074      Q(LK,1)=DYDZT/RDM1
ISN 0075      Q(LK,2)=-DXDZT/RDM1
ISN 0076      Q(LK,3)=-DYDXI/RDM1
ISN 0077      Q(LK,4)= DXDXI/RDM1
ISN 0078      510 CONTINUE
ISN 0079      520 CONTINUE
C
C      STORE DERIVATIVES ON TAPE 1
C
ISN 0080      LKMX=LMX*KMX
C      WRITE(1) KMX,LMX,LKMX,((Q(I,J),I=1,LKMX),J=1,4)
C
C
ISN 0087      STOP
ISN 0088      END

```

REFERENCES

1. Rae, W.J., "A Computer Program for the Ives Transformation in Turbo-machinery Cascades", Calspan Report No. 6275-A-3, AFOSR TR-81-0154 (November 1980).
2. Thompson, J.F., ed., Numerical Grid Generation, Elsevier Science Publishing Co., New York (1982).
3. Ives, D.C. and Liutermoza, J.F., "Analysis of Transonic Cascade Flow Using Conformal Mapping and Relaxation Techniques", AIAA Journal 15 (1977), pp. 647-652.
4. Rae, W.J., "Modifications of the Ives-Liutermoza Conformal-Mapping Procedure for Turbomachinery Cascades", ASME Paper 83-GT-116 (March 1983).
5. Rae, W.J., and Homicz, G.F., "A Rectangular-Coordinate Method for Calculating Nonlinear Transonic Potential Flowfields in Compressor Cascades", AIAA Paper 78-248 (January 1978).
6. Cooley, J.W., Lewis, P.A.W., and Welch, P.D., "The Fast Fourier Transform Algorithm: Programming Considerations in the Calculations of Sine, Cosine and Laplace Transforms", Journal of Sound and Vibration 12, (1970) pp. 315-337.
7. Warschawski, S.E., "On Theodorsen's Method of Conformal Mapping of Nearly Circular Regions", Quarterly of Applied Mathematics 3, (1945) pp. 12-28.
8. Henrici, P., "Fast Fourier Methods in Computational Complex Analysis", SIAM Review 21, (1979) pp. 481-527.
9. Mokry, M., "Comment on Analysis of Transonic Cascade Flow Using Conformal Mapping and Relaxation Techniques", AIAA Journal 16, No. 1, (January 1978) p. 96.
10. Nielsen, J.N. and Perkins, E.W., "Charts for the Conical Part of the Downwash Field of Swept Wings at Supersonic Speeds", NACA Technical Note 1780, (December 1948), Appendix C.
11. Byrd, P.F., and Friedman, M.D., Handbook of Elliptic Integrals for Engineers and Physicists. Springer Verlag, Berlin (1954).
12. Luke, Y.L., "Approximations for Elliptic Integrals", Mathematics of Computation, 22 (1968) 627-634.
13. Erdelyi, A., et al. Higher Transcendental Functions, Volume 2, p. 377, McGraw-Hill Book Company, New York (1953).

REFERENCES (Cont'd)

14. Nenni, J.P. and Rae, W.J., "Experience with the Development of an Euler Code for Rotor Rows", ASME Paper 83-GT-36 (March 1983).
15. Abramowitz, M. and Stegun, I.A., Handbook of Mathematical Functions, National Bureau of Standards, Applied Mathematics Series 55 (1964).
16. Kober, H., Dictionary of Conformal Transformations, Dover Publications, New York (1957).
17. Ahlberg, J.H., Nilson, E.N., and Walsh, J.L., The Theory of Splines and Their Applications Academic Press, New York (1967).
18. Hartree, D.R., Numerical Analysis, 2nd Edition, Oxford, Clarnedon Press (1958).

ATE
LMED
8