

## An Improved Method for Numerical Conformal Mapping

By John K. Hayes, David K. Kahaner and Richard G. Kellner\*

**Abstract.** A new technique for the numerical conformal mapping of a planar region onto the unit disk has been presented and tested by Symm. By elaborating on his methods, we have improved the accuracy of the numerical results by up to four orders of magnitude. For illustration, our methods have been applied to several of the same regions considered in the literature by Symm and Rabinowitz. A flexible FORTRAN code and User's Guide are reproduced on the microfiche card in this issue.

**1. Introduction.** A new technique for the numerical conformal mapping of a planar region onto the unit disk has been presented and tested by Symm [7], [8], [9]. By elaborating on his methods, we have improved the accuracy of the numerical results by up to four orders of magnitude. For illustration, our methods have been applied to several of the same regions considered in the literature by Symm [7] and Rabinowitz [6].

In this paper, we numerically approximate the univalent function  $f(z)$  which maps the bounded, simply-connected region  $D$  of the complex plane onto the unit disk. Let  $L$  be the boundary of  $D$  and choose  $z_0 \in D$  to be the point which is to be mapped into the center of the unit disk. It is known [7] that

$$w = f(z) = \exp[\log(z - z_0) + g(z) + ih(z)],$$

where  $g$  and  $h$  are real-valued harmonic conjugates, and  $g$  satisfies

$$\nabla^2 g(z) = 0 \quad \text{for } z \in D,$$

and

$$g(z) = -\log |z - z_0| \quad \text{for } z \in L.$$

The mapping function  $f(z)$  above is determined only to within an arbitrary rotation. This depends upon the branch of the logarithm used in the computation and the additive constant chosen for the function  $h$ .

Symm [7] numerically solves the integral equation of the first kind

$$(1) \quad \int_L \sigma(\zeta) \log |z - \zeta| |d\zeta| = -\log |z - z_0|, \quad z \in L.$$

This may always be done, subject to a possible rescaling of the region  $D$  [3], [5]. Then, for any  $z \in D + L$ ,  $g$  and  $h$  have the representation

Received June 21, 1971.

*AMS 1970 subject classifications.* Primary 30A28; Secondary 65E05.

*Key words and phrases.* Numerical conformal mapping, numerical solution of integral equations of the first kind.

\* This research was supported by the U. S. Atomic Energy Commission under Contract W-7405-ENG-36.

$$(2) \quad g(z) = \int_L \sigma(\zeta) \log |z - \zeta| |d\zeta|,$$

$$(3) \quad h(z) = \int_L \sigma(\zeta) \arg(z - \zeta) |d\zeta|.$$

The function  $\arg$  must be chosen in an appropriate manner [4].

**2. Description of the Method.** Our procedure for numerically mapping a region can be divided into two operational steps:

- (i) Solve Eq. (1) for the function  $\sigma$ .
- (ii) Evaluate Eqs. (2) and (3) for each point  $z \in D$  where we want to find  $f(z)$ .

Let the curve  $L$  have the parametric representation  $\{(\nu(t), w(t)) \mid t \in (0, d)\}$  with respect to arc length  $t$ . Here,  $d$  is the length of  $L$ . Define  $\zeta(t) = \nu(t) + iw(t)$ . With this notation, Eqs. (2) and (3) become

$$(4) \quad g(z) = \int_0^d \sigma(t) \log |z - \zeta(t)| dt, \quad z \in D + L,$$

$$(5) \quad h(z) = \int_0^d \sigma(t) \arg(z - \zeta(t)) dt, \quad z \in D + L,$$

where we have used  $\sigma(t)$  for  $\sigma(\zeta(t))$ .

Now, we will sketch how we compute the function  $\sigma(t)$  numerically. A detailed development is contained in [1]. Since  $\sigma(t)$  is a function of arc length, we extend it continuously as a periodic function on  $(-\infty, +\infty)$ . For ease of explanation, assume  $\sigma(t) \in C^3(-\infty, +\infty)$  and that  $L$  has no corners. Place on  $L$  a uniform mesh of  $n$  points ( $n$  even), each  $h = d/n$  units apart. (In actual practice, one might wish to divide  $L$  into several sections. The mesh points on each section would then be uniform with respect to arc length on that section. See the user's guide in the microfiche portion of this issue and also Example 2 of this paper.) Define a set of piecewise polynomial functions  $p_1(t), p_2(t), \dots, p_n(t)$  by

$$\begin{aligned} p_1(t) &= (t - h)(t - 2h)/2h^2, & 0 \leq t \leq 2h, \\ &= (t + h)(t + 2h)/2h^2, & -2h \leq t \leq 0, \\ &= 0, & \text{otherwise,} \\ p_2(t) &= -t(t - 2h)/h^2, & 0 \leq t \leq 2h, \\ &= 0, & \text{otherwise,} \\ p_{2i+1}(t) &= p_1(t - 2ih), & i = 1, 2, \dots, n/2 - 1, \end{aligned}$$

and

$$p_{2i}(t) = p_2(t - 2(i - 1)h), \quad i = 2, 3, \dots, n/2.$$

Define also  $\bar{\sigma}(t) = \sum_{i=1}^n \sigma(ih)p_i(t)$ . It is true that

- (i)  $\bar{\sigma}(t)$  is a polynomial of degree two on  $[ih, (i + 2)h]$ , for  $i = 0, 2, 4, \dots, n - 2$ .
- (ii)  $\bar{\sigma}(t) = \sigma(t)$  at  $t = ih$ , for  $i = 0, 1, 2, \dots, n$ .

$$(6) \quad \text{(iii)} \quad \sigma(t) = \bar{\sigma}(t) + O(h^3) = \sum_{i=1}^n \sigma_i p_i(t) + O(h^3),$$

where  $\sigma_i = \sigma(ih)$  for  $i = 1, 2, \dots, n$ .

Using the approximation Eq. (6) in Eq. (4), we get

$$(7) \quad \sum_{k=1}^n \sigma_k \int_0^d p_k(t) \log |z - \zeta(t)| dt = g(z) + O(h^3).$$

The function  $g(z) = -\ln |z - z_0|$  for  $z \in L$ . Thus, we can evaluate Eq. (7) at the points  $z = \zeta(ih)$  for  $i = 1, 2, \dots, n$ , and we will get  $n$  linear equations with constant coefficients for the variables  $\sigma_1, \sigma_2, \dots, \sigma_n$ . Set  $A = (a_{ik})$  and  $B = (b_i)$ , where

$$a_{ik} = \int_0^d p_k(t) \log |\zeta(ih) - \zeta(t)| dt, \quad \text{for } i, k = 1, 2, \dots, n,$$

$$b_i = -\log |\zeta(ih) - z_0|, \quad \text{for } i = 1, 2, \dots, n.$$

With this notation, Eq. (7) leads to the linear system  $A\delta = B + O(h^3)$ , where  $O(h^3)$  is a vector, with each component bounded by  $O(h^3)$ , and  $\delta = (\sigma_1, \sigma_2, \dots, \sigma_n)^T$ .

The matrix equation we actually solve is

$$(8) \quad \tilde{A}\tau = B,$$

where the elements of  $\tilde{A}$  are approximations to those of  $A$ . Using our representation for the  $p_k(t)$ , it is evident that to compute  $A$  it is sufficient to evaluate integrals of the form

$$(9) \quad \int_{(i-1)h}^{ih} t^j \log |z - \zeta(t)| dt$$

for  $i = 1, 2, \dots, n$  and  $j = 0, 1, 2$ . The  $\tilde{a}_{ij}$  are the result of approximating the integrals (9). For each fixed  $x, y$  and  $i$ , we approximate  $|z - \zeta(t)|$  by a polynomial  $q(t)$  of degree two on  $((i-1)h, ih)$ . We choose  $q(t)$  so that

$$q(t) = |z - \zeta(t)|^2 \quad \text{for } t = (i-1)h, (i - \frac{1}{2})h, ih.$$

Then

$$\int_{(i-1)h}^{ih} t^j \log |z - \zeta(t)| dt \approx \frac{1}{2} \int_{(i-1)h}^{ih} t^j \log [q(t)] dt.$$

The integrals on the right-hand side above can be evaluated explicitly. For certain special cases, for instance when  $|z - \zeta(t)| = 0$  on  $[(i-1)h, ih]$ , the treatment is slightly different in that a higher order polynomial is used.

We then solve the matrix equation  $\tilde{A}\tau = B$  for the vector  $\tau = (\tau_1, \tau_2, \dots, \tau_n)^T$  and use this as an approximation to  $\delta$ . Then

$$\|\delta - \tau\| \leq \|(A^{-1} - \tilde{A}^{-1})B\| + \|A^{-1}\| O(h^3).$$

We have found by experience on numerous problems that the error due to  $A^{-1} - \tilde{A}^{-1}$  seldom if ever dominates the  $\|A^{-1}\|O(h^3)$  term. Another analysis [2] strongly indicates that  $\|A^{-1}\| \leq O(1/h)$ .

Once  $\sigma(t)$  has been computed, we may calculate  $g(z)$  and  $h(z)$ .

$$g(z) = \int_0^d \sigma(t) \log |z - \zeta(t)| dt$$

$$\approx \sum_{k=1}^n \tau_k \int_0^d p_k(t) \log |z - \zeta(t)| dt.$$

These integrals are approximated as described above. The calculation of  $h(z)$  is more difficult. Using integration by parts, and approximations similar to those above, we are led to integrals of the form

$$\int_{(i-1)h}^{ih} t^j \arg(z - \zeta(t)) dt,$$

where  $i = 1, 2, \dots, n$  and  $j = 0, 1, 2, 3$ . The evaluation of these integrals is discussed in detail in [1].

The method set forth by Symm in [7] uses piecewise constant functions in Eq. (6) and evaluates the integrals in Eq. (7) by using Simpson's rule of integration.

**3. Tests.** A FORTRAN IV version of the algorithm described has been coded to run on our CDC 6600 and CDC 7600. This program is more or less machine independent, has flexible input, and is general enough to handle a large class of problems. It is a modification of a program described in [1] which has been in use for a few years. A limited number of copies of this deck and a user's guide are available from the authors. Using this code, we have computed some approximate conformal maps for several regions, including some used in [7]. Since our technique is an extension of the method used there, it is appropriate to compare our results with those. All of the regions selected for test have substantial symmetry. We have elected to ignore this symmetry in our code in order to give utmost flexibility. Taking advantage of symmetry ought to enhance the accuracy by reducing the volume of computation.

Symm has pointed out [8] that the maximum errors occur on the boundary of the region being mapped. Since points on the boundary have image points on the unit circle, it is easy to check the error in the modulus of an arbitrary boundary point. The data points themselves are constrained by the defining equations to be mapped onto  $|w| = 1$ , hence we check for modulus error at points midway between each of the data points. The columns labeled ERR-MOD contain the maxima of the quantities  $||w| - 1|$  at these intermediate points. Computing the error in the argument is more difficult. Symm provides an estimate of this in [8], denoted  $E_A$ . Our experience has indicated that as the region becomes less circular and more elongated, errors, particularly those in the argument, increase in a monotonic way. Since the numbers  $E_A$  provided by Symm did not have this property, we considered them somewhat unreliable and decided to use an alternative technique. The columns labeled ERR-ARG represent the maximum difference in the argument at the data points between two computations, the second corresponding to the largest number of data points used for the domain in question. It is reasonable to examine the argument at the data points rather than the intermediate points, since the argument is not constrained in any way by Eq. (1). This procedure does yield the monotonicity we expect. In certain cases, analytic expressions for the conformal maps are available. It is then possible to compute the absolute errors in the argument exactly. These numbers compare extremely well with the approximate errors ERR-ARG described above, and constitute our main justification for this approach.

Each of our test regions has its center point mapped into the origin. In what follows,  $h$  and  $n$  will have the same meaning as in Section 2.

*Example 1. Oval of Cassini.* This curve is defined by

$$[(x + 1)^2 + y^2][(x - 1)^2 + y^2] = \alpha^4, \quad \alpha > 1.$$

For  $\alpha$  near 1, the curve is elongated and nonconvex, becoming more circular as  $\alpha$  increases. For  $\alpha = 1.06$ , the width to height ratio is about 5. Points are distributed uniformly on the entire boundary. The exact mapping is given by

$$f(z) = \alpha z / (\alpha^4 - 1 + z^2)^{1/2},$$

and we use this to compute errors in the argument.

TABLE I  
*Oval of Cassini*

$\alpha$	$n$	$h$	ERR-MOD	ERR-ARG
1.06	65	.11	$2 \times 10^{-3}$	$2 \times 10^{-3}$
1.06	129	.06	$1 \times 10^{-4}$	$2 \times 10^{-4}$
1.8	33	.34	$1 \times 10^{-4}$	$1 \times 10^{-4}$
1.8	65	.18	$1 \times 10^{-5}$	$1 \times 10^{-5}$
1.8	129	.09	$6 \times 10^{-7}$	$7 \times 10^{-7}$

The maximum error occurs at or near  $x = 0$ . The errors near  $y = 0$  are smaller by a factor of 1/100. The comparison with Symm must be made carefully, since his data points are for the most part distributed uniformly with respect to  $x$  rather than  $t$ . As far as we can determine, errors in the modulus are from one to four orders of magnitude better than those in [7]. We should emphasize that our distribution of points is poor.

For  $\alpha = 1.06$ ,  $n = 65$ , Table II indicates errors for points inside the curve.

TABLE II  
*Oval of Cassini*

$\alpha = 1.06$	$n = 65$	ERRORS IN	
		MODULUS	ARGUMENT
$x$	$y$		
1.4	0.	$6 \times 10^{-7}$	$3 \times 10^{-5}$
1.26	0.	$6 \times 10^{-7}$	$3 \times 10^{-5}$
1.12	0.	$1 \times 10^{-6}$	$3 \times 10^{-5}$
0.98	0.	$2 \times 10^{-6}$	$3 \times 10^{-5}$
0.84	0.	$2 \times 10^{-6}$	$3 \times 10^{-5}$
0.7	0.	$4 \times 10^{-6}$	$3 \times 10^{-5}$
0.56	0.	$7 \times 10^{-6}$	$3 \times 10^{-5}$
0.42	0.	$2 \times 10^{-5}$	$4 \times 10^{-5}$
0.28	0.	$5 \times 10^{-5}$	$4 \times 10^{-5}$
0.14	0.	$1 \times 10^{-4}$	$4 \times 10^{-5}$

*Example 2. Rectangle.*  $-1 \leq x \leq +1$ ,  $-\alpha \leq y \leq \alpha$ .

The case  $\alpha = 1$  was computed exactly by the use of elliptic integrals. In the other cases, we use a comparison with the most accurate computed values. Points are uniformly spaced on each side, with  $n/4$  points per side. See Table III.

TABLE III  
*Rectangle*

$\alpha$	$n$	ERR-MOD	ERR-ARG
0.1	516	$4 \times 10^{-5}$	—
	260	$6 \times 10^{-4}$	$7 \times 10^{-4}$
	132	$5 \times 10^{-3}$	$6 \times 10^{-3}$
	68	$1 \times 10^{-2}$	$1 \times 10^{-2}$
	36	$6 \times 10^{-2}$	$5 \times 10^{-2}$
0.2	516	$3 \times 10^{-6}$	—
	260	$4 \times 10^{-5}$	$4 \times 10^{-5}$
	132	$5 \times 10^{-4}$	$6 \times 10^{-4}$
	68	$5 \times 10^{-3}$	$7 \times 10^{-3}$
	36	$1 \times 10^{-2}$	$2 \times 10^{-2}$
0.4	260	$3 \times 10^{-6}$	—
	132	$4 \times 10^{-5}$	$4 \times 10^{-5}$
	68	$6 \times 10^{-4}$	$7 \times 10^{-4}$
	36	$5 \times 10^{-3}$	$1 \times 10^{-2}$
0.5	260	$1 \times 10^{-6}$	—
	132	$2 \times 10^{-5}$	$2 \times 10^{-5}$
	68	$2 \times 10^{-4}$	$2 \times 10^{-4}$
	36	$2 \times 10^{-3}$	$3 \times 10^{-3}$
0.8	260	$2 \times 10^{-7}$	—
	132	$3 \times 10^{-6}$	$5 \times 10^{-6}$
	68	$4 \times 10^{-5}$	$8 \times 10^{-5}$
	36	$6 \times 10^{-4}$	$3 \times 10^{-3}$
1.0	260	$9 \times 10^{-8}$	—
	132	$1 \times 10^{-6}$	$1 \times 10^{-6}$
	68	$2 \times 10^{-5}$	$3 \times 10^{-5}$
	36	$2 \times 10^{-4}$	$1 \times 10^{-3}$

Both ERR-MOD and ERR-ARG are monotonic with respect to  $n$  and  $\alpha$  for  $\alpha \leq 1$ . Errors in the modulus are from one to two orders of magnitude better than in [7]. It should be noted that for small  $\alpha$  the distribution of boundary points is poor. This is true for most of the examples. The only reason for using the given distribution of points is to compare with [7]. Our experience shows that a good rule of thumb

for the distribution of boundary points is to keep the distance between successive boundary points and the distance from the boundary points to the center in a nearly constant ratio. Thus, for  $\alpha$  small, we want more points near the centers of the longer two sides and fewer points on the shorter two sides. This can be done by dividing the boundary into sections as mentioned in the paragraph following Eq. (5). We ran the problem with  $\alpha = 0.1$  again, using a particularly simple redistribution of the boundary points. For a fixed number of points, the errors decreased by about 1/50. Using an optimal distribution of points, one would get more accuracy.

*Example 3. Ellipse.*  $x^2/\alpha^2 + y^2 = 1$ . The data points are uniformly distributed on the boundary of the ellipse. See Table IV.

TABLE IV

*Ellipse*

$\alpha$	$n$	ERR-MOD	ERR-ARG
1.25	257	$3 \times 10^{-8}$	—
	129	$3 \times 10^{-7}$	$2 \times 10^{-7}$
	65	$4 \times 10^{-6}$	$4 \times 10^{-6}$
	33	$5 \times 10^{-5}$	$4 \times 10^{-5}$
2.5	257	$3 \times 10^{-7}$	—
	129	$4 \times 10^{-6}$	$5 \times 10^{-6}$
	65	$5 \times 10^{-5}$	$6 \times 10^{-5}$
	33	$7 \times 10^{-4}$	$9 \times 10^{-4}$
5.0	257	$4 \times 10^{-6}$	—
	129	$4 \times 10^{-5}$	$5 \times 10^{-5}$
	65	$7 \times 10^{-4}$	$2 \times 10^{-3}$
	33	$6 \times 10^{-3}$	$5 \times 10^{-3}$
10	257	$4 \times 10^{-5}$	—
	129	$6 \times 10^{-4}$	$6 \times 10^{-4}$
	65	$5 \times 10^{-3}$	$6 \times 10^{-3}$
	33	$1 \times 10^{-2}$	$6 \times 10^{-2}$
20	257	$5 \times 10^{-4}$	—
	129	$5 \times 10^{-3}$	$6 \times 10^{-3}$
	65	$1 \times 10^{-2}$	$3 \times 10^{-3}$
	33	$7 \times 10^{-2}$	$5 \times 10^{-2}$

Again, we note monotonicity with respect to  $n$  and  $\alpha$  for  $\alpha \geq 1$ , with maximum error near the center of the side intersected by the minor axis. Improvements over [7] are from one to three orders of magnitude, with the least improvement for  $\alpha = 20$ .

*Example 4. Isosceles Triangle.* The corners of the triangle are at  $(0, 1)$ ,  $(2, -1)$ ,  $(-2, -1)$ , and  $(0, 0)$  is mapped into the origin of the unit circle. There are equal numbers of points on each side.

TABLE V  
Triangle

$n$	ERR-MOD	ERR-ARG
65	$1 \times 10^{-6}$	—
33	$2 \times 10^{-5}$	$6 \times 10^{-5}$
17	$2 \times 10^{-4}$	$9 \times 10^{-4}$

4. Timing. There are three operations that are important as far as timing is concerned: (i) generating the matrix  $\tilde{A}$  of Eq. (8), (ii) solving the matrix Eq. (8), and (iii) evaluating the function  $f(z)$  at a given point. The time required for (i) is proportional to  $n^2$  and is about 0.85 sec\*\* for  $n = 200$ . The time required for (ii) is proportional to  $n^3$  and is about 2 sec for  $n = 200$ . The time required for (iii) is proportional to  $n$  and is 0.016 sec for  $n = 200$ .

University of California  
Los Alamos Scientific Laboratory  
Los Alamos, New Mexico 87544

1. J. HAYES, *Four Computer Programs Using Green's Third Formula to Numerically Solve Laplace's Equation in Inhomogeneous Media*, Los Alamos Scientific Laboratory Report, LA-4423, April 1970.

2. J. HAYES & R. KELLNER, *The Eigenvalue Problem for a Pair of Coupled Integral Equations Arising in the Numerical Solution of Laplace's Equation*, Los Alamos Scientific Laboratory Report, LA-DC-12009, October 1970.

3. M. A. JASWON, "Integral equation methods in potential theory. I," *Proc. Roy. Soc. Ser. A*, v. 275, 1963, pp. 23-32. MR 27 #4034.

4. M. MAITI, "A note on the integral equation methods in potential theory," *Quart. Appl. Math.*, v. 25, 1968, pp. 480-484. MR 37 #447.

5. N. I. MUSKHELISHVILI, *Singular Integral Equations. Boundary Problems of Function Theory and their Application to Mathematical Physics*, OGIZ, Moscow, 1946; English transl., Noordhoff, Groningen, 1953. MR 8, 586; MR 15, 434.

6. PHILIP RABINOWITZ, "Numerical experiments in conformal mapping by the method of orthonormal polynomials," *J. Assoc. Comput. Mach.*, v. 13, 1966, pp. 296-303.

7. G. T. SYMM, "An integral equation method in conformal mapping," *Numer. Math.*, v. 9, 1966, pp. 250-258. MR 34 #7056.

8. G. T. SYMM, "Numerical mapping of exterior domains," *Numer. Math.*, v. 10, 1967, pp. 437-445. MR 36 #3525.

9. G. T. SYMM, "Conformal mapping of doubly-connected domains," *Numer. Math.*, v. 13, 1969, pp. 448-457. MR 40 #3736.

\*\* All times given here are for the CDC 7600.



to go to a more exact representation of this section of  $L_f$ . Thus, there are three ways to approximate a boundary section.

11. How many mesh points are to be required on this section? See Fig. 1.

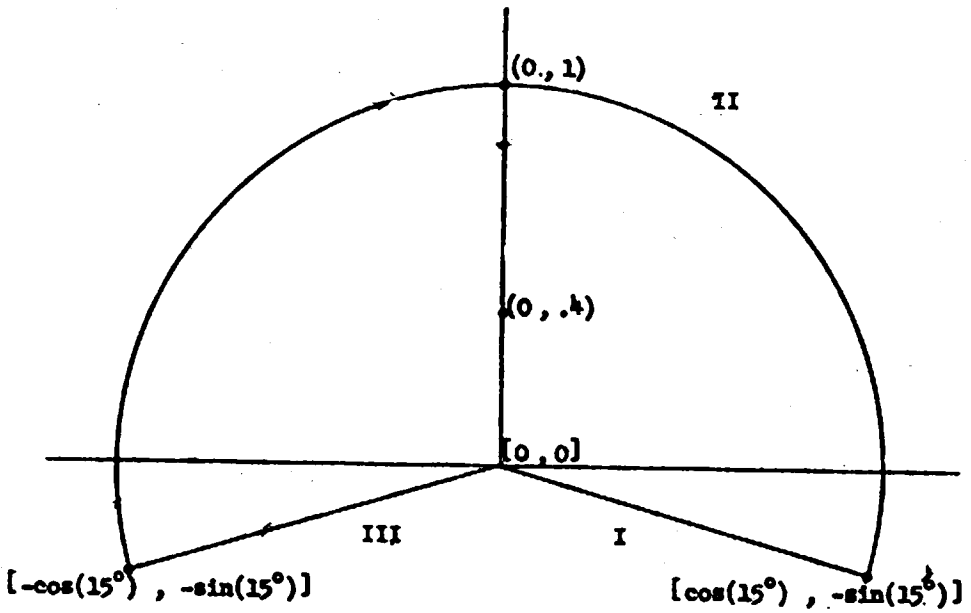


Fig. 1. Example

In the example there are three sections naturally imposed by the geometry. Sections I and III are exactly represented by line segments and Section II by a circular arc.

On the data cards a line segment is defined by giving its two endpoints. A circular arc is defined by giving its two endpoints and any interior point on the arc. This defines the type of approximation used for that boundary section.

The data card is seven fields long. Each of the first six fields is ten characters in length and is read with an E10.0 format. If a card is used to describe a line segment, the first four fields are used for the  $(x,y)$  coordinates of the beginning and end of the segment. The fifth and sixth fields must be blank. If a card is used to describe a circular arc, the first six fields are used to give the  $(x,y)$  coordinates of the beginning, interior, and endpoint of the arc. The endpoint of section  $k$  (corresponding to data card  $k$ ) must agree with the initial point of section  $k+1$  (data card  $k+1$ ), and the endpoint of the last boundary section must agree with the initial point of the first boundary section.

The input for the curve  $L$  must have a positive (counterclockwise) orientation. The same orientation suffices for interior or exterior problems. The direction of the curve  $L$  is determined by the order of the points on the data cards. The direction for a line segment or circular arc is from the initial point to the endpoint. It has been our experience that errors in the orientation of  $L$  are difficult to detect.

The seventh field on the data card determines the number of mesh points  $n_s$  per section. These points are uniformly distributed with respect to arc

length along the approximating curve. This number is read with an I3 format in columns 61-63. It must be odd and equal to three or more.

One possible set of data cards for the curve given in Fig. 1 is

80 COLUMN ENTRY																		
PROGRAMMER										PROBLEM				DATE		PAGE		OF
01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19
0.																		
0.6392																		
0.6392																		

Note the positive, counterclockwise orientation.

As a convenience, one can leave the first two fields blank on a data card used to describe a line segment or circular arc if the previous card was either a line segment or circular arc. An equivalent set of data cards for Fig. 1 is then

80 COLUMN ENTRY																		
PROGRAMMER										PROBLEM				DATE		PAGE		OF
01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19

The next two sets of data cards represent input for the rectangle given in Fig. 2. Here the natural sections are line segments, but in the second data set below, the long sides of the rectangle have themselves been divided into three sections. This allows for a different mesh spacing, if desired, on each section.



Occasionally, it is not satisfactory to represent a section of  $L$  by either a line segment or a circular arc. In that case "generalized" boundary input for that section is available in the form of a user provided subroutine BDRY. When line segments and circular arcs are the only type of input used, BDRY appears as a program that is referenced, but not loaded. This is normal and will cause no difficulty.

Suppose that the user decides to use "generalized" boundary data on the  $i$ th and  $j$ th section. He must then write a subroutine of the form

SUBROUTINE BDRY(K,T,X,Y,XP,YP,V).

Input to this subroutine will be  $K$  and  $T$ . Depending on when it is called  $K$  will take on the value  $I$  or  $J$ .  $T$  will be a real number ranging from zero to the length of the  $K$ th subinterval. The pair of numbers  $K, T$  then uniquely defines a point.  $T$  units in the positive direction along the  $K$ th subinterval. If we denote that point by  $(X(T), Y(T))$ , the output of BDRY is

$$\begin{aligned} X &: X(T) \\ Y &: Y(T) \\ XP &: \frac{dX(T)}{dT} \\ YP &: \frac{dY(T)}{dT} \\ V &= \frac{1}{2} \left[ XP \cdot \frac{d^2Y(T)}{dT^2} - YP \cdot \frac{d^2X(T)}{dT^2} \right] \end{aligned}$$

To keep the orientation correct  $(YP, -XP)$  must be the exterior normal at  $(X,Y)$ .

To define the  $K$ th section as being given by "generalized" boundary data, the  $K$ th data card must have in columns 51-60 the arc length  $d$  of that section. Columns 1-50 must be blank and the number of mesh points appears as before in columns 61-63.

It is always possible to approximate sections of  $L$  by curves other than lines or circular arcs. In that case, the subroutine BDRY would output the parameters of the approximating curve, and  $d$  would be its length.

By way of illustration we give the input data and BDRY for the example of Fig. 1.

C ← For comment		PROBLEM	DATE	PAGE	OF	PROGRAMMER
Statement number		FORTRAN STATEMENT				Modification
1		SUBROUTINE BDRY(I,T,X,Y,W,V)				72,73
		DATA PI /3.1415926535897/				
		GOTO(1,2,3),I				
		X = T * COS(PI/12.)				
		Y = -PI * SIN(PI/12.)				
		W = X/T				
		V = Y/T				
		V = 0.				
		RETURN				
2		X = COS(T - PI/12.)				
		Y = SIN(T - PI/12.)				
		W = -T				
		V = X				
		V = .5				
		RETURN				
3		X = -COS(PI/12.) * (1.-T)				
		Y = -SIN(PI/12.) * (1.-T)				
		W = -2/(1.-T)				
		V = -2/(1.-T)				
		V = 0.				
		RETURN				
		END				

80 COLUMN ENTRY															
14	24	34	44	54	64	74	84	94	04	14	24	34	44	54	64

It is, of course, necessary for the user to separately compute the lengths of the "generalized" boundary sections..

Storage Requirements:

The present CMP requires about  $37000_8$  words of central memory. This includes all the subroutines, associated system programs, and internal dimensioned variables, but does not include the major storage needed for the matrix  $\tilde{A}$  of Eq. (8).<sup>\*</sup> If there are a total of  $n$  mesh points on the boundary then  $n^2 + 1500_{10}$  words additional are needed. The CMP has been written to allow utmost flexibility in selecting the site for this storage. All the references to this data are through a subroutine ECRD with formal parameters

ECRD(A,M,L,K)

and additional entry point ECWR .

If the user operates on the LASL system with Extended Core Storage (ECS) available, this subroutine is generated by the system and he need not be concerned with it. His only requirement is to request  $n^2 + 1500_{10}$  words of ECS and observe the restriction that  $n < 630$  unless present dimension statements are changed.

If  $n < 200$ , the entire problem can be run in  $200000_8$  words of central memory. This is easily done by inserting into the deck the subroutine ECRD, a copy of which is included in the listings. In that subroutine the dimension of the local variable A must be at least  $n^2 + 1500_{10}$ . Problems with smaller  $n$  will run in even less central memory after the appropriate adjustments have been made to this dimension statement.

Users whose facilities prohibit the use of ECS or even  $200000_8$  words

<sup>\*</sup>All referenced equations are in "An Improved Method for Numerical Conformal Mapping."

of central memory will find it possible to rewrite their own version of subroutine ECRD to store  $\tilde{\lambda}$  on tape or disk. To run a given problem the routine ECRD will need  $n^2 + 1500$  words of some type of available storage. The entry point ECR(A,M,L,K) should store L words from central memory starting at location A in available storage starting at location M. The entry point ECRD(A,M,L,K) should read L words from available storage starting at location M and store them starting at location A. The variable K is only used to make the subroutine compatible with the LASL system. K should merely be set equal to 0 on each entry.

To effect the conformal mapping it is necessary to call SUBROUTINE CONFORM(XO,YO,K). The parameters (XO,YO) define the point that we wish to map into the origin of the unit circle. The input parameter K can take on values 0 or 1, and determines whether an interior or exterior problem is to be solved. For  $K = 1$ , an exterior problem, XO and YO are not used in the calculation. The call to this subroutine triggers the coding that will read in the data cards describing D (discussed above) and perform the calculations necessary to solve the integral equation of the first kind for the potential function  $\sigma(t)$ . The input data will be printed so that visual checks can be made. The only other printed output is a few possible error messages based on some consistency checks that are made on the data. All the other immediately available output, of which there is a great deal, resides in a COMMON block that the user must refer to (see below).

After the call to CONFORM has been completed, the image of any desired points can be obtained by calling

SUBROUTINE FE(X,Y,R,T)



with input  $X, Y$ . The output,  $R, T$  contains the image of point  $(X, Y)$  under the conformal map just executed. The image point is given in polar coordinates  $(R, \theta)$ . This subroutine may be called as many times as necessary.

Two restrictions on the use of  $FN$  are

(i) For interior problems, points  $(X, Y)$  outside  $D + L$  should not be given as input. For exterior problems points inside  $D + L$  should not be given.

(ii) Because of coding peculiarities, not all points on  $L$  can be given as input. The only completely acceptable input points on  $L$  are those appearing as mesh points on the boundary. Points halfway (in arc length) between the mesh points are also acceptable, but the value of  $T$  returned will be wrong. It is suggested that users wishing the images of boundary points other than mesh points perform quadratic interpolation through the mesh points. Errors in  $R$  and  $T$  are usually greatest for points near the boundary curve and least for points near the center of  $D$ . Exceptions to this are mesh points. Unless the nearest boundary section is a straight line, values of  $R$  and  $T$  obtained within  $h/4$  of the boundary should not be trusted.

There are four labelled common blocks used to store intermediate data by the CMP. They are called  $CONF$ ,  $MODE$ ,  $ZLAP$ , and  $LTT$ . The card images are in Fig. 3.

```
COMMON /MODE/ IN,  
COMMON /CONF/ P(30),X0,Y0,CD(30),C(30),KV(10),ND(10),OK(30),  
1 X(30),X(30),Y(30),Y(30),X(30),X(30),Y(30),Y(30),X(30),Y(30),Y(30),X(30)  
2 CM  
COMMON /ZLAP/ V,DI(1),M,MS,MSO2,MO(1),M,ME,ZZ,ZZ,DP,AR,MS,1,MS,PS  
1 BQ,ZE,M,M  
COMMON /LTT/ BC(10)
```

Fig. 3. Card Images of Common Statements

MODE contains only the variable IN where the third parameter in the call statement to CONFORM is stored. Thus IN = 0 for interior problems, and IN = 1 for exterior problems. The labelled common ZLAP is used to transmit information between subroutines. A user would probably never need to interact with these variables. The labelled common CONF contains the variables of most interest to the user. The variable NDC is used to store the number of boundary sections for a given problem. For instance, for the example in Fig. 1 NDC = 3. The dimensioned variable KV is used to give a running total of the number of approximation points on the boundary sections. We have  $KV(1) = 0$  and  $KV(J+1) - KV(J)$  is equal to the number of approximation points on the  $J$ th boundary section. For the input given for the example in Fig. 1 we have  $KV(1) = 0$ ,  $KV(2) = 31$ ,  $KV(3) = 116$ , and  $KV(4) = 147$ . The variable N is the total number of approximation points for a given problem. Obviously  $N = KV(NDC+1)$ . The dimensioned variable HD is used to store the distance between approximation points along the curve L for each of the boundary sections given as input. For the example in Fig. 1 we have  $HD(1) = HD(3) = 1/30$  and  $HD(2) = \pi(1 + 1/6)/84$ . The variables X0 and Y0 are the coordinates of the inverse image of the origin for an interior problem. For an exterior problem these two numbers are set equal to zero. The dimensioned variables X(I), Y(I),  $I = 1, 2, \dots, N$  are coordinates of the approximation points. For any given boundary section there are two approximation points at the ends of the boundary section and the other approximation points for that boundary are equally spaced with respect to arc length along the section. Because we define our points this way, it is true that  $[X(KV(J+1)), Y(KV(J+1))]$

=  $[X(KV(J + 1) + 1), Y(KV(J + 1) + 1)]$  for  $J = 2, 3, \dots, NDC-1$  and  $[X(1), Y(1)] = [X(N), Y(N)]$ . It may seem wasteful to the user to have two sets of coordinates for the same point, but it makes the programming logic much simpler for certain cases. For example, we must allow for discontinuities in the unit normal for  $L$  at corners and we must allow for discontinuities in the values of  $\sigma$  at corners. By defining our approximation points in the above manner, it makes the indexing much easier in these cases.

For  $I = 1, 2, \dots, N$ , we have that  $[XN(I), -YN(I)]$  is the unit normal vector for  $L$  at the point  $[X(I), Y(I)]$ . The approximate value of  $\sigma$  at the point  $(X(I), Y(I))$  is stored in the dimensioned variable  $F(I)$  for  $I = 1, 2, \dots, N$ . For exterior problems  $F(N + 1)$  contains the logarithm of the approximate transfinite diameter.  $F$  is also used for intermediate computations in CONFORM.

For  $I = 1, 2, \dots, N$  the variable  $G(I)$  contains the value of  $\frac{1}{2} \left[ \frac{dx}{dt} \frac{dy}{dt} - \frac{dy}{dt} \frac{dx}{dt} \right]$  at the point  $[X(I), Y(I)]$ . Between each consecutive pair of approximation points on each boundary section we store the location of an intermediate point on the boundary  $L$ . The coordinates of the intermediate point are stored in the dimensioned variables  $[XI, YI]$ , and the corresponding unit normal is stored in  $[XIN, YIN]$ . Thus the point midway on the curve  $L$  between the points  $[X(1), Y(1)]$  and  $[X(2), Y(2)]$  has its coordinates stored in  $[XI(2), YI(2)]$  and the unit normal at that point is  $[XIN(2), -YIN(2)]$ . Because of the manner in which we store our approximation points, the points  $[XI(KV(J) + 1), YI(KV(J) + 1)]$  for  $J = 1, 2, \dots, NDC$  are not defined even though the corresponding points  $[X(KV(J) + 1), Y(KV(J) + 1)]$  are well defined. The two dimensioned variables  $CE$  and  $CD$  are used to transmit information between subroutines. The  $N$  different integrals that appear in Eq. (7) are stored in  $CE$ . The corresponding  $N$  integrals for  $b(s)$  are stored in  $CD$ .

The variables CH and CD are equivalent to the variables A and A2 respectively and are also used for intermediate storage when solving Eq. (6).

The labelled common LIT contains only the dimensioned variable DC(J) for  $J = 1, 2, \dots, NDC$  contains the arc length of the Jth boundary section.

CONFORMAL MAPPING 1  
 HAYES, KAHANER, KELLNER

```

PROGRAM NEME (INPUT,OUTPUT)
C TEST FOR CONFORMAL MAPPING PACKAGE
C COMMON /CONF/ F(630),XO,YO,CD(630),CN(630),KV(270),MD(210),G(630),
1 X(630),KN(630),Y(630),YN(630),XI(630),XJN(630),Y(630),YJN(630),ND
2 C,N
C DEFINE POINTS TO BE MAPPED INTO THE CENTER OF THE CIRCLE
XO=YO=0.
C SOLVE THE INTEGRAL EQUATION FOR SIGMA. FOR THIS PROBLEM, THE
C BOUNDARY OF THE ELLIPSE IS DEFINED BY A USER SUPPLIED SUBROUTINE
CALL CONFORM (XO,YO,1)
C PRINT SIGMA
PRINT 5
C PRINT 6, (F(M),M=1,N)
D=EXP(F(N+1))
C PRINT TRANSFINITE DIAMETER, STORED IN F(N+1).
C THIS IS AN EXTERIOR PROBLEM
PRINT 4, D
PRINT 5
C FIND THE IMAGES OF VARIOUS BOUNDARY POINTS
DO 1 M=1,ND
M1=KV(M)+2
M2=KV(M+1)
C FIND THE IMAGES OF BOUNDARY MESH POINTS
CALL FN (X(M1-1),Y(M1-1),R,TH)
CALL OUTP (R,TH)
DO 1 K=M1,M2
C FIND THE IMAGES OF POINTS INTERMEDIATE TO MESH POINTS
CALL FN (X(K),Y(K),R,TH)
TH=12345.
CALL OUTP (R,TH)
CALL FN (X(K),Y(K),R,TH)
CALL OUTP (R,TH)
1 THE FOLLOWING CALL CLEARS THE OUTPUT BUFFER
C CALL OUTC (R,TH)
C THE FOLLOWING DO LOOP COMPUTES THE CONFORMAL MAP OF THREE
C SEPARATE PROBLEMS.
C 1 A RECTANGLE WITH THE SAME NUMBER OF POINTS PER SIDE
C 2 SAME RECTANGLE WITH A DIFFERENT DISTRIBUTION OF BOUNDARY
C POINTS
C 3 A CIRCLE WITH A 160 DEGREE SECTOR CUT OUT.
C THE FIRST TWO USE STRAIGHT LINE BOUNDARIES. THE LAST USES
C STRAIGHT LINE AND CIRCULAR ARC BOUNDARIES
DO 3 I=1,3
IF (I.EQ.3) YO=4
CALL CONFORM (XO,YO,0)
PRINT 5
PRINT 6, (F(M),M=1,N)
PRINT 5
DO 2 M=1,ND
M1=KV(M)+2
M2=KV(M+1)
CALL FN (X(M1-1),Y(M1-1),R,TH)
CALL OUTP (R,TH)
DO 2 K=M1,M2
CALL FN (X(K),Y(K),R,TH)
TH=12345.
CALL OUTP (R,TH)
CALL FN (X(K),Y(K),R,TH)
CALL OUTP (R,TH)
2 CALL OUTC (R,TH)
3 CONTINUE
C
4 FORMAT ('0*.TRANSFINITE DIAMETER = *.E15.8)
5 FORMAT ('0*')
6 FORMAT ('1H 1P9E15.7)
END

```

NEME 10  
 NEME 20  
 NEME 30  
 NEME 40  
 NEME 50  
 NEME 60  
 NEME 70  
 NEME 80  
 NEME 90  
 NEME 100  
 NEME 110  
 NEME 120  
 NEME 130  
 NEME 140  
 NEME 150  
 NEME 160  
 NEME 170  
 NEME 180  
 NEME 190  
 NEME 200  
 NEME 210  
 NEME 220  
 NEME 230  
 NEME 240  
 NEME 250  
 NEME 260  
 NEME 270  
 NEME 280  
 NEME 290  
 NEME 300  
 NEME 310  
 NEME 320  
 NEME 330  
 NEME 340  
 NEME 350  
 NEME 360  
 NEME 370  
 NEME 380  
 NEME 390  
 NEME 400  
 NEME 410  
 NEME 420  
 NEME 430  
 NEME 440  
 NEME 450  
 NEME 460  
 NEME 470  
 NEME 480  
 NEME 490  
 NEME 500  
 NEME 510  
 NEME 520  
 NEME 530  
 NEME 540  
 NEME 550  
 NEME 560  
 NEME 570  
 NEME 580  
 NEME 590  
 NEME 600  
 NEME 610  
 NEME 620  
 NEME 630  
 NEME 640  
 NEME 650  
 NEME 660  
 NEME 670

CONFORMAL MAPPING 2  
 HAYES, KAHANER, KELLNER

	SUBROUTINE OUTP (R,TH)	OUTP 10
	OUTPUT SUBROUTINE FOR TEST PROGRAM	OUTP 20
	DIMENSION DAT(10), BCDDAT(10)	OUTP 30
	DATA NUMB/0	OUTP 40
	DAT(NUMB+1)=R	OUTP 50
	DAT(NUMB+2)=TH	OUTP 60
	NUMB=NUMB+2	OUTP 70
	IF (NUMB.NE.10) RETURN	OUTP 80
1	NUMB=0	OUTP 90
	DO 2 I=1,10	OUTP 100
	ENCODE (10.5,BCDDAT(I))DAT(I)	OUTP 110
2	IF (DAT(I).EQ.12345.) BCDDAT(I)=1H	OUTP 120
	CONTINUE	OUTP 130
	PRINT 4, (BCDDAT(I),I=1,10)	OUTP 140
	RETURN	OUTP 150
	ENTRY DUTC	OUTP 160
3	IF (NUMB.EQ.10) GO TO 1	OUTP 170
	DAT(NUMB+1)=12345.	OUTP 180
	DAT(NUMB+2)=12345.	OUTP 190
	NUMB=NUMB+2	OUTP 200
	GO TO 3	OUTP 210
C		OUTP 220
4	FORMAT (10(2X,A10))	OUTP 230
5	FORMAT (F10.7)	OUTP 240
	END	OUTP 250
	SUBROUTINE BDRY (M,S,X,Y,XP,YP,V)	BDRY 10
C	GENERALIZED BOUNDARY ROUTINE FOR CONFORMAL MAP OF ELLIPSE	BDRY 20
C	DIFFERENT SUBSECTIONS ARE TO UTILIZE SYMMETRY ON CURVE TO	BDRY 30
C	AVOID REPEATING CALCULATIONS	BDRY 40
C	ELLI IS AN ELLIPTIC INTEGRAL. SUBROUTINE NEEDED IN THE CALCULATIONS	BDRY 50
C	FOR THE BOUNDARY OF THE ELLIPSE	BDRY 60
	DATA IFLAG/0	BDRY 70
	IF (IFLAG.EQ.1) GO TO 1	BDRY 80
	AA=5.	BDRY 90
	EK=SQRT(24.J/5.	BDRY 100
	CALL ELLI (1.570796326,EK,Z1,Z2)	BDRY 110
	BLO4=6.*Z2	BDRY 120
	IFLAG=1	BDRY 130
1	IF (S.GT.BLO4*1.00000001) GO TO 2	BDRY 140
	S1=S2=1.	BDRY 150
	T=S	BDRY 160
	GO TO 5	BDRY 170
2	IF (S.GT.BLO4*2.00000002) GO TO 3	BDRY 180
	S1=1.	BDRY 190
	S2=1.	BDRY 200
	T=2.00000002*BLO4 S	BDRY 210
	GO TO 5	BDRY 220
3	IF (S.GT.BLO4*3.00000003) GO TO 4	BDRY 230
	S1=1.	BDRY 240
	S2=1.	BDRY 250
	T=3.00000003*BLO4	BDRY 260
	GO TO 5	BDRY 270
4	S1=1.	BDRY 280
	S2=1.	BDRY 290
	T=4.00000004*BLO4 S	BDRY 300
5	TMI=0.	BDRY 310
	TMA=1.570796326	BDRY 320
	DO 6 J=1,40	BDRY 330
	ZI=.5*(TMA+TMI)	BDRY 340
	CALL ELLI (ZI,EK,Z,S)	BDRY 350
	IF (SI*AA.GE.T) TMA=ZI	BDRY 360
	IF (SI*AA.LE.T) TMI=ZI	BDRY 370
6	CONTINUE	BDRY 380
	X=AA*SIN(ZI)*S1	BDRY 390
	Y=SQRT(1.-(X/AA)**2)*S2	BDRY 400
	A2=Y*AA**2	BDRY 410
	XP=A2/SQRT(A2**2+X**2)	BDRY 420
	YP=X/SQRT(A2**2+X**2)	BDRY 430
	V=.5*AA**4/(X**2+A2**2)**1.5	BDRY 440
	F=0.	BDRY 450
	RETURN	BDRY 460
	END	BDRY 470

CONFORMAL MAPPING 3  
HAYES, KAHANER, KELLNER

```

SUBROUTINE ELLI (CHI,CAY,F,E)                                ELLI 10
THE FOLLOWING ROUTINE IS AN ELLIPTIC INTEGRAL ROUTINE USED ONLY IN  ELLI 20
THE CALCULATION OF THE BOUNDARY OF THE ELLIPSE                ELLI 30
DIMENSION MSG1(10), MSG2(10)                                ELLI 40
DATA MSG1/50HELLI, K0.GT. ONE          F-E=PHI              .5  ELLI 50
1  ONCELLI - K0.GT. ONE          F-E=PHI                    /    ELLI 60
DATA MSG2/50HELLI, K-1,PHI,GE,PI/2,SO F SET TO SIGN(PHI)*1.E+294.5 ELLI 70
1  ONCELLI - K-1, SO F SET TO 1.E+294                        /    ELLI 80
DATA P12,PI,TOP/172062207732504205518,3.14158265358979,17175067480 ELLI 90
1  3334471048/                                              ELLI 100
DATA EPS,EPS2/1.E-13,2.E+13/                                ELLI 110
PHI=CHI                                                       ELLI 120
IND=1                                                         ELLI 130
1  F=PHI                                                      ELLI 140
E=F                                                           ELLI 150
PSI=ABS(PHI)                                                 ELLI 160
IF ((PSI.LT.EPS).OR.(CAY.EQ.0)) RETURN                       ELLI 170
S1=CAY*CAY                                                   ELLI 180
RAD=1. S1                                                     ELLI 190
IF (RAD) 10,9,2                                             ELLI 200
2  ALPHA=1.                                                  ELLI 210
BETA=SQRT(RAD)                                              ELLI 220
S2=0.                                                        ELLI 230
PR2=1.                                                       ELLI 240
PWR2=1.                                                      ELLI 250
FINT=PSI*TOP                                                ELLI 260
NOUAD=INT(FINT)                                              ELLI 270
FINT=FINT-FLDAT(NOUAD)                                       ELLI 280
IF (AMIN1(FINT,1.-FINT).LT..5E-13) GO TO 6                 ELLI 290
TANPSI=TAN(PHI)                                             ELLI 300
NOUAD=NOUAD+1                                               ELLI 310
3  IF (ABS(ALPHA-BETA).LE.EPS) GO TO 7                       ELLI 320
PWR2=2.*PWR2                                                ELLI 330
PR2=PWR2                                                     ELLI 340
DENOM=ALPHA-BETA*TANPSI**2                                  ELLI 350
TOP=(ALPHA+BETA)*TANPSI                                     ELLI 360
CN=.5*(ALPHA-BETA)                                          ELLI 370
S1=S1+PWR2*CN*CN                                           ELLI 380
TEMP=SQRT(ALPHA*BETA)                                       ELLI 390
ALPHA=(ALPHA+BETA)*.5                                       ELLI 400
BETA=TEMP                                                    ELLI 410
IF (DENOM.EQ.0) GO TO 4                                     ELLI 420
TANPSI=TOP/DENOM                                            ELLI 430
IF (ABS(TANPSI).GE.EPS2) GO TO 4                           ELLI 440
NOUAD=2*NOUAD                                               ELLI 450
IF (TANPSI.GT.0) NOUAD=NOUAD-1                              ELLI 460
IF (ABS(TANPSI).LE.EPS) GO TO 5                             ELLI 470
MOUAD=MOD(NOUAD,4)                                          ELLI 480
SINP=TANPSI/SQRT(1.+TANPSI*TANPSI)                          ELLI 490
IF ((MOUAD.EQ.3).OR.(MOUAD.EQ.2)) SINP=-SINP              ELLI 500
S2=S2+CN*SINP                                              ELLI 510
GO TO 3                                                      ELLI 520
4  FINT=2*NOUAD-1                                           ELLI 530
PSI=FINT*PI                                                 ELLI 540
SINP=1.                                                      ELLI 550
IF (AMOD(FINT,4.).GT.2.) SINP=-1.                          ELLI 560
S2=S2+CN*SINP                                              ELLI 570
GO TO 6                                                      ELLI 580
5  FINT=NOUAD/2                                             ELLI 590
PSI=FINT*PI                                                 ELLI 600
6  IF (ABS(ALPHA-BETA).LT.EPS) GO TO 8                       ELLI 610
PR2=2.*PR2                                                  ELLI 620
CN=.5*(ALPHA-BETA)                                          ELLI 630
S1=S1+PR2*CN*CN                                           ELLI 640
TEMP=SQRT(ALPHA*BETA)                                       ELLI 650
ALPHA=(ALPHA+BETA)*.5                                       ELLI 660
BETA=TEMP                                                    ELLI 670
GO TO 6                                                      ELLI 680
7  FINT=NOUAD/2                                             ELLI 690
PSI=ATAN(TANPSI)+FINT*PI                                    ELLI 700

```

CONFORMAL MAPPING 4  
HAYES, KAHANER, KELLNER

8	F=PSI/(ALPHA*PWR2)	ELLI 710
	S1=L 0.5*S1	ELLI 720
	E=S2+S1*F	ELLI 730
	F=SIGN(F,PHI)	ELLI 740
	E=SIGN(E,PHI)	ELLI 750
	RETURN	ELLI 760
9	FINT=AINT(TDP*PSI)	ELLI 770
	E=FINT+ABS(SIN(PSI)-SIN(FINT*P12))	ELLI 780
	E=SIGN(E,PHI)	ELLI 790
	IF (PSI.GE.P12) GO TO 11	ELLI 800
	DENOM=1.-SIN(PHI)	ELLI 810
	IF ((DENOM.EQ.0.)OR.(DENOM.EQ.2.)) GO TO 11	ELLI 820
	F=5*ALOG(12.-DENOM)/DENOM)	ELLI 830
	RETURN	ELLI 840
	ENTRY CELLI	ELLI 850
	PHI=P12	ELLI 860
	IND=6	ELLI 870
	GO TO 1	ELLI 880
10	STOP	ELLI 890
	RETURN	ELLI 900
11	F=SIGN(1.E+294,PHI)	ELLI 910
	STOP	ELLI 920
	RETURN	ELLI 930
	END	ELLI 940
	 SUBROUTINE CONFORM (XP,YP,IP)	CONF 10
C	..... CONFORMAL MAPPING PACKAGE.....	CONF 20
C	..... SUBROUTINES REQUIRED .....	CONF 30
C	CONFORM	CONF 40
C	ATAN3	CONF 50
C	FN	CONF 60
C	BDRZ	CONF 70
C	QV	CONF 80
C	ECSW	CONF 90
C	ECRD. IFF ALL CALCULATIONS ARE DONE IN CORE	CONF 100
C	BDRY. IFF GENERALIZED BOUNDARY DATA ARE USED	CONF 110
C	PIVOTL. IFF THE DO LOOP IN CONFORM ENDING AT	CONF 120
C	STATEMENT 28 IS LEFT OUT	CONF 130
C	.....	CONF 140
C	.....	CONF 150
C	..... VARIABLES USED .....	CONF 160
C	.....	CONF 170
C	X,Y VECTORS OF BOUNDARY POINTS	CONF 180
C	(XN(I),YN(I)) UNIT NORMAL VECTOR AT (X(I),Y(I))	CONF 190
C	G(I) CURVATURE AT (X(I),Y(I))	CONF 200
C	XI,YI VECTORS OF POINTS ON BOUNDARY INTERMEDIATE TO X,Y	CONF 210
C	(XIN(I),YIN(I)) UNIT NORMAL AT I TH INTERMEDIATE BOUNDARY POINT	CONF 220
C	KV(I) IS THE NUMBER OF BOUNDARY POINTS ON THE FIRST THROUGH I-1ST	CONF 230
C	BOUNDARY SECTION	CONF 240
C	NDC TOTAL NUMBER OF BOUNDARY SECTIONS	CONF 250
C	DC(M) LENGTH OF M TH BOUNDARY SECTION SET IN SUBROUTINE BDRZ	CONF 260
C	HD(M) ARCLNGTH SPACING BETWEEN BOUNDARY POINTS ON M TH SECTION	CONF 270
C	N=KV/NDC+1 TOTAL NUMBER OF BOUNDARY POINTS	CONF 280
C	F 1. AT EXIT FROM LAPLACE F STORES SIGMA AT BOUNDARY POINTS	CONF 290
C	(X(I),Y(I)) I=1,N	CONF 300
C	2 DURING COMPUTATIONS F STORES RIGHT HAND SIDE OF MATRIX	CONF 310
C	EQUATION	CONF 320
C	IP EQUALS 0 FOR INTERIOR PROBLEM, 1 FOR EXTERIOR PROBLEM	CONF 330
C	.....	CONF 340
C	DIMENSION A(630),AA(630)	CONF 350
C	COMMON /MODE/ IN	CONF 360
C	COMMON /CONF/ F(630),XO,YO,CD(630),CN(630),KV(210),HD(210),G(630),	CONF 370
1	X(630),XN(630),Y(630),YN(630),XI(630),XIN(630),YI(630),YIN(630),ND	CONF 380
2	C,N	CONF 390
1	COMMON /ZLAP/ V,Q(12),H,HS,HSO2,HSO3,HI,HIS,ZZ,BZ,DP,AR,ME1,HSM,FB	CONF 400
	,BO,ZE,RI,W	CONF 410
	COMMON /LTT/ DC(210)	CONF 420
	LOGICAL BZ,IT,T1,T2	CONF 430
	EQUIVALENCE (A,CN), (AA,CD)	CONF 440
C	THE LOGICAL VARIABLE BZ=F MEANS THE PROBLEM HAS BEEN	CONF 450
C	RUN TO COMPLETION	CONF 460



CONFORMAL MAPPING 5  
HAYES, KAHANER, KELLNER

```

B2-T.
IF (IP.NE.1.AND.IP.NE.0) STOP
XO=XP
YO=YP
IN=IP
KV(1)=0
M=1
IF (IP.NE.1) GO TO 1
XO=0.
YO=0.
1 CALL BDRX (M,S,Z1,Z2,Z3,Z4,Z5)
C READ BOUNDARY DATA CARD CORRESPONDING TO SECTION M.
C Z1=0. IMPLIES A BLANK CARD
IF (Z1.EQ.0.) GO TO 3
NDC=M
KV(M+1)=KV(M)+IFIX(Z2)
HD(M)=DC(M)/FLOAT(KV(M+1)-KV(M)-1)
M1=KV(M)+1
N=KV(M)+1
C THE FOLLOWING DO LOOP GENERATES BOUNDARY AND INTERMEDIATE POINTS
C FOR SECTION M
DO 2 L=M1,N
Z=AMIN(DC(M),FLOAT(L-M1)*HD(M))
IF (L.NE.M1) CALL BDRZ (M,G.5*HD(M),X(L),Y(L),YN(L),XIN(L),
A)
CALL BDRZ (M,Z,X(L),Y(L),YN(L),XN(L),G(L))
2 CONTINUE
M=M+1
GO TO 1
C LAST DATA CARD FOR THIS COMPUTATION HAS BEEN READ
3 CONTINUE
C ALL DATA READ. START GENERATING EQUATIONS
DO 4 KO=1,N
C COMPUTE AND STORE EACH ROW OF THE MATRIX A AND VECTOR E.
CALL FN (X(KO),Y(KO),B,C)
C EACH CALL TO FN GENERATES THE LEFT HAND SIDE OF THE KO TH EQUATION
C AND STORES IT IN CN
F(KO)=.5*ALOG((X(KO)-XP)**2+(Y(KO)-YP)**2)
C ECSW TRANSFERS ROW OF MATRIX TO SUPPLEMENTAL STORAGE.
C DISK, CORE, TAPE, ECS, AT USERS OPTION
CN(N+1)=1.
4 CALL ECSW (CN,KO,1,N+1N)
C INSERT THE CORNER CONDITIONS
DO 10 M=1,NDC
M1=KV(M+1)
DO 5 L=1,NDC
M2=KV(L)+1
IF (ABS(X(M2)-X(M1))+ABS(Y(M2)-Y(M1)).LT.HD(M)*1.E-4)
5 CONTINUE
PRINT 19, M
RETURN
C AT THIS POINT IT MUST BE TRUE THAT THE END OF M-TH BOUNDARY
C SECTION JOINS THE START OF THE L-TH BOUNDARY SECTION
C AN*PI IS THE ANGLE AT WHICH THE CURVE M INTERSECTS THE CURVE L.
6 AN=ATAN2(YN(M1)*XN(M2)-XN(M1)*YN(M2),XN(M1)*XN(M2)+YN(M1)*YN(M2))
1 IF (ABS(AN).LT.1.E-4.OR.ABS(1.5707963-ABS(AN)).LT.1.E-4.OR.IN.E
1 Q.1
1 GO TO 7
CALL FN (X(M2+1),Y(M2+1),B,C)
A(N+1)=1.
CALL ECSW (A,M2,1,N+1N)
F(M2)=.5*ALOG((X(M2+1)-XP)**2+(Y(M2+1)-YP)**2)
CALL FN (X(M1),Y(M1),B,C)
A(N+1)=1.
CALL ECSW (A,M1,1,N+1N)
F(M1)=.5*ALOG((X(M1)-XP)**2+(Y(M1)-YP)**2)
GO TO 10
7 DO 8 L=1,N
8 A(L)=0.
CALL ECSW (A,M1,1,N+1N)
F(M1)=0.
IF (ABS(AN).LT.1.E-4.OR.IN.EQ.1) GO TO 9
CONF 470
CONF 480
CONF 490
CONF 500
CONF 510
CONF 520
CONF 530
CONF 540
CONF 550
CONF 560
CONF 570
CONF 580
CONF 590
CONF 600
CONF 610
CONF 620
CONF 630
CONF 640
CONF 650
CONF 660
CONF 670
CONF 680
CONF 690
CONF 700
CONF 710
CONF 720
CONF 730
CONF 740
CONF 750
CONF 760
CONF 770
CONF 780
CONF 790
CONF 800
CONF 810
CONF 820
CONF 830
CONF 840
CONF 850
CONF 860
CONF 870
CONF 880
CONF 890
CONF 900
CONF 910
CONF 920
CONF 930
CONF 940
CONF 950
CONF 960
CONF 970
CONF 980
CONF 990
CONF 1000
CONF 1010
CONF 1020
CONF 1030
CONF 1040
CONF 1050
CONF 1060
CONF 1070
CONF 1080
CONF 1090
CONF 1100
CONF 1110
CONF 1120
CONF 1130
CONF 1140
CONF 1150
CONF 1160
CONF 1170
CONF 1180
CONF 1190
CONF 1200
CONF 1210

```

	CALL ECSW (A,M2,1,N+IN)	CONF1220
	F(M2)=0.	CONF1230
	CALL ECSW (1.,M1,M1,1)	CONF1240
	CALL ECSW (1.,M2,M2,1)	CONF1250
	GO TO 10	CONF1260
9	CONTINUE	CONF1270
C	ANGLE BETWEEN TWO BOUNDARY SECTIONS IS ZERO, HENCE SIGMA IS	CONF1280
C	THE SAME AT BOTH ENDPOINTS	CONF1290
C	1 IN A(M1,M2)	CONF1300
C	1 IN A(M1,M1)	CONF1310
	CALL ECSW (1.,M1,M2,1)	CONF1320
	CALL ECSW (1.,M1,M1,1)	CONF1330
10	CONTINUE	CONF1340
	DO 12 M=1,NDC	CONF1350
	M1=KV(M)+2	CONF1360
	M2=KV(M)+1	CONF1370
	DO 11 J=M1,M2,2	CONF1380
	CN(J)=4.*HOM	CONF1390
11	CN(J+1)=2.*HD(M)	CONF1400
	CN(M1-1)=HDD(M)	CONF1410
12	CN(M2)=HDM	CONF1420
	CN(N+1)=0.	CONF1430
	IF (IN.EQ.1) CALL ECSW (CN,N+1,1,N+IN)	CONF1440
	F(N+1)=0.	CONF1450
	L=N+IN	CONF1460
C	SOLVE THE MATRIX EQUATION AF=D USING GAUSSIAN ELIMINATION.	CONF1470
	NM=L-1	CONF1480
	DO 17 J=1,NM	CONF1490
	M2=J+1	CONF1500
C	THE FOLLOWING 14 STATEMENTS ARE FOR ROW PIVOTING ON THE LARGEST	CONF1510
C	ELEMENT OF THE J TH COLUMN.	CONF1520
	CALL ECSR (T,J,J,1)	CONF1530
	T=ABS(T)	CONF1540
	M1=J	CONF1550
	DO 13 I=M2,L	CONF1560
	CALL ECSR (Z,I,J,1)	CONF1570
	Z=ABS(Z)	CONF1580
	IF (Z.LE.T) GO TO 13	CONF1590
	T=Z	CONF1600
	M1=I	CONF1610
13	CONTINUE	CONF1620
	CALL ECSR (AA(J),M1,J,L+1-J)	CONF1630
	IF (M1.EQ.J) GO TO 14	CONF1640
	CALL ECSR (A(J),J,L+1-J)	CONF1650
	CALL ECSW (A(J),M1,J,L+1-J)	CONF1660
	T=F(J)	CONF1670
	F(J)=F(M1)	CONF1680
	F(M1)=T	CONF1690
14	F(J)=F(J)/AA(J)	CONF1700
	DO 15 I=J,L	CONF1710
	K=L+J-I	CONF1720
15	AA(K)=AA(K)/AA(J)	CONF1730
	CALL ECSW (AA(J),J,L+1-J)	CONF1740
	DO 16 I=M2,L	CONF1750
	CALL ECSR (A(J),I,J,L+1-J)	CONF1760
	CALL PIVOTL (A,AA,M2,L)	CONF1770
C	*****MACHINE LANGUAGE REPLACEMENT*****	CONF1780
C	THE PREVIOUS CALL IS TO A MACHINE LANGUAGE CODE TO REPLACE	CONF1790
C	THE L**J DO LOOP BELOW. IF THIS CALL IS REMOVED, AND THE	CONF1800
C	COMMENTS REMOVED FROM THE DO, THE CODE WILL BE ALL FORTRAN	CONF1810
C	DO2RK M2,1	CONF1820
C	28 A(K)=A(K)/AA(K)*A(J)	CONF1830
C	*****MACHINE LANGUAGE REPLACEMENT*****	CONF1840
C	F(I)=F(I)/F(J)*A(J)	CONF1850
16	CALL ECSW (A(M2),1,M2,L+1-M2)	CONF1860
17	CONTINUE	CONF1870
C	THE FOLLOWING TEN STATEMENTS ARE FOR BACK SUBSTITUTION	CONF1880
	CALL ECSR (ZS,L,L,1)	CONF1890
		CONF1900

CONFORMAL MAPPING 7  
 HAYES, KANANER, KELLNER

```

FIL1=FI1/ZS
CALL ECSW (1.,L.L,1)
DO 18 I=2,L
    N1=L+1-I
    CALL ECSR (A(N1),N1,N1,L+1-N1)
    I1=I-1
    DO 18 J=1,I1
        L1=L+1-J
        F(N1)-F(I1): F(L1)*A(L1)
    
```

```

BZ=F.
RETURN

```

```

FORMAT (50H THE BOUNDARY IS NOT CLOSED AT THE END OF SECTION ,I2)
ENO

```

```

REAL FUNCTION ATAN3(Y,X)
ATAN3=ATAN2(Y,X)
THIS ROUTINE COMPUTES THE ARCTAN OF Y/X. THE COMPLICATED LOGIC
INSURES THAT THIS FUNCTION IS ALWAYS INCREASING AS THE CURVE
IS TRAVERSED IN POSITIVE SENSE IF THE DOMAIN IS CONVEX.
IF THE SITUATION 0/0 OCCURS WE FORCE THE ANGLE TO INCREASE
IF (ABS(ATAN3-ANGPRE).LT.3.1416) GO TO 3
ATAN3=ATAN3+2.*3.1415926535898
GO TO 1
ENTRY ATAN4
ATAN3=0.
ANGPRE=ATAN2(Y,X)
IF (ANGPRE.LE.1.E-12) ANGPRES=ANGPRE+2.*3.1415926535898
RETURN
ENTRY ATAN5
ATAN3=ATAN2(Y,X)
IF (ATAN3.GT.ANGPRE+.05) GO TO 3
ATAN3=ATAN3+2.*3.1415926535898
GO TO 2
ANGPRE=ATAN3
RETURN
END

```

```

SUBROUTINE FN (S,T,R,TH)
FN SERVES DUAL FUNCTION
1. GENERATE ONE ROW OF MATRIX, AND RETURN
2. COMPUTES (R,TH) CORRESPONDING TO INPUT (S,T)
LOGICAL BZ,R1,BQ
COMMON /ZLAP/ V,Q(12),H,HS,HSO2,HCO3,HI,HIS,ZZ,BZ,DP,AR,ME1,HBM,F
,BQ,ZE,R1,W
COMMON /CONF/ F(630),XO,YO,CD(630),CN(630),KV(210),HD(210),G(630),
X(630),XN(630),Y(630),YN(630),XI(630),XIN(630),YI(630),YIN(630),ND
C,N
COMPLEX GP
IF (ABS(X(1)-S)+ABS(Y(1)-T).GE.1.E-3*HD(1)) GM=ATAN4(Y(1)-T,X(1)-S)
IF (ABS(X(1)-S)+ABS(Y(1)-T).LT.1.E-3*HD(1)) GM=ATAN4(XN(1)-YN(1))
GM=HM=0.
PUT THE VARIABLES IN COMMON.
V=S
W=T
DO 1 ME0=1,NDC
INITIALIZE VARIABLES FOR THE ROUTINE QV.
MO=KV/(ME0+1)
ME1=KV/(ME0+1)-2
H=HD/(ME0)
ZZ=ALOG(H)
HSO2=H**2/2.
HCO3=H**3/3.
TPIH2=.25/(3.1415926535898*HSO2)
TPIH=.5*TPIH2
HS=HSO2**2.
HS2=2.*HS
H2=2.*H
H3=3.*H

```

CONF 1910  
 CONF 1920  
 CONF 1930  
 CONF 1940  
 CONF 1950  
 CONF 1960  
 CONF 1970  
 CONF 1980  
 CONF 1990  
 CONF 2000  
 CONF 2010  
 CONF 2020  
 CONF 2030  
 CONF 2040

ATN3 10  
 ATN3 20  
 ATN3 30  
 ATN3 40  
 ATN3 50  
 ATN3 60  
 ATN3 70  
 ATN3 80  
 ATN3 90  
 ATN3 100  
 ATN3 110  
 ATN3 120  
 ATN3 130  
 ATN3 140  
 ATN3 150  
 ATN3 160  
 ATN3 170  
 ATN3 180  
 ATN3 190  
 ATN3 200  
 ATN3 210  
 ATN3 220

FN 10  
 FN 20  
 FN 30  
 FN 40  
 FN 50  
 FN 60  
 FN 70  
 FN 80  
 FN 90  
 FN 100  
 FN 110  
 FN 120  
 FN 130  
 FN 140  
 FN 150  
 FN 160  
 FN 170  
 FN 180  
 FN 190  
 FN 200  
 FN 210  
 FN 220  
 FN 230  
 FN 240  
 FN 250  
 FN 260  
 FN 270  
 FN 280  
 FN 290  
 FN 300  
 FN 310  
 FN 320  
 FN 330

	HI=1/H	FN 340
	HSM=08*HS	FN 350
	HIS=1/H*SO2	FN 360
	ZE=H/(46.*3.1415926535898)	FN 370
	CN(M0)-CD(M0)=0.	FN 380
C	THE ENTRY POINT QVF INITIALIZES VARIABLES USED BY QV.	FN 390
	CALL QVF (M0+1)	FN 400
C	STORES MATRIX ROW IN CN	FN 410
C	BQ FALSE -- COMPUTE INTEGRAL THE LONG WAY -- POLYNOMIAL REPLACEMENT	FN 420
C	BQ TRUE -- COMPUTE INTEGRAL THE SHORT WAY -- SIMPSONS RULE	FN 430
	BQ=F.	FN 440
	DO 1 M1=M0,ME1,2	FN 450
	CALL QVS (M1+1)	FN 460
	IF (BQ) GO TO 1	FN 470
C	Q(K+1) . . . K=0,1,2 CONTAIN INTEGRALS OF T**K*THETA FOR 2. ABOVE	FN 480
C	Q(K+4) . . . K=0,1,2 CONTAIN INTEGRALS OF T**K*LOG(R) FOR 2. ABOVE.	FN 490
C	OR R SMALL RELATIVE TO H	FN 500
C	CN(K) INTEGRAL OF PIECEWISE POLYNOMIAL OF DEGREE 2. TIMES LOG	FN 510
C	CD(K) INTEGRAL OF PIECEWISE POLYNOMIAL OF DEGREE 2. TIMES ARCTAN	FN 520
	TT=Q(3)-H3*Q(2)+HS2*Q(1)	FN 530
	TS=Q(6)-H3*Q(5)+HS2*Q(4)	FN 540
	CD(M1+1)=H2*Q(2)-Q(3)	FN 550
	CN(M1+1)=H2*Q(5)-Q(6)	FN 560
	CD(M1+2)=Q(3)-H*Q(2)	FN 570
	CN(M1+2)=Q(6)-H*Q(5)	FN 580
	CALL OV (M1+2)	FN 590
	CD(M1)=CD(M1)+(TT+Q(3)-H*Q(2))*TPIH2	FN 600
	CN(M1)=CN(M1)+(TS+Q(6)-H*Q(5))*TPIH	FN 610
	CD(M1+1)=(CD(M1+1)+Q(1)*HS-Q(3))*TPIH2*2.	FN 620
	CN(M1+1)=(CN(M1+1)+Q(4)*HS-Q(6))*TPIH*2.	FN 630
	CD(M1+2)=(CD(M1+2)+Q(3)+H*Q(2))*TPIH2	FN 640
	CN(M1+2)=(CN(M1+2)+Q(6)+H*Q(5))*TPIH	FN 650
1	CONTINUE	FN 660
	IF (BZ) RETURN	FN 670
C	COMPUTE APPROXIMATION OF G AND H BY COMPUTING DOT PRODUCT OF	FN 680
C	SIGMA WITH CN AND CD	FN 690
	DO 2 M=-1,N	FN 700
	GM=GM-CN(M)*F(M)	FN 710
	HM=HM-CD(M)*F(M)	FN 720
2	CONTINUE	FN 730
	GP=CMPLX(S.XO,T-YO)*CEXP(CMPLX(GM-F(N+1),HM))	FN 740
	R=CABS(GP)	FN 750
	TH=ATAN2(AIMAG(GP),REAL(GP))	FN 760
	RETURN	FN 770
	END	FN 780
	SUBROUTINE BDRZ (M,S,XX,YY,XP,YP,G)	BDRZ 10
C	GIVEN INPUT M-BOUNDARY SECTION, S=ARCLENGTH ALONG THAT SECTION	BDRZ 20
C	OUTPUT XX,YY,XP,YP,G.=COORDINATES OF POINT, DXDS, DYDS, AND	BDRZ 30
C	CURVATURE	BDRZ 40
C	BDRZ ENTRY SETS UP DATA POINTS ON BOUNDARY CURVE AFTER BDRX PICKS	BDRZ 50
C	TYPE OF SECTION	BDRZ 60
	COMMON /LTT/ DC(210)	BDRZ 70
C	THE FUNCTION ABE IS USED TO DETERMINE IF A CARD FIELD IS BLANK	BDRZ 80
C	ABE=-1 IF AND ONLY IF X=0.	BDRZ 90
	ABE(X)=ABS(X)+SIGN(1.,X)	BDRZ 100
	DIMENSION A(7)	BDRZ 110
	COMPLEX Z2,Z3	BDRZ 120
	EQUIVALENCE (A(1),GM), (A(3),AL), (A(4),BE), (A(5),R), (A(6),D), (	BDRZ 130
	A(7),GA1)	BDRZ 140
	CALL ECSR7W (A,M,0,7)	BDRZ 150
	IF (GM.NE.0.) GO TO 1	BDRZ 160
	IF (SIGN(1.,GM).NE.-1.) GO TO 2	BDRZ 170
C	CALL BDRY IF THE BOUNDARY DATA IS GENERALIZED	BDRZ 180
	CALL BDRY (M,S,XX,YY,XP,YP,G)	BDRZ 190
	RETURN	BDRZ 200
C	GENERATE BOUNDARY INFORMATION FOR CIRCULAR ARC	BDRZ 210
1	XX=R*COS(D*S/R+GA1)+AL	BDRZ 220
	YY=R*SIN(D*S/R+GA1)+BE	BDRZ 230
	XP=(YY-BE)*D/R	BDRZ 240

CONFORMAL MAPPING 9  
HAYES, KAHANER, KELLNER

	YP=(XX-AL)*O/R	BDRZ 250
	GO TO 3	BDRZ 260
C	GENERATE BOUNDARY INFORMATION FOR A LINE SEGMENT	BDRZ 270
2	XX=AL*S+D	BDRZ 280
	YY=BE*S+R	BDRZ 290
	XP=AL	BDRZ 300
	YP=BE	BDRZ 310
3	G=GM	BDRZ 320
	RETURN	BDRZ 330
	ENTRY BDRX	BDRZ 340
C	ENTRY BDRX READS DATA CARDS AND DECIDES IF BOUNDARY SECTION IS	BDRZ 350
C	LINE, CIRCULAR ARC, OR GIVEN EXPLICITLY BY USER SUBROUTINE	BDRZ 360
	XX=0.	BDRZ 370
C	READ BOUNDARY DATA CARD	BDRZ 380
4	READ 11, X1,Y1,X2,Y2,X3,Y3,KV	BDRZ 390
	IF (ABE(X1)+ABE(Y1)+ABE(X2)+ABE(Y2)+ABE(X3)+ABE(Y3).NE. 6) GO TO	BDRZ 400
	5	BDRZ 410
	IF (M.NE.1) RETURN	BDRZ 420
C	IF ANY CARD IS BLANK EXCEPT THE FIRST ONE, RETURN WITH XX=0.	BDRZ 430
	GO TO 4	BDRZ 440
5	XX=1.	BDRZ 450
	KV=MAX(KV,3)+MOD(MAX(KV,3),2)-1	BDRZ 460
C	REQUIRES NUMBER OF POINTS ON A SECTION BE 3 OR MORE AND ODD	BDRZ 470
	YY=FLOAT(KV)*.1	BDRZ 480
	IF (MOD(M,55).EQ.1) PRINT 12	BDRZ 490
C	55 LINES PER PAGE, NEW HEADING ON EACH PAGE	BDRZ 500
	IF (ABE(X1)+ABE(Y1)+ABE(X2)+ABE(Y2)+ABE(X3).GT. 5) GO TO 6	BDRZ 510
C	THIS SECTION HAS GENERALIZED BOUNDARY DATA	BDRZ 520
	PRINT 13, M,Y3,KV	BDRZ 530
	CALL ECSR7W (.0,M,1,1)	BDRZ 540
C	ECSR7W IS ENTRY POINT TO ECSSW, WRITES 7 WORDS INTO ECS.	BDRZ 550
C	CORRESPONDING TO ONE BOUNDARY SECTION. FOR GENERALIZED BOUNDARY	BDRZ 560
C	THIS IS WRITTEN AS .0.	BDRZ 570
	DC(M)=Y3	BDRZ 580
	IF (DC(M).GT.0) RETURN	BDRZ 590
	PRINT 14	BDRZ 600
	STOP	BDRZ 610
6	IF (ABE(X1)+ABE(Y1).NE. 2.OR.M.EQ.1) GO TO 7	BDRZ 620
C	CHECK TO SEE IF BOUNDARY SECTION IS LINE OR CIRCULAR ARC	BDRZ 630
C	IF FIRST TWO FIELDS BLANK, BOUNDARY SECTION STARTS AT THE	BDRZ 640
C	END OF LAST SECTION	BDRZ 650
	X1=X0	BDRZ 660
	Y1=Y0	BDRZ 670
7	IF (ABE(X3)+ABE(Y3).EQ. 2) GO TO 8	BDRZ 680
	PRINT 15, M,X1,Y1,X2,Y2,X3,Y3,KV	BDRZ 690
	GO TO 10	BDRZ 700
C	PROCESS BOUNDARY DATA	BDRZ 710
C		BDRZ 720
8	PRINT 16, M,X1,Y1,X2,Y2,KV	BDRZ 730
C	THIS SECTION IS FOR PROCESSING BOUNDARY CARDS HAVING LINE SEGMENTS	BDRZ 740
	GM=0.	BDRZ 750
	F=SQRT((X1-X2)**2+(Y1-Y2)**2)	BDRZ 760
	D=X1	BDRZ 770
	R=Y1	BDRZ 780
	AL=(X2-X1)/F	BDRZ 790
	BE=(Y2-Y1)/F	BDRZ 800
	X0=X2	BDRZ 810
	Y0=Y2	BDRZ 820
9	CALL ECSR7W (A,M,1,7)	BDRZ 830
	DC(M)=F	BDRZ 840
	RETURN	BDRZ 850
C	THIS SECTION IS FOR PROCESSING BOUNDARY CARDS HAVING CIRCULAR ARCS	BDRZ 860
10	R2=(X1**2+Y1**2)-(X2**2+Y2**2)	BDRZ 870
	R3=(X1**2+Y1**2)-(X3**2+Y3**2)	BDRZ 880
	DET=4.*((X1-X2)*(Y1-Y3)-(Y1-Y2)*(X1-X3))	BDRZ 890
	AL=2.*((R2)*(Y1-Y3)-(R3)*(Y1-Y2))/DET	BDRZ 900
	BE=2.*((X1-X2)*R3-(X1-X3)*R2)/DET	BDRZ 910
	R=SQRT((X1-AL)**2+(Y1-BE)**2)	BDRZ 920
	GA1=ATAN2(Y1-BE,X1-AL)	BDRZ 930

CONFORMAL MAPPING 10  
HAYES, KAHANER, KELLNER

	Z2=CMPLX(X2,AL,Y2,BE)/CMPLX(X1,AL,Y1,BE)	BDRZ 940
	Z3=CMPLX(X3,AL,Y3,BE)/CMPLX(X2,AL,Y2,BE)	BDRZ 950
	D=SIGN(1,AIMAG(Z2))	BDRZ 950
	T1=ATAN2(AIMAG(Z2),REAL(Z2))	BDRZ 970
	T2=ATAN2(AIMAG(Z3),REAL(Z3))	BDRZ 980
	IF (ABS(T1).GT.ABS(T2)) D=SIGN(1,AIMAG(Z3))	BDRZ 980
	F=(ATAN2(Y3-BE,X3-AL)-GA1)*R*O	BDRZ1000
	IF (F.LT.O.) F=F+6.28318530717966*R	BDRZ1010
	XD=X3	BDRZ1020
	YD=Y3	BDRZ1030
	GM=D*.5/R	BDRZ1040
	GO TO 9	BDRZ1050
C		BDRZ1060
11	FORMAT (6E10.0,13)	BDRZ1070
12	FORMAT (*1SECTION*.72X*.Y3, OR SPECIAL POINTS IN	BDRZ1080
1	*' NUMBER*.6X*.X1*.12X*.Y1*.12X*.X2*.12X*.Y2*.12X*.X3*.9X*.BOUND	BDRZ1080
2	ARY LENGTH SECTION *)	BDRZ1100
13	FORMAT (1H 13.74X,1PE14.6,12X,13)	BDRZ1110
14	FORMAT (*O THE NEXT BOUNDARY SECTION HAS NON-POSITIVE LENGTH.*	BDRZ1120
1	)	BDRZ1130
15	FORMAT (1H 13.4X,1PE14.6,2X,10X,13)	BDRZ1140
16	FORMAT (1H 13.4X,1PE14.6,40X,13)	BDRZ1150
	END	BDRZ1160
	SUBROUTINE QV (I)	QV 10
	LOGICAL BQ	QV 20
	COMMON /ZLAP/ V,Q(12),H,HS,HSO2,HCOS,HI,HIS,ZZ,BZ,DP,AR,ME1,MSM,FB	QV 30
1	,BQ,ZE,RI,W	QV 40
	COMMON /CONF/ F(630),XO,YO,CD(630),CN(630),KV(210),MD(210),G(630),	QV 50
1	X(630),XN(630),Y(630),YN(630),XI(630),XIN(630),YI(630),YIN(630),ND	QV 60
2	,CN	QV 70
	DIMENSION MP(13)	QV 80
C	THE VARIABLES R3 AND AL3 ARE STORED FROM A PREVIOUS CALL TO QV.	QV 90
1	AL1=AL3	QV 100
	R1=R3	QV 110
	BQ=F.	QV 120
C	THE VARIABLES R1,R2 AND R3 ARE USED TO CONSTRUCT THE POLYNOMIAL	QV 130
C	Q(T), FOLLOWING EQ (9) OF THE WRITE-UP.	QV 140
	R2=(X(II)-V)**2+(Y(II)-W)**2	QV 150
	R3=(X(II)-V)**2+(Y(II)-W)**2	QV 160
	AL2=(X(II)-V)*XIN(II)-(Y(II)-W)*YIN(II)	QV 170
	AL3=(X(II)-V)*XN(II)-(Y(II)-W)*YN(II)	QV 180
C	THE FOLLOWING TWO STATEMENTS ARE USED TO DETERMINE IF AND WHERE	QV 190
C	Q(T)=0.	QV 200
	IF ((R1.GE.HSM).AND.(R2.GE.HSM).AND.(R3.GE.HSM)) GO TO 2	QV 210
C	THE FOLLOWING TWO STATEMENTS ARE USED TO DETERMINE WHERE Q(T)=0.	QV 220
	IF ((R3.LT.HS*1.E-6).OR.(R1.LT.HS*1.E-6)) GO TO 10	QV 230
	IF (R2.LT.HS*1.E-6) GO TO 15	QV 240
C	THIS SECTION OF THE PROGRAM EVALUATES THE INTEGRALS IN EQ. (9)	QV 250
2	A=((R1+R3)-2.*R2)*HIS	QV 260
	B=(4.*R2-(3.*R1+R3))*HI	QV 270
	AL=(AL1+AL3-2.*AL2)*HIS	QV 280
	BE=(4.*AL2-(3.*AL1+AL3))*HI	QV 290
	ELC=EL	QV 300
	EL=ALOG(R3)	QV 310
	IF (ABS(A).LT.R1*HIS*1.E-4) GO TO 7	QV 320
C	IF A**2 IS VERY SMALL COMPARED WITH R1 AND ONE USES THE EXPLICIT	QV 330
C	EVALUATION OF THE INTEGRALS, THEN ROUND-OFF ERROR WILL BE A PROBLEM	QV 340
	P=4.*A**R1-B*B	QV 350
	E=2.*R1+B**H	QV 360
	PS=SQRT(ABS(P))	QV 370
C	THE FOLLOWING TWO STATEMENTS DETERMINE WHETHER THE DISCRIMINANT OF	QV 380
C	Q(T) IS POSITIVE, NEGATIVE OR APPROXIMATELY ZERO.	QV 390
	IF (P.GE.HS*1.E-8) GO TO 4	QV 400
	IF (P.LE.-HS*1.E-8) GO TO 3	QV 410
	QO=ALOG((E+H*PS)/(E-H*PS))/PS	QV 420
	GO TO 5	QV 430
3	QO=2.*H/E	QV 440
	GO TO 5	QV 450
4	QO=2.*ATAN2(H*PS,E)/PS	QV 460
5	E=1/A	QV 470
	ZZZ=ATAN3(Y(II)-W,X(II)-V)	QV 480
	Q1=.5*E*(EL-ELC-B*QO)	QV 490
	Q2=E*(H-B*Q1-R1*QO)	QV 500
	Q3=E*(HSO2-B*Q2-R1*Q1)	QV 510
	Q4=E*(HCOS-B*Q3-R1*Q2)	QV 520
	Q5=E*(H**4/4.-B*Q4-R1*Q3)	QV 530
6	Q(1)=AL*Q3+BE*Q2+AL1*Q1-H*ZZZ	QV 540
	Q(2)=(AL*Q4+BE*Q3+AL1*Q2)/2.-HSO2*ZZZ	QV 550
	Q(3)=(AL*Q5+BE*Q4+AL1*Q3)/2.-HCOS*ZZZ	QV 560

CONFORMAL MAPPING 11  
HAYES, KAHANER, KELLNER

	Q(4)=EL*H. 2.*A*Q2-B*Q1	QV 570
	Q(5)=EL*HSO2.*A*Q. 0.5*B*Q2	QV 580
	Q(6)=EL*HCO3. (2.*A*Q4+B*Q3)*.333333333333333	QV 590
	RETURN	QV 600
C	IF B*H IS ALSO SMALL IN COMPARISON WITH R1 THEN ANOTHER METHOD IS	QV 610
C	NEEDED TO EVALUATE Q0,Q1....Q4.	QV 620
7	IF (ABS(B).LT.R1*HI*.1E-3) GO TO 8	QV 630
	ZZZ=ATAN3(Y(II). W,X(II).V)	QV 640
	BI=1/B	QV 650
	Q0=BI*ALOG(1.+B*H/R1)	QV 660
	Q1=BI*(H. R1*Q0)	QV 670
	Q2=BI*(HSO2. R1*Q1)	QV 680
	Q3=BI*(HCO3. R1*Q2)	QV 690
	Q4=BI*(HSO2**2. R1*Q3)	QV 700
	Q5=BI*(H**5/5. R1*Q4)	QV 710
	GO TO 6	QV 720
8	DO 9 J=4,12	QV 730
9	HP(J)=H**J/FLOAT(J)	QV 740
	ZZZ=ATAN3(Y(II). W,X(II). V)	QV 750
	<del>HP(1)=1.</del>	QV 760
	P=A/R1	QV 770
	E=B/R1	QV 780
	Q5=(HP(7). E*HP(8)+(E*E. P)*HP(9)+E*(2.*P. E*E)*HP(10)+P*P. 3.*E*E)*H	QV 790
1	P(11). 3.*E*P*P*HP(12). (P**3)*HP(13))/R1	QV 800
1	Q4=(HP(6). E*HP(7)+(E*E. P)*HP(8)+E*(2.*P. E*E)*HP(9)+P*P. 3.*E*E)*HP	QV 810
	(10). 3.*E*P*P*HP(11). (P**3)*HP(12))/R1	QV 820
	Q3=H**4/(4.*R1). E*Q4.P*Q5	QV 830
	Q2=HCO3/R1. E*Q3.P*Q4	QV 840
	Q1=HSO2/R1. E*Q2.P*Q3	QV 850
	Q0=H/R1. E*Q1.P*Q2	QV 860
	GO TO 5	QV 870
10	IF (R1.LT.HS*.1E-6) GO TO 11	QV 880
C	IN THIS CASE R3 IS ZERO.	QV 890
	AL=(AL1/R1+G(II). 2.*AL2/R2)*HIS	QV 900
	BE=(4.*AL2/R2. 3.*AL1/R1. G(II))*HI	QV 910
	AL1=AL1/R1	QV 920
	A=R1/HIS	QV 930
	B=(B.*R2. 2.*R1)/(H**3)	QV 940
	ZS=1.	QV 950
	ZZZ=ATAN3(XN(II). YN(II))	QV 960
	GO TO 12	QV 970
	IN THIS CASE R1 IS ZERO	QV 980
11	AL=(G(II). 1)*AL3/R3. 2.*AL2/R2)*HIS	QV 990
	BE=(4.*AL2/R2. 3.*G(II). 1). AL3/R3)*HI	QV 1000
	AL1=G(II). 1)	QV 1010
	A=(B.*R2. R3)/HIS	QV 1020
	B=2.*(R3. 4.*R2)/(H**3)	QV 1030
	EL=ALOG(R3)	QV 1040
	ZS=0.	QV 1050
	ZZZ=ATAN5(Y(II). W,X(II). V)	QV 1060
12	Q1=B*H/A	QV 1070
	P=ALOG(A)+2.*ZZ	QV 1080
	R1=1.	QV 1090
	Q2=Q1**2	QV 1100
	Q3=Q2*Q1	QV 1110
	IF (ABS(Q1).LT.5.E-4) GO TO 14	QV 1120
	E=ALOG(1.+Q1)	QV 1130
	Q(4)=H*(P. 2.+(1.+Q1)*E. Q1)/Q1	QV 1140
	Q(5)=HSO2*(P. (1.+2.*ZS)+(Q2. 1.)*E+Q. 0.5*Q2)/Q2	QV 1150
13	Q(6)=HCO3*(P. (2./3.+3.*ZS)+(Q3+.1)*E. Q1+.5*Q2. Q3/3.)/Q3	QV 1160
	Q(1)=(AL.*H**4/4.+BE*HCO3*AL1*HSO2)/R1. ZZZ*H	QV 1170
	Q(2)=(AL.*H**5/5.+BE*HSO2**2*AL1*HCO3)/(2.*R1). ZZZ*HSO2	QV 1180
	Q(3)=(AL.*H**6/6.+BE*H**5/5.+AL1*H**4/4.)/(3.*R1). ZZZ*HCO3	QV 1190
	RETURN	QV 1200
14	Q(4)=H*(P. 2.+Q1*(.5. Q1/8.+Q2/12. 0.06*Q3))	QV 1210
	Q(5)=HSO2*(P. (1.+2.*ZS)+Q1*(.8666666666666666. 0.26*Q1+Q2/7.5. Q3/12.))	QV 1220
	Q(6)=HCO3*(P. (2./3.+3.*ZS)+Q1*(.7. 0.3*Q1+Q2/6.+Q3/9.33333333333333))	QV 1230
	GO TO 13	QV 1240
C	IN THIS CASE R2 IS ZERO. THE VARIABLE G HAS BEEN COMPUTED JUST	QV 1250
C	PREVIOUS TO THE CALL TO FN. IT IS NOT THE VALUE OF G(1)	QV 1260

CONFORMAL MAPPING 12  
 HAYES, KAHANER, KELLNER

C			
15	CORRESPONDING TO THE POINT (X(I),Y(I)).		QV 1270
	AL=(AL1/R1+AL3/R3-2.*G)*HS		QV 1280
	BE=(4.*G-3.*AL1/R1-AL3/R3)*HI		QV 1290
	AL1=AL1/R1		QV 1300
	R1=1.		QV 1310
	ELC=EL		QV 1320
	EL=ALOG(R3)		QV 1330
	Q(4)=H*((ELC+EL)*6-2)		QV 1340
	Q(5)=HBO2*((ELC-EL-8.)*.25+EL)		QV 1350
	Q(6)=HCO3*((ELC-EL-13.333333333333)*.125+EL)		QV 1360
	ZZ2=ATAN3(Y(I)-W.X(I)-V)		QV 1370
	GO TO 13		QV 1380
	ENTRY QVF		QV 1390
C	THIS ENTRY POINT IS FOR INITIALIZING THE VARIABLES AL3 AND R3.		QV 1400
	AL3=(X(I)-1)-V**2*(Y(I)-1)-W**2*Y(I)-1)		QV 1410
	R3=(X(I)-1)-V**2*(Y(I)-1)-W**2		QV 1420
	IF (R3.GT.0.) EL=ALOG(R3)		QV 1430
	RETURN		QV 1440
	ENTRY QVS		QV 1450
	IF (R3.LT.50.*HS) GO TO 1		QV 1460
C	IF R30.GE.50.*HS, THEN ON THE TWO INTERVALS TO BE CONSIDERED ONE		QV 1470
C	CANNOT GUARANTEE THAT THE DISTANCE REMAINS GREATER THAN 5.*H.		QV 1480
C	THIS IS NOT THE BEST POSSIBLE TEST, BUT THE LOGIC NEEDED FOR		QV 1490
C	THE BEST POSSIBLE TEST IS RELATIVELY DIFFICULT.		QV 1500
C	THESE VARIABLES ARE USED IN THE CALCULATION BELOW.		QV 1510
	IF (.NOT.B2) GO TO 1		QV 1520
	R1=R3		QV 1530
	R12=(X(I)-1)-V**2*(Y(I)-1)-W**2		QV 1540
	R2=(X(I)-1)-V**2*(Y(I)-1)-W**2		QV 1550
	R13=(X(I)-1)-V**2*(Y(I)-1)-W**2		QV 1560
	R3=(X(I)-1)-V**2*(Y(I)-1)-W**2		QV 1570
	IF (B0) GO TO 16		QV 1580
C	IN THIS CASE THE CONTRIB. FROM THE PART OF THE BOUNDARY BETWEEN		QV 1590
C	(X(I)-1),Y(I-1)) AND (X(I+1),Y(I+1)) IS ADDED TO CD(I-1)		QV 1600
C	AND CN(I-1).		QV 1610
	CN(I-1)=CN(I-1)+ZE*ALOG(R1*((R1*R12**2)**3/R13**2)**2)/2.		QV 1620
	GO TO 17		QV 1630
C	THE TWO INTEGRALS STORED IN CD(I-1) AND CN(I-1) EVALUATED IN ONE		QV 1640
C	STEP. THIS INVOLVES AN INTEGRATION OVER THE PART OF THE BOUNDARY		QV 1650
C	BETWEEN (X(I-3),Y(I-3)) AND (X(I+1),Y(I+1)).		QV 1660
16	CN(I-1)=ZE*ALOG(R1*((R1*R12*R102)**3/(R13*R103)**2)		QV 1670
C	THE TWO INTEGRALS STORED IN CD(I) AND CN(I) ARE EVAL. IN ONE STEP.		QV 1680
C	THIS ONLY INVOLVES AN INTEGRATION OVER THE PART OF THE BOUNDARY		QV 1690
C	BETWEEN (X(I-1),Y(I-1)) AND (X(I+1),Y(I+1)).		QV 1700
17	CN(I)=ZE*6.*ALOG((R12/R13)**2*R2)		QV 1710
C	THESE FOUR VARIABLES ARE STORED FOR POSSIBLE USE IN THE NEXT CALL TO		QV 1720
C	QV.		QV 1730
	R103=R12		QV 1740
	R102=R13		QV 1750
	B0=T.		QV 1760
C	THE FOLLOWING STATEMENT ASKS IF THE POINT (X(I+1),Y(I+1)) IS AT END		QV 1770
C	OF THE BOUNDARY SECTION OR IF THE POINT IS TOO CLOSE TO (V,W) FOR		QV 1780
C	SIMPSONS RULE INTEGRATION TO BE USED		QV 1790
	IF (R3.GT.50.*HS.AND.1.NE.ME1) RETURN		QV 1800
C	IN THIS CASE THE CONTRIBUTNS TO THE INTEGRALS CD(I+1) AND CN(I+1)		QV 1810
C	FROM THE PART OF THE BOUNDARY BETWEEN (X(I-1),Y(I-1)) AND (X(I+1),		QV 1820
C	Y(I+1)) MUST BE ADDED IN NOW.		QV 1830
	EL=ALOG(R3)		QV 1840
	CN(I+1)=ZE*ALOG(R3*((R3*R13**2)**3/R12**2)**2)/2.		QV 1850
	RETURN		QV 1860
	END		QV 1870



CONFORMAL MAPPING 13  
HAYES, KAHANER, KELLNER

	SUBROUTINE ECSW (A,I,J,L)	ECSW 10
	COMMON /MODE/ IN	ECSW 20
	COMMON /CONF/ FN(30),X0,Y0,CD(630),CN(630),KV(210),MD(210),G(630),	ECSW 30
1	X(630),XN(630),Y(630),YN(630),XI(630),XIN(630),YI(630),YIN(630),NO	ECSW 40
2	C,N	ECSW 50
	COMMON /ZLARI/ V,Q(12),H,HS,HSO2,MCOS,HI,HS,ZZ,BZ,DP,AR,ME1,MSM,FB	ECSW 60
1	BO,ZE,RI,W	ECSW 70
C	THIS ENTRY POINT WRITES L WORDS INTO THE I-TH ROW OF THE MATRIX A	ECSW 80
C	STARTING IN COLUMN J.	ECSW 90
	KN=6	ECSW 100
	M=(I-1)*(N+IN)+J-1+1470	ECSW 110
1	CALL ECWR (A,M,L,JI)	ECSW 120
	IF (J.JE.Q.0) RETURN	ECSW 130
	KN=KN-1	ECSW 140
	IF (KN.GT.0) GO TO 1	ECSW 150
	PRINT 3, L,M	ECSW 160
	STOP	ECSW 170
	ENTRY ECSR	ECSW 180
C	THIS ENTRY POINT READS L WORDS FROM THE I-TH ROW OF THE MATRIX A	ECSW 190
C	STARTING IN COLUMN J.	ECSW 200
	M=(I-1)*(N+IN)+J-1+1470	ECSW 210
	KN=6	ECSW 220
2	CALL ECRD (A,M,L,JI)	ECSW 230
	IF (J.JE.Q.0) RETURN	ECSW 240
	KN=KN-1	ECSW 250
	IF (KN.GT.0) GO TO 2	ECSW 260
	PRINT 4, L,M	ECSW 270
	STOP	ECSW 280
	ENTRY ECSRW	ECSW 290
C	THIS ENTRY POINT IS USED TO READ AND WRITE	ECSW 300
C	SELDOM USED BOUNDARY DATA.	ECSW 310
	M=7*(I-1)	ECSW 320
	KN=6	ECSW 330
	IF (J.E.Q.1) GO TO 1	ECSW 340
	GO TO 2	ECSW 350
C		ECSW 360
3	FORMAT (43H ECS WRITE ERROR FLAG SET. TRYING TO WRITE ,13,28H WORD	ECSW 370
1	S STARTING AT LOCATION ,17,1H.)	ECSW 380
4	FORMAT (41H ECS READ ERROR FLAG SET. TRYING TO READ ,13,28H WORDS	ECSW 390
1	STARTING AT LOCATION ,17,1H.)	ECSW 400
	END	ECSW 410
	SUBROUTINE ECRD (AA,I,J,JI)	ECRD 10
C	THIS ROUTINE CAN BE REMOVED IF E.C.S. IS AVAILBLE	ECRD 20
C	LEAVING THIS SUBROUTINE IN CAUSES THE ENTIRE PROBLEM TO	ECRD 30
C	BE RUN IN CORE	ECRD 40
C	IF THIS SUBROUTINE IS RETAINED THE VECTOR A BELOW MUST BE	ECRD 50
C	DIMENSIONED N*N+1500	ECRD 60
C	IN TEST DECK 23404 - 148*148 + 1500	ECRD 70
C	THIS CORRESPONDS TO THE LARGEST N FOR OUR TEST PROBLEMS	ECRD 80
	DIMENSION A(23404)	ECRD 90
	DIMENSION AA(2)	ECRD 100
	JJ=0	ECRD 110
	I1=I+1	ECRD 120
	I2=I+J	ECRD 130
	M=1	ECRD 140
	DO 1 K=I1,I2	ECRD 150
	AA(M)=A(K)	ECRD 160
	M=M+1	ECRD 170
	RETURN	ECRD 180
	ENTRY ECWR	ECRD 190
	JJ=0	ECRD 200
	M=1	ECRD 210
	I1=I+1	ECRD 220
	I2=I+J	ECRD 230
	DO 2 K=I1,I2	ECRD 240
	A(K)=AA(M)	ECRD 250
	M=M+1	ECRD 260
	RETURN	ECRD 270
	END	ECRD 280

CONFORMAL MAPPING 14  
 HAYES, KAHANER, KELLNER

IDENT	MACHLNG	MACH
ENTRY	PIVOTL	MACH 10
.....	THIS MACHINE LANGUAGE ROUTINE REPLACES THE L**J	MACH 20
.....	LOOP IN SUBROUTINE CONFORM	MACH 30
	VFD 42/8LPIVOTL,18/4	MACH 40
PIVOTL	DATA 0	MACH 50
	SA1 B3	MACH 60
	SA2 B4+08	MACH 70
+	SB6 2	MACH 80
	IX3 X2-X1	MACH 90
+	SB7 X1-1	MACH 100
	SB6 X3-1	MACH 110
+	SA2 B1+B7	MACH 120
	SA3 B2+B7	MACH 130
	NG B5,PIV4	MACH 140
+	SA5 A2-1	MACH 150
	SA4 A3+1	MACH 160
+	SA1 A2-2	MACH 170
	SA2 A5	MACH 180
	BX0 X5	MACH 190
+	LT B5,B6,PIV2	MACH 200
PIV1	SA1 A1+B6	MACH 210
	SA2 A2+2	MACH 220
	FX6 X3*X0	MACH 230
+	FX7 X4*X5	MACH 240
	SA3 A3+B6	MACH 250
	SA4 A4+2	MACH 260
+	FX6 X1-X6	MACH 270
	FX7 X2-X7	MACH 280
	NX6 B0,X6	MACH 290
+	SB6 B5-B6	MACH 300
	NX7 B7,X7	MACH 310
	SA6 A1	MACH 320
	SA7 A2	MACH 330
	GE B5,B6,PIV1	MACH 340
PIV2	SA1 A1+B6	MACH 350
	SA2 A2+B6	MACH 360
+	FX6 X3*X0	MACH 370
	FX7 X4*X5	MACH 380
	FX6 X1-X6	MACH 390
+	FX7 X2-X7	MACH 400
	NX6 B0,X6	MACH 410
	SA6 A1	MACH 420
	NX7 B0,X7	MACH 430
	SA7 A2	MACH 440
	GE B0,B5,PIVOTL	MACH 450
	SA3 A3+B6	MACH 460
	SA1 A1+B6	MACH 470
	FX6 X3*X0	MACH 480
	FX6 X1-X6	MACH 490
	NX6 B0,X6	MACH 500
	SA6 A1	MACH 510
	EQ PIVOTL	MACH 520
PIV4	SA5 A2-1	MACH 530
	FX6 X5*X3	MACH 540
	FX6 X2-X6	MACH 550
	NX6 B0,X6	MACH 560
	SA6 A2	MACH 570
	EQ PIVOTL	MACH 580
	END	MACH 590
		MACH 600

Complex Zeros of Two Incomplete Riemann Zeta Functions . . . K. S. KÖLBIG	551
Some Results for $k! \pm 1$ and $2 \cdot 3 \cdot 5 \cdots p \pm 1$ . . . . . ALAN BORNING	567
On Thabit ibn Kurrah's Formula for Amicable Numbers . . . WALTER BORHO	571
On the Equation $\phi(n) = \phi(n + k)$ . . . . . M. LAL & P. GILLARD	579
REVIEWS AND DESCRIPTIONS OF TABLES AND BOOKS . . . . .	585
COLLATZ, MEINARDUS, UNGER & WERNER, Editors <b>17</b> , DEJON & HENRICI, Editors <b>19</b> , FETTIS & CASLIN <b>24</b> , KRYLOV & PAL'TSEV <b>18</b> , KUHN, Editor <b>20</b> , KUNTSEVICH, OLEKHOVICH & SHELEG <b>26</b> , LAPIDUS & SEINFELD <b>22</b> , PRABHU <b>25</b> , PSHENICHNYI <b>21</b> , VLADIMIROV <b>23</b> , WEINTRAUB <b>27</b> .	
TABLE ERRATA . . . . .	597
BYRD & FRIEDMAN <b>488</b> , COLLATZ <b>489</b> , ERDÉLYI, MAGNUS, OBERHET- TINGER & TRICOMI <b>490</b> , ERDÉLYI, MAGNUS, OBERHETTINGER & TRICOMI <b>491</b> , GRADSHTEYN & RYZHIK <b>492</b> , HENDERSON <b>493</b> .	
CORRIGENDUM . . . . .	601
GRADSHTEYN & RYZHIK	
NOTICE . . . . .	603
NEW JOURNAL: COMPUTERS & STRUCTURES	

---

The editorial committee would welcome readers' comments about this microfiche feature. Please send comments to Professor Eugene Isaacson, MATHEMATICS OF COMPUTATION, Courant Institute of Mathematical Sciences, New York University, 251 Mercer Street, New York, New York 10012.

# Mathematics of Computation

## TABLE OF CONTENTS

APRIL 1972

Block Five Diagonal Matrices and the Fast Numerical Solution of the Biharmonic Equation . . . . .	LOUIS BAUER & EDWARD L. REISS	311
An Improved Method for Numerical Conformal Mapping JOHN K. HAYES, DAVID K. KAHANER & RICHARD G. KELLNER		327
Coupled Harmonic Equations, SOR, and Chebyshev Acceleration L. W. EHRLICH		335
Norms of the Successive Overrelaxation Method . . . . .	DAVID R. KINCAID	345
Existence and Error Estimates for Solutions of a Discrete Analog of Non-linear Eigenvalue Problems . . . . .	R. B. SIMPSON	359
The Numerical Solution of Hyperbolic Systems Using Bicharacteristics R. L. JOHNSTON & S. K. PAL		377
Some Properties of a Class of Band Matrices W. D. HOSKINS & P. J. PONZO		393
Norms on Direct Sums and Tensor Products P. LANCASTER & H. K. FARAHAT		401
One-Step Piecewise Polynomial Galerkin Methods for Initial Value Problems BERNIE L. HULME		415
On the Solution of Systems of Equations by the Epsilon Algorithm of Wynn E. GEKELER		427
Anharmonic Frequency Analysis . . . . .	A. K. PAUL	437
An Integral Analogue of Taylor's Series and Its Use in Computing Fourier Transforms . . . . .	THOMAS J. OSLER	449
Monotonicity in Romberg Quadrature . . . . .	TORSTEN STRÖM	461
Minimal Error Constant Numerical Differentiation (N.D.) Formulas A. PELIOS & R. W. KLOPFENSTEIN		467
Minimum Norm Differentiation Formulas with Improved Roundoff Error Bounds . . . . .	DAVID K. KAHANER	477
Uniform Approximation of Vector-Valued Functions with a Constraint GENEVA G. BELFORD		487
Uniform Approximation Through Partitioning . . . . .	S. E. WEINSTEIN	493
Best $L_p$ Approximation . . . . .	S. W. KAHNG	505
A Method for Computing Bessel Function Integrals . . . . .	PETER LINZ	509
The Asymptotic Expansions of Hankel Transforms and Related Integrals ROBERT F. MACKINNON		515
On Evaluation of Moments of $K_\nu(t)/I_\nu(t)$ . .	CHIH-BING LING & JUNG LIN	529
A Note on the Evaluation of the Complementary Error Function D. B. HUNTER & T. REGAN		539
An Algorithm for Computing Logarithms and Arctangents . .	B. C. CARLSON	543