NUMERICAL COMPUTATION OF THE SCHWARZ-CHRISTOFFEL TRANSFORMATION

by

Lloyd Trefethen

STAN-CS-79-710
March 1979

COMPUTER SCIENCE DEPARTMENT
School of Humanities and Sciences
STANFORD UNIVERSITY

Numerical Computation of the Schwarz-Christoffel Transformation

Lloyd N. Trefethen*
Computer Science Department
Stanford University
Stanford, California 94305

*This work was supported in part by Office of Naval Research Contract N00014-75-C-1132 and in part by a National Science Foundation Graduate Fellowship.

Abstract

A program is described which computes Schwarz-Christoffel transformations that map the unit disk conformally onto the interior of a bounded or unbounded polygon in the complex plane. The inverse map is also computed. The computational problem is approached by setting up a nonlinear system of equations whose unknowns are essentially the "accessory parameters" z_k. This system is then solved with a packaged subroutine.

New features of this work include the evaluation of integrals within the disk rather than along the boundary, making possible the treatment of unbounded polygons; the use of a compound form of Gauss-Jacobi quadrature to evaluate the Schwarz-Christoffel integral, making possible high accuracy at reasonable cost; and the elimination of constraints in the nonlinear system by a simple change of variables.

Schwarz-Christoffel transformations may be applied to solve the Laplace and Poisson equations and related problems in two-dimensional domains with irregular or unbounded (but not curved or multiply connected) geometries. Computational examples are presented. The time required to solve the mapping problem is roughly proportional to N^3, where N is the number of vertices of the polygon. A typical set of computations to 8-place accuracy with $N \leq 10$ takes 1 to 10 seconds on an IBM 370/168.

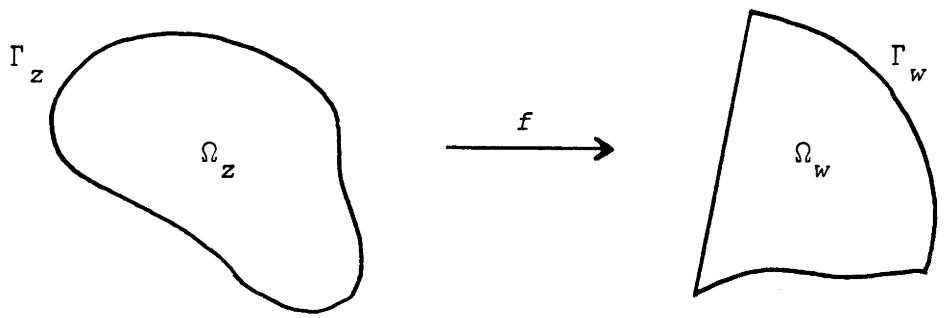
CONTENTS

FIGURES

I. INTRODUCTION

1. Conformal mapping and its applications

One of the classical applications of complex analysis is conformal mapping: the mapping of one open region in the complex plane C onto another by a function which is analytic and one-to-one and has a **nonzero** derivative everywhere. Such a map preserves angles between intersecting arcs in the domain and image regions; hence the name conformal. The Riemann Mapping Theorem asserts that any simply connected region in the plane which is not all of C can be mapped in this way onto any other such region. The **theorem** does not say what this mapping may look like, however, and the determination of particular conformal maps for particular mapping problems has been an active problem since at least 1850.



The usefulness of conformal mapping for applied problems stems from the fact that the Laplacian operator transforms in a simple way under a conformal map. Let j:C $\rightarrow$C map a region Ω_z in the z-plane conformally onto a region Ω_w in the w-plane, and let A, and Δ_w denote the Laplacian operators $\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$ and $\frac{\partial^2}{\partial u^2} + \frac{\partial^2}{\partial v^2}$, respectively, where $z = x + iy$ and $w = u + iv$. Then we may easily show,

$$\Delta_z \phi(z) = |f'(z)|^2 \, \Delta_w \phi(f^{-1}(w)) \tag{1.1}$$

for $\phi:\Omega_z \rightarrow \mathbf{R}$ suitably differentiable, A conformal map has $|f'(z)| > 0$ **every-**

where; thus from (1.1) it follows that if $\phi(z)$ is the solution to the Laplace equation $\Delta_z\phi = 0$ in Ω_z, subject to Dirichlet boundary condition6 $\phi(z) =$ g(z) on the boundary Γ_z, then $\psi(w) = \phi(f^{-1}(w))$ is a solution to the Laplace equation $\Delta_w\psi = 0$ in the image region $\Omega_w = f(\Omega_z)$, subject to the image boundary condition6 $\psi(w) = g(f^{-1}(w))$ on the boundary $\Gamma_w = f(\Gamma_z)$. (We have assumed that f map6 Γ_z bijectively onto the boundary of Ω_w. This is not always true, but it is true if both region6 are bounded by Jordan curves. See [Henrici, 1974], Thm. 5.10e.)

More generally, from (1.1) we can see that Poisson's equation, $\Delta_z\phi(z) =$ p(z), transforms under a conformal transformation into a Poisson equation in the w-plane with altered right hand side:

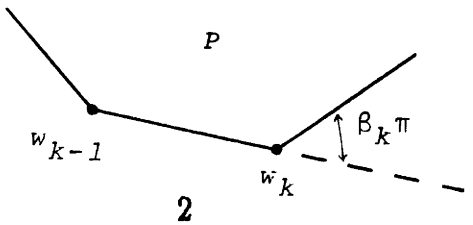
$$\Delta_w\psi(w) = |f'(f^{-1}(w))|^{-2}\,\rho(f^{-1}(w))\,. \tag{1.2}$$

Furthermore, more general boundary condition6 than Dirichlet also transform in a simple way. For example, the Neumann condition $\frac{\partial}{\partial n_z}\phi(z) =$ h(z), where $\frac{\partial}{\partial n_z}$ is a normal derivative in the x-plane, transforms to $\frac{\partial}{\partial n_w}\psi(w) = |f'(f^{-1}(w))|^{-1}h(f^{-1}(w))$. We do not pursue such possibilities further here; for a systematic treatment 6ee chapter VI of [Kantorovich & Krylov, 1958]. Some computed example6 are given in Section V.

Traditionally, conformal mapping ha6 been applied most often in two areas. One is plane electrostatics, where the electrostatic potential φ satisfies Laplace's equation. The other is irrotational, nonviscous fluid flow in the plane, which may be described in term6 of a velocity potential φ that also satisfies Laplace's equation.

2. The Schwarz-Christoffel transformation

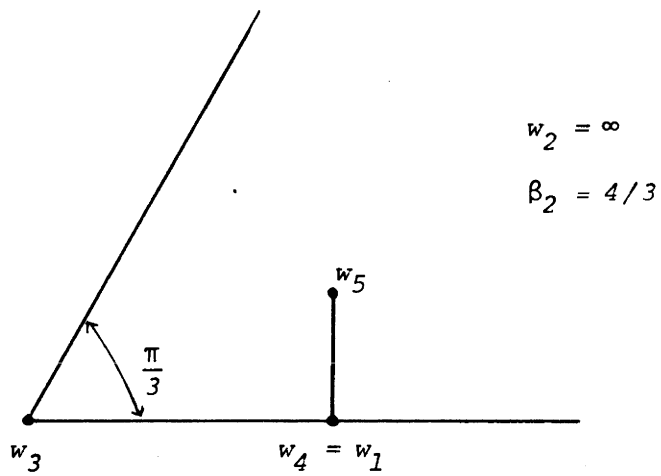
The problem of mapping one complex region conformally onto another is in general very difficult, but for the special case of polygonal region6 it can be greatly simplified, Suppose that we seek a conformal map from the unit disk in the x-plane to the interior of a polygon P in the w-plane whose vertices are $w_1, \ldots, w_N$, numbered in counterclockwise order, For each k, denote by $\beta_k\pi$ the exterior angle of P at w_k:



For any polygon we have a simple relationship among the numbers β_k:

$$\sum_{k=1}^{N} \beta_k = 2 . \tag{1.3}$$

If w_k is a finite vertex, we have $-1 \leq \beta_k < 1$. We need not require, however, that P be bounded, It may have a number of vertices at complex infinity, and the exterior angle6 corresponding to these may fall anywhere in the range $1 \leq \beta_k \leq 3$. Such angle6 are defined to be equal to 2π minus the external angle formed in the plane by the intersection of the two sides involved, if they are extended back away from infinity. The following example should illustrate what is meant by various values of β_k: it is a polygon with five vertices w_k (in this case $w_1 = w_4$), with corresponding values $(\beta_1, \ldots . \beta_5) = (\frac{1}{2}, \frac{4}{3}, \frac{2}{3}, \frac{1}{2}, -1)$:



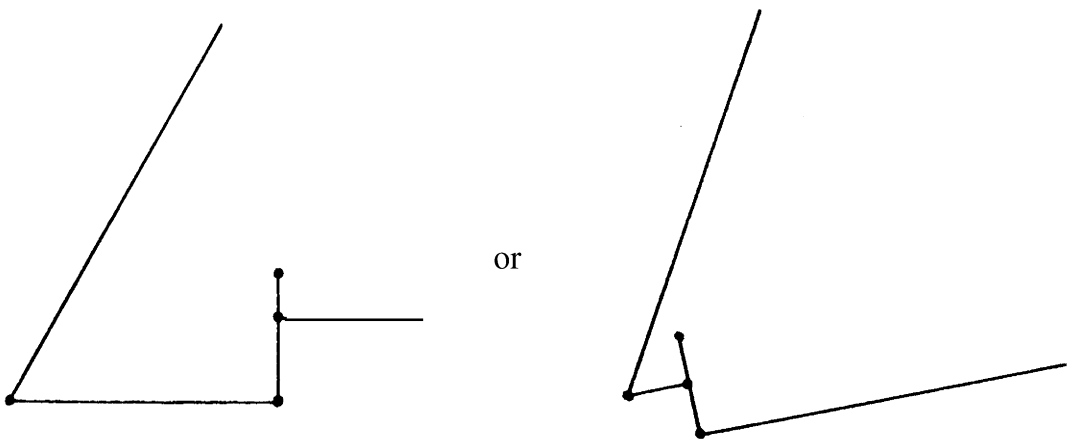
As always, (1.3) holds for this example.

Let us now pick at random N point6 z_k ("prevertices") in counterclockwise order around the unit circle and two complex constants C and w_c, and consider the Schwarz-Christoffel formula:

$$w = f(z) = w_c + C \int_0^z \prod_{k=1}^{N} \left(1 - \frac{z'}{z_k}\right)^{-\beta_k} dz' . \tag{1.4}$$

The quantities $(1 - z'/z_k)$ always lie in the disk $|w - 1| < 1$ for $|z| < 1$. Therefore, if we choose a branch of log(z) with a branch cut on the negative real axis by means of which to define the powers in (1.4), $w(z)$ defines an analytic function of z in the disk $|z| < 1$, continuous on $|z| \leq 1$ except possibly at the vertices z_k.

The Schwarz-Christoffel formula is chosen so as to force the image of the unit disk to have corners in it with the desired exterior angles $\beta_k \pi$. It is not hard to see from (1.4) that at each point z_k, the image $w(z)$ must turn a corner of precisely this angle. This is in keeping with our purpose of mapping the disk onto the interior of P. What the map will in general fail to do is to reproduce the lengths of sides of P correctly, and to be a one-to-one correspondence. For a suitable choice of parameters $\{z_k\}$, C, and w_c, the image under f of the unit disk might be, for example,



or

Only the angles are guaranteed to come out right.

The variables $z_1, \ldots, z_N$, C, and w_c are the *accessory* parameters of the Schwarz-Christoffel mapping problem. Our first problem-the parameter problem-is to determine values of the accessory parameters so that the lengths of sides of the image polygon do come out right. The central theorem of Schwarz-Christoffel transformations asserts that there always exists such a set of accessory parameters:

Theorem 1 (Schwarz-Christoffel transformation). Let D be a simply connected region in the complex plane bounded by a polygon P with vertices

$z_1, \ldots, z_N$ and exterior angle6 $\pi\beta_k$, where $-1 \leq \beta_k < 1$ if z_k is finite and $1 \leq \beta_k \leq 3$ if $z_k = \infty$. Then there exists an analytic function mapping the unit disk in the complex plane conformally onto D, and every such function may be written in the form (1.4).

Proof: [Henrici, 1974], Thm. 5.12e.

In fact, for any given polygon there is not just one but infinitely many such conformal mappings, To determine the map uniquely we may fix exactly three points z_k at will, or fix one point z_k and also fix the complex value w_c, or (a6 in a standard proof of the Riemann mapping theorem) fix w_c and the argument of the derivative $f'(0)$.

The simplicity of the explicit formula (1.4) is attractive. But because the problem of determining the accessory parameter6 is intractable analytically, application6 of it have almost always been restricted to problem6 simplified by having very few vertices or one or more axes of symmetry. General Schwarz-Christoffel map6 do not appear to have been used a6 a computational tool, although experiment6 have been made in computing them.

3. Numerical computation of the Schwarz-Christoffel Transformation

In the early day6 of computers, when *a* number of relatively pure mathematicians were growing interested in computational mathematics, the numerical computation of conformal map6 in general and Schwarz-Christoffel transformations in particular received a flurry of attention, As early as 1949, the National Bureau of Standard6 sponsored a symposium on numerical conformal mapping. It wa6 too early, however, for algorithms to result from this period which we could now consider practical.

In more recent years interest in numerical conformal mapping ha6 been modest. Gaier [1964] produced a comprehensive work describing method6 for various problems in constructive conformal mapping. For the Schwarz-Christoffel problem, he proposed determining the accessory parameters z_k by setting up a constrained nonlinear system of N — 3 equation6 relating (1.4) to the known distances $|w_k - w_j|$, and solving it iteratively by Newton's method [Gaier, p. 171]. Such a procedure ha6 been tried by at least three sets of people: [Meyer, 1979], [Howe, 1973], and [Vecheslavov&Kokoulin, 1973].

The present work follow6 Gaier and others in formulating the parameter problem a6 a constrained nonlinear system of equations, We believe that this is the first fully practical program for computing Schwarz-Christoffel

transformations, however, and the first which is capable of high accuracy without exorbitant cost.

One innovation which make6 accurate but cheap computations possible here is the use of a compound form of Gauss-Jacobi quadrature to evaluate the integral in (1.4). The evaluation of this integral is central in all Schwarz-Christoffel computations, both in determining the accessory parameters and in evaluating the map and it6 inverse once the accessory parameters are known. We have found that a straightforward application of Gauss-Jacobi quadrature, a6 6ome other6 have used, can achieve only very low accuracy in realistic problems, and we have developed a compound form of Gauss-Jacobi quadrature to get around this difficulty (see II.3).

A second innovation here is that the computation may be performed not just for bounded polygons, but for polygon6 with any number of vertices at infinity. This is made possible by taking the unit disk as the model domain rather than the upper half plane, which other6 have used, and evaluating complex contour integrals within the disk rather than only along the boundary. The ability to handle unbounded polygon6 is important for applications, since one of the attraction6 of conformal mapping is that it can reduce an unbounded problem domain to a bounded one.

The treatment of the constraints in the nonlinear system is a third new feature in this work. We have employed a simple change of variables to eliminate these constraints directly. This approach appear6 to be more efficient than other technique6 which have been tried (see [Howe,1973] and [Vecheslavov&Kokoulin,1973]), and eliminates the need for an initial guess of the accessory parameters.

Wc have depended in several place6 on the use of a sophisticated library of "black box" numerical routines. Library program6 come into play here for Gauss-Jacobi quadrature, for the solution of the nonlinear system, and for the solution of an ordinary differential equation. Others have been used in various experiment6 with applications. The Schwarz-Christoffel problem is essentially a simple problem numerically once the machinery is in place, but it is only in recent year6 that this kind of numerical machinery has begun to be broadly available,

II. DETERMINATION OF THE ACCESSORY PARAMETERS

1. Formulation as a con&rained nonlinear system (subroutine SCFUN)

The first matter to be settled in formulating the parameter problem numerically is, what parameter6 in the map (1.4) shall we fix at the outset to determine the Schwarz-Christoffel transformation uniquely? One choice would be to fix three of the boundary points z_k: say, $z_1 = 1$, $z_2 = i$, $z_N = -i$. This normalization ha6 the advantage that the resulting nonlinear system ha6 size only (N — 3)-by-(N — 3), which for a typical problem with N=8 may lead to a solution in less than half the time that a method involving an $(N-1)$-by-$(N-1)$ sy6tem requires. Nevertheless, we have chosen here to normalize by the conditions:

$$z_N = 1 \tag{2.1}$$

w_c = arbitrary point within P

which lead to an (N — 1)-by-$(N-1)$ system. This choice is motivated by considerations of numerical scaling: it allow6 the vertices to distribute themselves more evenly around the unit circle than they might otherwise. (An carlier version of the program mapped from the upper half plane instead of the unit disk, but wa6 rejected: once points z_k began appearing far from the origin at x = 10^4, scaling became a problem,) After a map has been computed according to any normalization, it is of course an easy matter to transform it analytically to a different domain or a different normalization by a Mobius transformation.

Now the nonlinear system must be formulated. The final map must satisfy N complex conditions,

$$w_k - w_c = C \int_0^{z_k} \prod_{j=1}^{N} \left(1 - \frac{z'}{z_j}\right)^{-\beta_j} dz', \qquad 1 \le k \le N. \tag{2.2}$$

These amount to 2N real condition6 to be satisfied, but they are heavily over-dctermincd, for the form of the Schwarz-Christoffel formula (1.4) guarantees that the angles will be correct no matter what accessory parameters are chosen. We must reduce the number of operative equations to N — 1. This

is a tricky matter when unbounded polygons are allowed, for one must be careful that enough information about the polygon P is retained that no degrees of freedom remain in the computed solution.

We proceed as follows. First, we require that every connected component of P contain at least one vertex w_k. Thus even an infinite straight boundary must be considered to contain a (degenerate) vertex. This restriction eliminates any translational degrees of freedom. Second, at least one component of P must in fact contain two finite vertices, and w_N and w_1 will be taken to be two such, This restriction eliminates rotational degrees of freedom.

Now define

$$C = (w_N - w_c) \Big/ \int_0^{z_N} \prod_{j=1}^{N} \left(1 - \frac{z'}{z_j}\right)^{-\beta_j} dz' , \tag{2.3}$$

where $z_N = 1$ is fixed permanently by (2.1). Next, impose the complex condition (real equations 1,2)

$$w_1 - w_c = C \int_0^{z_1} \prod_{j=1}^{N} \left(1 - \frac{z'}{z_j}\right)^{-\beta_j} dz' . \tag{2.4a}$$

This amounts to two real equations to be satisfied.

Denote by $\Gamma_1, \ldots . \Gamma_m$ the distinct connected components of P, numbered in counterclockwise order. For each $\ell \geq 2$, impose one more complex condition: if z_{k_ℓ} is the last vertex of Γ_ℓ in the counterclockwise direction, then (real equations $3,4,\ldots,2m$)

$$w_{k_\ell} - w_c = C \int_0^{z_{k_\ell}} \prod_{j=1}^{N} \left(1 - \frac{z'}{z_j}\right)^{-\beta_j} dz' . \tag{2.4b}$$

Finally, $N - 2m - 1$ conditions of side length are imposed. For each pair (z_k, z_{k+1}) beginning at $k = 1$ and moving counterclockwise, where both vertices are finite, we require (real equations $2m+1,\ldots,N-1$)

$$|w_{k+1} - w_k| = \left| C \int_{z_k}^{z_{k+1}} \prod_{j=1}^{N} \left(1 - \frac{z'}{z_j}\right)^{-\beta_j} dz' \right| \tag{2.4c}$$

until a total of N — 1 conditions have been imposed. If P contains at least one vertex at infinity, then every bounded side will have been represented in a condition of the form (2.4c) except for the side (w_N, w_1), which is already taken care of by (2.1) and (2.4a). If P is bounded, then the last two sides in counterclockwise order—(w_{N-2}, w_{N-1}) and (w_{N-1}, w_N)—will not be 60 represented.

We have not stated over what contours the integrals of eqs. (2.4) are defined. This does not matter mathematically, as the integrand is analytic, but it may matter numerically. In this work we have evaluated them always over the straight line segment between the two endpoints, a procedure which poses no domain problem6 since the unit disk is strictly convex. Figure 2.1 illustrates what contours are involved in computing the integrals in (2.3) and (2.4), for a sample case with N = 10, $m = 3$.

The nonlinear system is now determined, and its unique solution will give the unknown parameters C and $z_1, \ldots, z_{N-1}$ for the Schwarz-Christoffel mapping. We must, however, take notice of two special cases in which the solution is not completely determined by eqs. (2.4). It was remarked that if P is bounded, then nowhere in eqs. (2.4) does the point w_{N-1} appear. If $\beta_{N-1} \neq -1$ or 0, then this omission is of no consequence, for the geometry of the problem forces w_{N-1} to be correct. If $\beta_{N-1} = 0$ or -1, however, then w_{N-1} is not determined a priori. The former case is of little consequence, for since $\beta_{N-1} = 0$ the value taken for z_{N-1} has no effect on the computed mapping, as may be seen in (1.4), nor is there any purpose in including w_{N-1} among the vertices of P in the first place. (Still, there may be problems in solving the system (2.4) numerically, for it is now underdetermined.) The latter case, $\beta_{N-1} = -1$, is more serious, and must be avoided in the numbering of the vertices w_k.

2. Transformation to an unconstrained system (subroutine YZTRAN)

The nonlinear system (2.4) ostensibly involves N — 1 complex unknown points $z_1, \ldots, z_{N-1}$ on the unit circle, In dealing with such a system, we naturally begin by considering not the points z_k themselves, but their arguments θ_k, given by

$$z_k = e^{i\theta_k}, \qquad 0 < \theta_k \leq 2\pi . \tag{2.5}$$

Now the system depends on N — 1 real unknowns, and the solution in terms of the θ_k is fully determined.

However, the system (2.4) as it stands must be subject to a set of strict

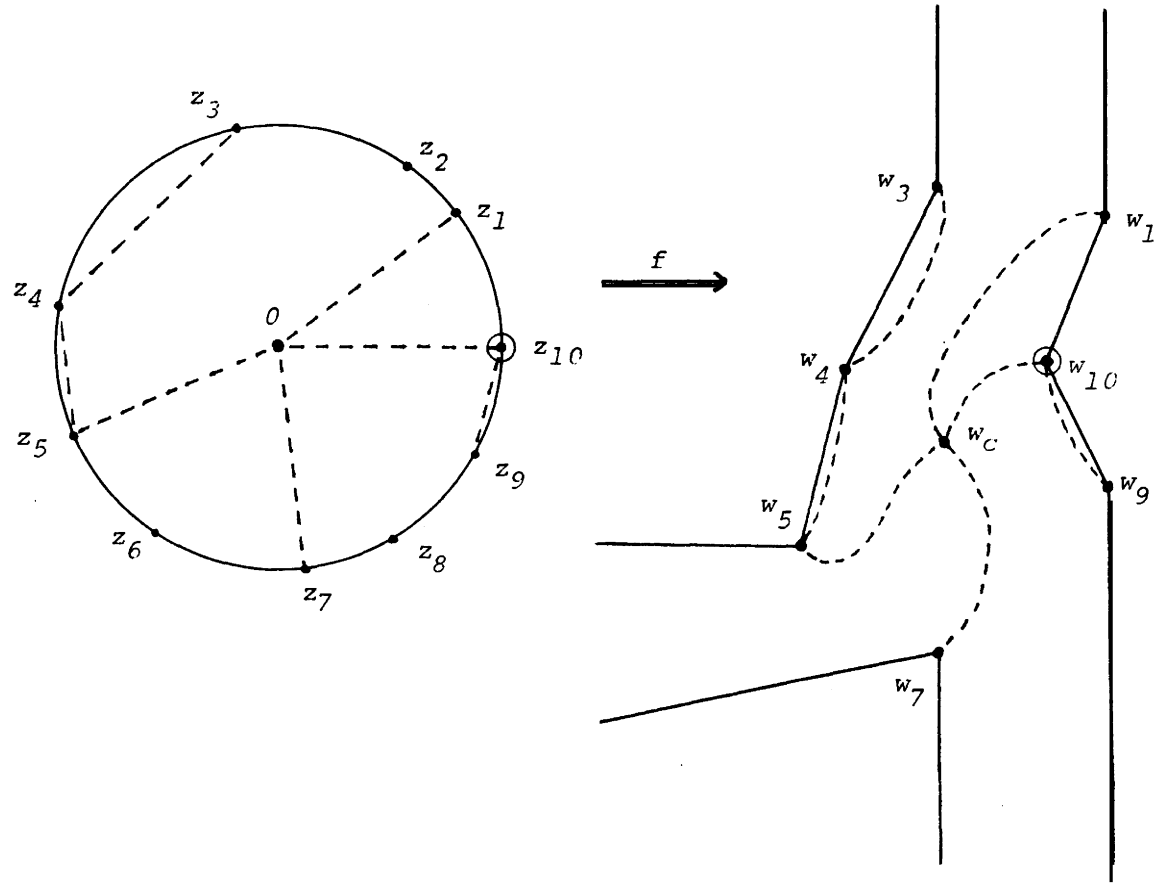


Figure 2.1 – Contours of integration within the disk. A sample Schwarz-Christoffel problem is shown with $N = 10$ vertices of which $m = 3$ vertices are at infinity, illustrating what integrals are computed to evaluate the system (2.4):

- 1 radial integral along $(0 - z_{10})$ defines C (eq. 2.3)
- 1 radial integral along $(0 - z_1)$ determines two real equations to fix w_1 (eq. 2.4a)
- 2 radial integrals along $(0 - z_5)$ and $(0 - z_7)$ determine four real equations to fix w_5 and w_7 (eq. 2.4b)
- 3 chordal integrals along $(z_3 - z_4)$, $(z_4 - z_5)$, and $(z_9 - z_{10})$ determine three real equations to fix $|w_4 - w_3|$, $|w_5 - w_4|$, and $|w_{10} - w_9|$ (eq. 2.4c)

TOTAL: $N - 1 = 9$ real equations

inequality constraints,

$$0 < \theta_k < \theta_{k+1}, \qquad 1 \le k \le N-1, \tag{2.6}$$

which embody the fact that the vertices z_k must lie in ascending order counterclockwise around the unit circle. To solve the system numerically, it is desirable to eliminate these constraints somehow. We do this by transforming eqs. (2.4) to a system in N — 1 variables $y_1, \ldots, y_{N-1}$, defined by the formula

$$y_k = \log \frac{\theta_k - \theta_{k-1}}{\theta_{k+1} - \theta_k}, \qquad 1 \le k \le N-1, \tag{2.7}$$

where θ_0 and θ_N, two different names for the argument of $z_N = 1$, are taken for convenience as 0 and 2π, respectively.

At each iterative step in the solution of the nonlinear system (2.4), we begin by computing a set of angles $\{\theta_k\}$ and then vertices $\{z_k\}$ from the current trial set $\{y_k\}$. This is easy to do, though not immediate since the equations (2.7) are coupled. In this way the problem is reduced to one of solving an unconstrained nonlinear system of equations in N — 1 real variables.

3. Integration by compound Gauss-Jacobi quadrature (subroutine ZQUAD)

The central computation in solving the parameter problem, and indeed in all Schwarz-Christoffel computations, is the numerical evaluation of the Schwarz-Christoffel integral (1.4) along some path of integration. Typically one or both endpoints of this path are prevertices z_k on the unit circle, and in this case a singularity of the form $(1 - z/z_k)^{-\beta}$ is present in the integrand at one or both endpoints.

A natural way to compute such integrals quickly is by means of Gauss-Jacobi quadrature (see [Davis & Rabinowitz, 1975], p. 75). A Gauss-Jacobi quadrature formula is a sum $\sum_{i=1}^{NPTS} w_i f(x_i)$, where the weights w_i and nodes x_i have been chosen in such a way that the formula computes the integral $\int_{-1}^{+1} f(x)(1-x)^{\alpha}(1+x)^{\beta}\,dx$ exactly for $f(x)$ a polynomial of as high a degree as possible. Thus Gauss-Jacobi quadrature is a generalization of pure Gaussian quadrature to the case where singularities of the general form $(1-x)^{\alpha}(1+x)^{\beta}$ $(\alpha, \beta > -1)$ are present. The required nodes and weights can be computed numerically; we have used the program GAUSSQ by Golub and Welsch [Golub & Welsch, 1969] for this purpose.

Gauss-Jacobi quadrature appears made-to-order for the Schwarz-Christoffel problem, and at least two previous experimenters have used it or or a closely related technique ([Howe,1973], [Vecheslavov & Kokoulin,1973]). We began by doing the same, and got good results for many polygons with a small number of vertices, In general, however, we found this method of integration very inaccurate. For a typical sample problem with N = 12 and $NPTS = 16$, it produced integrals accurate to only about 10^{-2}, and it does much worse if one chooses polygons designed to be troublesome.

What goes wrong is a matter of resolution. Consider a problem like the one shown in Figure 2.2. We wish to compute the integral (1.4) along the segment from z_k to some point p. (In the parameter problem p might be 0 or z_{k-1}; in later computations it might be any point in the disk.) Now direct application of a Gauss-Jacobi formula will involve sampling the integrand at only $NPTS$ nodes between z_k and p. If the singularity z_{k+1} is so close to the path of integration that the distance $\epsilon = |z_{k+1} - z_k|$ is comparable to the distance between nodes, then obviously the Gauss-Jacobi formula will yield a very poor result. It turns out that in Schwarz-Christoffel problems the correct spacing of prevertices z_k around the unit circle is typically very irregular, so the appearance of this problem of resolution is the rule, not the exception. (See examples in V.)

To maintain high accuracy without giving *up* much speed, we have switched to a kind of compound Gauss-Jacobi quadrature (see [Davis & Rabinowitz, 1975], p. 56). We adopt, somewhat arbitrarily, the following quadrature principle:

> *No* singularity z_k *shall lie* closer to an interval of integration than half *the* length of that interval,

To achieve this goal, the quadrature subroutine ZQUAD must be able to divide an interval of integration into shorter subintervals as necessary, working from the endpoints in. On the short subinterval adjacent to the endpoint Gauss-Jacobi quadrature will be applied; on the longer interval (or intervals) away from the endpoint pure Gaussian quadrature will be applied. The effect of this procedure is that number of integrand evaluations required to achieve a given accuracy is reduced from $O(\frac{1}{\epsilon})$ to $O(\log_2 \frac{1}{\epsilon})$.

Figure 2.2 shows the intervals of integration that come into play in compound Gauss-Jacobi quadrature. For a plot comparing the accuracy of simple and compound Gauss-Jacobi quadrature in another typical problem, see IV.1.

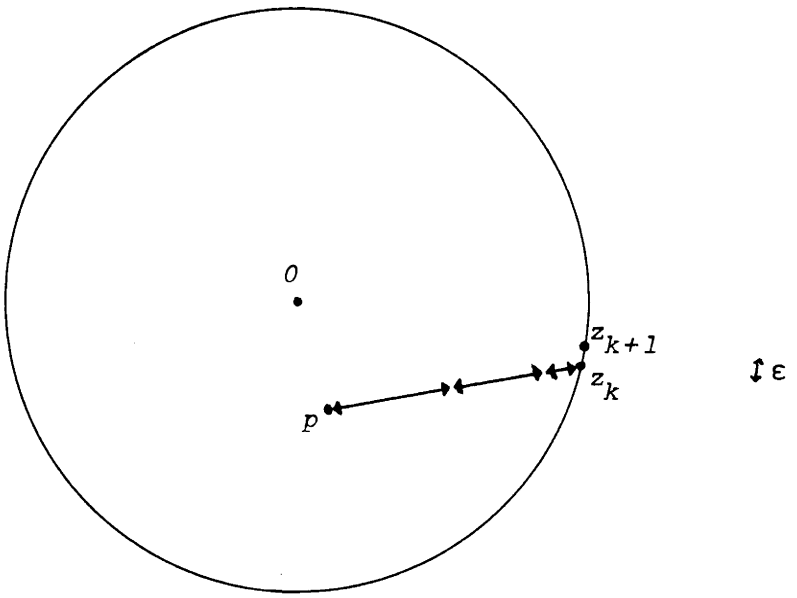


Figure 2.2 - Compound Gauss-Jacobi quadrature. Division of an interval of integration into subintervals to maintain desired resolution.

With the use of compound Gauss-Jacobi quadrature, we now achieve high accuracy in little more than the time that direct Gauss-Jacobi quadrature takes. This is possible because only a minority of integrals have a singularity close enough that subdivision of the interval of integration is required. In the 12-vertex example mentioned above, the switch to compound Gauss-Jacobi integration decreased the error from 10^{-2} to $2 \cdot 10^{-7}$.

There remains one circumstance in which integration by compound Gauss-Jacobi quadrature as described here is unsuccessful. This is the case of an integration interval with one endpoint quite near to some prevertex z_k corresponding to a vertex $w_k = \infty$. We cannot evaluate such an integral by considering an interval which begins at z_k, for the integral would then be infinite. The proper approach to this problem is probably the use of integration by parts, which can reduce the singular integrand to one that is not infinite Depending on the angle β_k, one to three applications of integration by parts will be needed to achieve this. We have not implemented this procedure.

The subtlety of the integration problem in Schwarz-Christoffel computations is worth emphasizing. It is customary to dispatch the integration problem as quickly as possible, in order to concentrate on the "difficult" ques-

Gions: computation of accessory parameters and inversion of the Schwarz-Christoffel map. We believe, however, that the more primary problem of computing Schwarz-Christoffel integrals-the "forward" problem-should always remain a central concern. Any numerical approach to the parameter problem or the inversion problem is likely to employ an iterative scheme which depends at each step on an evaluation of the integral (1.4), and so the results can only be as accurate as that evaluation.

4. Solution of system by packaged solver (subroutine SCSOLV)

The unconstrained nonlinear system is now in place and ready to be solved, For this purpose we employ a library subroutine: NSOlA, by M.J.D. Powell ([Powell, 1968]) which uses a steepest descent search in early iterations if necessary followed by a variant of Newton's method later on. (The routine does not use analytic derivatives.) It is assumed that a variety of other routines would have served comparably well,

We make no attempt to tailor the numerical solution procedure to the particular Schwarz-Christoffel problem under consideration, In particular, all iterations begin with the trivial initial estimate $y_k = 0 \quad (1 \leq k \leq N - 1)$. This corresponds to trial vertices spaced evenly around the unit circle. The following input parameters to NSOlA have generally remained fixed: DSTEP$=10^{-8}$ (step size used to estimate derivatives by finite differences), DMAX $=$ 10 (maximum step size), MAXFUN $= 15(N - 1)$ (maximum number of iterations).

A fourth parameter, EPS, defines the convergence criterion-how large a function vector (square root of sum of squares of functions values) will be considered to be satisfactorily close to zero. We have most often taken 10^{-8} or 10^{-14} here. The choice of EPS is not very critical, however, as convergence in NSOlA is generally quite fast in the later stages.

In the course of this work about a hundred Schwarz-Christoffel transformations have been computed, ranging in complexity from N $=$ 3 to N $=$ 18, NSOlA has converged successfully to an accurate solution in all of these trials. Section V.1 gives a series of plots showing this convergence graphically for a simple example,

III. COMPUTATION OF THE S-C MAP AND ITSINVERSE

Determining the accessory parameters is the most formidable task in computing numerical Schwarz-Christoffel transformations, Once this is done, evaluation of the map and of its inverse follow relatively easily. The foundation of these computations continues to be compound Gauss-Jacobi quadrature.

1. From disk to polygon: w = $w(z)$ (subroutine WSC)

To evaluate the forward map $w(z)$ for a given point z in the disk or on the circle, we must compute the integral

$$w = w_0 + C\int_{z_0}^{z}\prod_{j=1}^{N}\left(1-\frac{z'}{z_j}\right)^{-\beta_j}dz' \qquad (3.1)$$

with $w_0 = w(z_0)$, where the endpoint z_0 may be any point in the closed disk at which the image $w(z_0)$ is known and not infinite. Three possible choices for z_0 suggest themselves-

(1) $z_0 = 0$; hence $w_0 = w_c$;

(2) $z_0 = z_k$ for some k; hence $w_0 = w_k$, a vertex of P;

(3) z_0 = some other point in the disk at which w has previously been computed.

In cases (1) and (3), neither endpoint has a singularity, and an evaluation of (3.1) by compound Gauss-Jacobi quadrature reduces to the use of compound Gauss quadrature. In case (2) a singularity of the form $(1 - z/z_k)^{-\beta}$ is present at one of the endpoints and the other endpoint has no singularity.

The best rule for computing $w(z)$ is: if z is close to a singular point z_k (but not one with $w_k = \infty$), use method (2); otherwise, use method (1). In either case we employ compound Gauss-Jacobi quadrature, taking normally the same number of nodes as was used in solving the parameter problem. By this procedure we evaluate $w(z)$ readily to "full" accuracy-that is, the accuracy to which the accessory parameters have been computed, which is directly related to the number of points chosen for Gauss-Jacobi quadrature (see IV.1). Quadrature nodes and weights need only be computed once, of course.

We should emphasize that even in the vicinity of a singularity z_k, the evaluation of the map $w = w(z)$ is inherently very accurate. This very satisfactory treatment of singular vertices is a considerable attraction of the Schwarz-Christoffel approach for solving problems of Laplace type. In particular, in a potential problem the Schwarz-Christoffel transformation "automatically" handles the singularities correctly at any number of reentrant corners,

2. From polygon to disk: $z = z(w)$ (subroutine ZSC)

For computing the inverse mapping $z = x(w)$ at least two possibilities exist, both of them quite powerful, The most straightforward approach is to view the formula $w(z) = w$ as a nonlinear equation to be solved for z, given some fixed value w. The solution may then be found iteratively by Newton's method or a related device, $w(z)$ should be evaluated at each step of such a process by compound Gauss-Jacobi quadrature along a straight line segment whose initial point remains fixed throughout the iteration.

An alternative approach is to invert the Schwarz-Christoffel formula,

$$\frac{dw}{dz} = C \prod_{k=1}^{N} \left(1 - \frac{z}{z_k}\right)^{-\beta_k},$$

to yield the formula

$$\frac{dz}{dw} = \frac{1}{C} \prod_{k=1}^{N} \left(1 - \frac{z}{z_k}\right)^{+\beta_k}. \tag{3.2}$$

This inversion is possible because $w = w(z)$ is a conformal mapping, which means $|dw/dz| > 0$ everywhere. (3.2) may now be thought of as an ordinary

differential equation (o.d.e.),

$$\frac{dz}{dw} = g(w, z) , \tag{3.3}$$

in one complex variable w. If a pair of values (z_0, w_0) is known and the new value $z = z(w)$ is sought, then z may be computed by applying a numerical o.d.e. solver to the problem (3.3), taking as a path of integration any curve from w_0 to w which lies within the polygon P.

In our program we have chosen to combine these two methods, using the second method to generate an initial estimate for use in the first. We begin with the o.d.e. formulation, using the code ODE by Shampine and Gordon, and for convenience we integrate whenever possible along the straight line segment from w_c to w. (ODE, like most o.d.e. codes, is written for problem6 in real arithmetic, so that we must first express (3.2) as a system of first-order o.d.e.'s in two real variables.) Since P may not be convex, more than one line segment step may be required to get from w_0 to w in this way. It will not do to take $w_0 = w_k$ for some vertex w_k without special care, because (3.2) is singular at w_k.

From ODE we get a rough estimate $\tilde{z}$ of z(w), accurate to roughly 10^{-2}. This estimate is now used a6 an initial guess in a Newton iteration to solve the equation $w(z) = w$. This method is faster than the o.d.e. formulation for getting a high-accuracy answer, More important, it is based on the central Gauss-Jacobi quadrature routine, unlike the o.d.e. computation.

In summary, we compute the inverse map $z = z(w)$ rapidly to full accuracy by the following steps:

(1) Solve (3.2) to low accuracy with package ODE, integrating whenever possible along the line segment from w_c to w; call the result $\tilde{z}$;

(2) Solve the equation $w(z) = w$ for z by Newton's method, using $\tilde{z}$ a6 an initial guess.

IV. ACCURACY AND SPEED

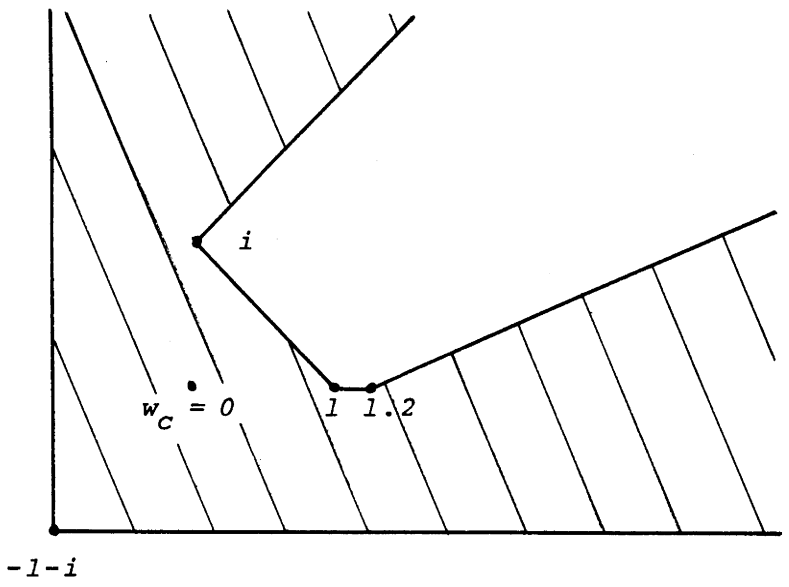
1. Accuracy

The central computational step is the evaluation of the Schwarz-Christoffel integral, and the accuracy of this evaluation normally determine6 the accuracy of the overall computation. A6 a consequence of the quadrature principle adopted in II.3—that no quadrature interval shall be longer than twice the distance to the nearest singularity z_k—the compound Gauss-Jacobi formulation achieve6 essentially the full accuracy typical of Gaussian quadrature rules operating upon smooth integrands. That is, the number of digits of accuracy is closely proportional to NPTS, the number of quadrature node6 per half-interval, with a very satisfactory porportionality constant in practice of approximately 1.

It is important not only to be capable of high accuracy, but to be able to measure how much accuracy one has in fact achieved in a given computation, To do this we employ a subroutine TEST, which is regularly called immediately after the parameter problem is solved. Given a computed set of accessory parameter6 C and $\{z_k\}$, TEST computes the distances $|w_k - w_c|$ for each $w_k \neq$ oo and the distances $|w_{k-1} - w_{k+1}|$ for each $w_k = \infty$, making use of the standard subroutine ZQUAD for compound Gauss-Jacobi quadrature. The numbers obtained are compared with the cxacf distance6 specified by the geometry of the polygon, and the maximum error, RADEMX, is printed as an indication of the magnitude of error6 in the converged solution. It is now probable that subsequent computations of w(z) or z(w) will have errors no greater than roughly RADEMX,

Most often wc have chosen to use an 8-point quadrature formula. Since each interval of integration is initially divided in half by subroutine ZQUAD, this mcan6 in reality at least 16 node6 per integration. With this choice RADEMX consistently ha6 magnitude $\sim 10^{-8}$ for polygons on the scale of unity.

Figure 4.1 gives an indication of the relationship between number of quadrature node6 and error RADEMX; it shows RADEMX a6 a function of NPTS for a 6-gon which is shown at the top of the next page. Two curve6



are shown: one for simple Gauss-Jacobi quadrature, and one for compound Gauss-Jacobi quadrature. The exact quantities here should not be taken too seriously; examples could easily have been devised to make the difference in performance of the two quadrature methods much smaller or much greater.

2. Speed

Any application of **Schwarz-Christoffel** transformations consists of a sequence of steps:

INIT – set up problem

QINIT – compute quadrature nodes and weights

SCSOLV – solve parameter problem

TEST – estimate accuracy of solution

ZSC, WSC, etc. – compute forward and inverse transformation6 in various applications

Among these tasks INIT, QINIT, and TEST all take negligible amounts of time relative to the other computations: typically less than 0.1 secs. on the IBM 370/168 for INIT and QINIT, and for TEST a variable time that

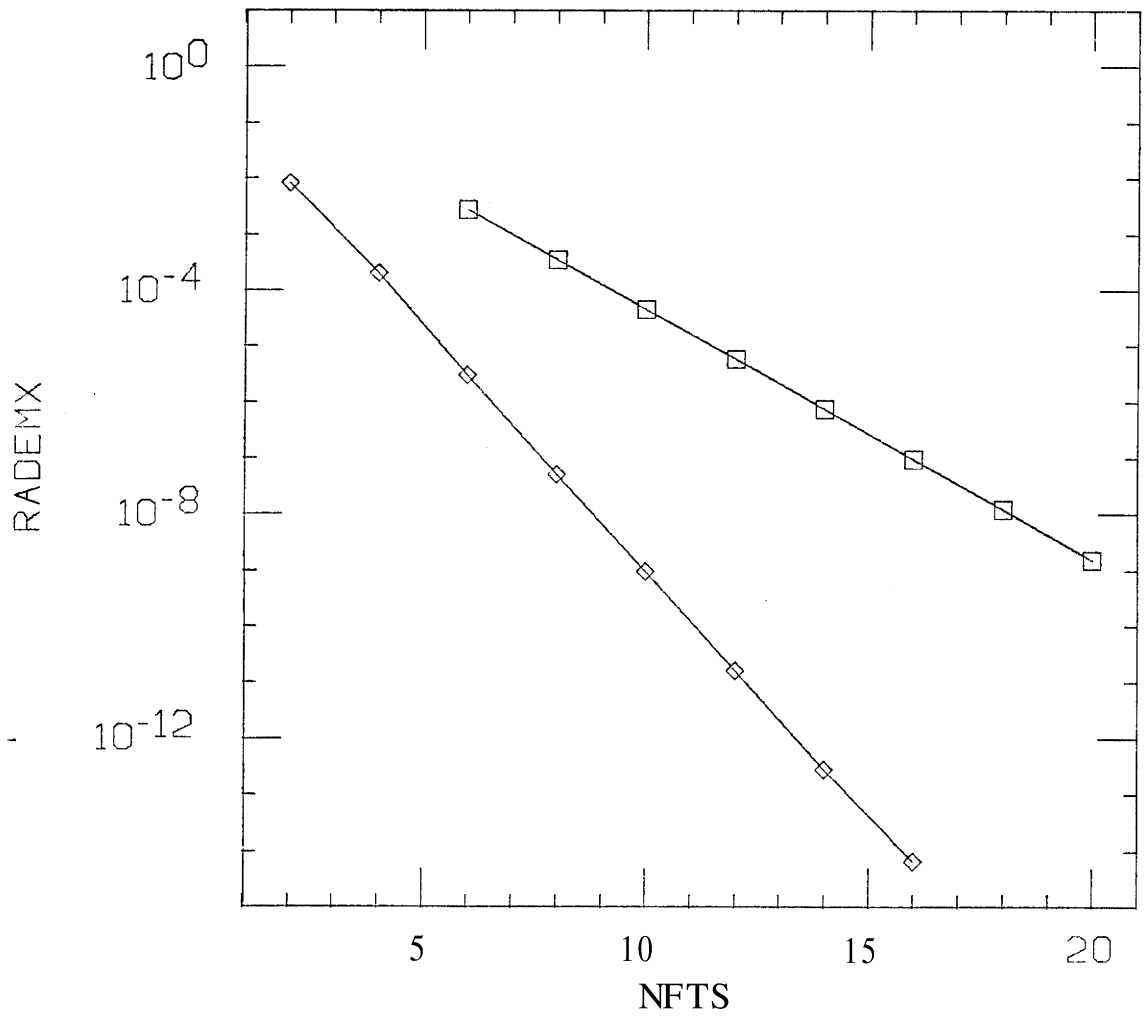


Figure 4.1 – Quadrature accuracy as a function of number of nodes. The error estimate RADEMX is plotted as a function of NPTS for the polygon shown on p. 19. The upper and lower curves correspond to simple Gauss-Jacobi and compound Gauss-Jacobi quadrature, respectively.

is usually less than 5% of the time required by SCSOLV, What remains are three main time consumers: SCSOLV, ZSC, and WSC.

We begin with WSC, which performs the central evaluation of (1.4) by compound Gauss-Jacobi quadrature. This evaluation takes time proportional to NPTS (the number of quadrature nodes) and to N (the number of vertices). The first proportionality is obvious, and the second results from the fact that the integrand of (1.4) is an N-fold product. Very roughly, we may estimate

$$\text{time to solve } w = w(z): \qquad 0.25 \cdot NPTS \cdot N \ msec. \tag{4.1a}$$

for double precision computations on the IBM 370/168. Taking a typical value of NPTS=8, which normally leads to &digit accuracy, (4.1a) may be rewritten

$$\boxed{\text{time to solve } w = w(z): \qquad 2N \ msec.} \tag{4.1b}$$

For the minority of cases in which the interval must be subdivided to maintain the required resolution, these figures will be larger.

To estimate the time required to solve the parameter problem, we combine (4.1) with an estimate of how many integrals must be computed in the course of solving this problem. To begin with, at each iteration about N integrals arc required by NS01A (the exact number depends on the number of vertices at infinity). On top of this, it is a fair estimate to say that 4N iterations will be required by NSOlA to achieve a high-accuracy solution. We are therefore led to the estimate

$$\text{time } to \text{ solve parameter } problem: \qquad NPTS \cdot N^3 \ msec. \tag{4.2a}$$

or, taking again NPTS=8,

$$\boxed{\text{time } to \ solve \ parameter \ problem: \qquad 8N^3 \ msec.} \tag{4.2b}$$

These estimates correspond fairly well with observed computation times for the parameter problem: two problems with N = 5 and N = 18 may be expected to take about 1 and 50 seconds, respectively. It is clear that computing a Schwarz-Christoffel transformation becomes quite a sizeable problem for polygons with more than ten vertices, In particular, such computations are much too time-consuming for it to be practical to approximate a curved domain by a polygon with a large number of vertices.

Finally, we must consider the time taken by subroutine **ZSC** to invert the Schwarz-Christoffel map. This too is proportional to **NPTS,** and quite problem dependent. We estimate very roughly:

$$\text{time to solve } z = z(w): \qquad NPTS \cdot N \text{ msec.} \tag{4.3a}$$

or, with **NPTS=8,**

$$\boxed{\text{time to solve } z = z(w): \qquad 8N \text{ msec.}} \tag{4.3b}$$

Note that inverting the **Schwarz-Christoffel** map is only about four times as time-consuming as computing it in the forward direction.

In practice, computational applications will vary considerably in the use they make of a Schwarz-Christoffel transformation once the parameter problem is solved. If only a few dozen applications of **ZSC** or WSC are required, then the computational time for solving the parameter problem will dominate. If thousands of such computations are needed, on the other hand, then the parameter problem may become relatively insignificant. The latter situation is most likely to hold when plotting is being done, or when a high-accuracy solution in the model domain is to be computed by means of finite differences.

In summary, high accuracy is cheap in Schwarz-Christoffel transformations; what consumes time is solving problems involving a large number of vertices.

V. COMPUTED EXAMPLES AND APPLICATIONS

1. Iterative *process* for a single example

Figure 5.1 shows graphically the process of convergence from the initial estimate in an example involving a 4-gon. Routine NS01A begins by evaluating the function vector (2.4) at the initial guess, then at each of N — 1 input vectors determined by perturbing the initial guess by the small quantity DSTEP in each component, As a result, the first N pictures always look almost alike, which is why the series shown begins at NEVAL=4 rather than NEVAL=1. Each plot shows the current image polygon together with the images of concentric circles in the unit disk (which appear as "contours") and the images of radii leading from the center of the disk to the current prever t ices z_k.

These pictures have a beautiful bonus feature about them: they may be interpreted as showing not only the image polygon but simultaneously the domain disk, including the prevertices z_k along the unit circle. To see this, look at one of the inner "contour" curves, one which is apparently circular, and the radii within it. Since w $= w(z)$ is a conformal map within the interior of the disk, the radii visible in this circle must intersect at the same angles as their preimages in the domain disk, Thus the inner part of any one of these image plots is a faithful representation on a small scale of the circular domain, We see in Figure 5.1 that the prevertices are equally spaced around the unit circle initially (NEVAL $= 4$), but move rapidly to a very uneven distribution. This behavior, which is typical, indicates why the USC of a compound form of Gauss-Jacobi quadrature is so important (see rr.3).

The sum-of-squares error in solving the nonlinear system is plotted as a function of iteration number in Figure 5.2, for the same 4-vertex example, Convergence is more or less quadratic, as one would expect for Newton's method. The irregularity at iteration 19 is caused by the finite difference step size of 10^{-8} used to estimate derivatives, and would have been repeated at each alternate step thereafter if the iteration had not terminated.

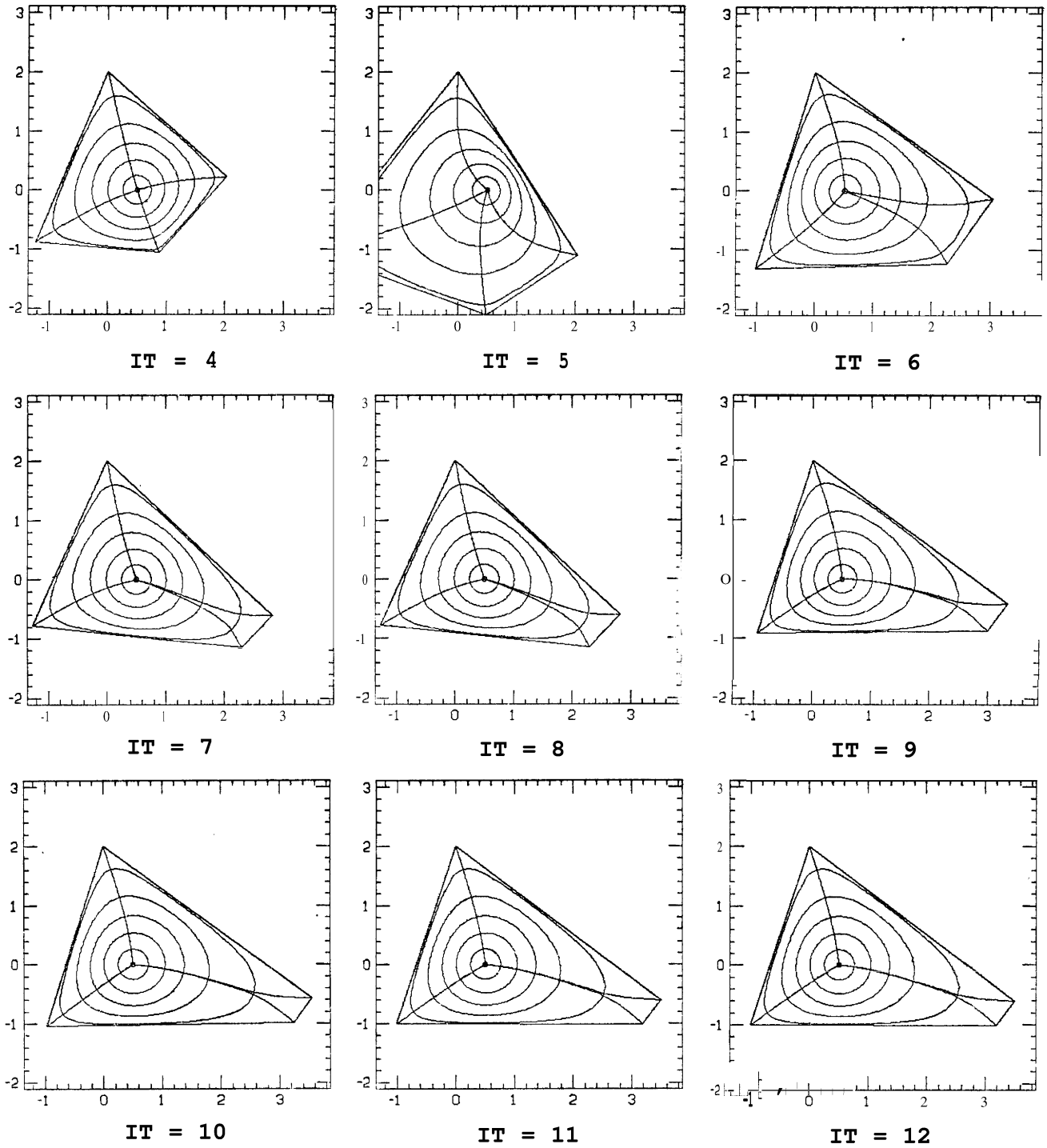


Figure 5.1 – Convergence to a solution of the parameter problem. Plots show the current image polygon at each step as the accessory parameters $\{z_k\}$ and C are determined iteratively, for a problem with $N = 4$.

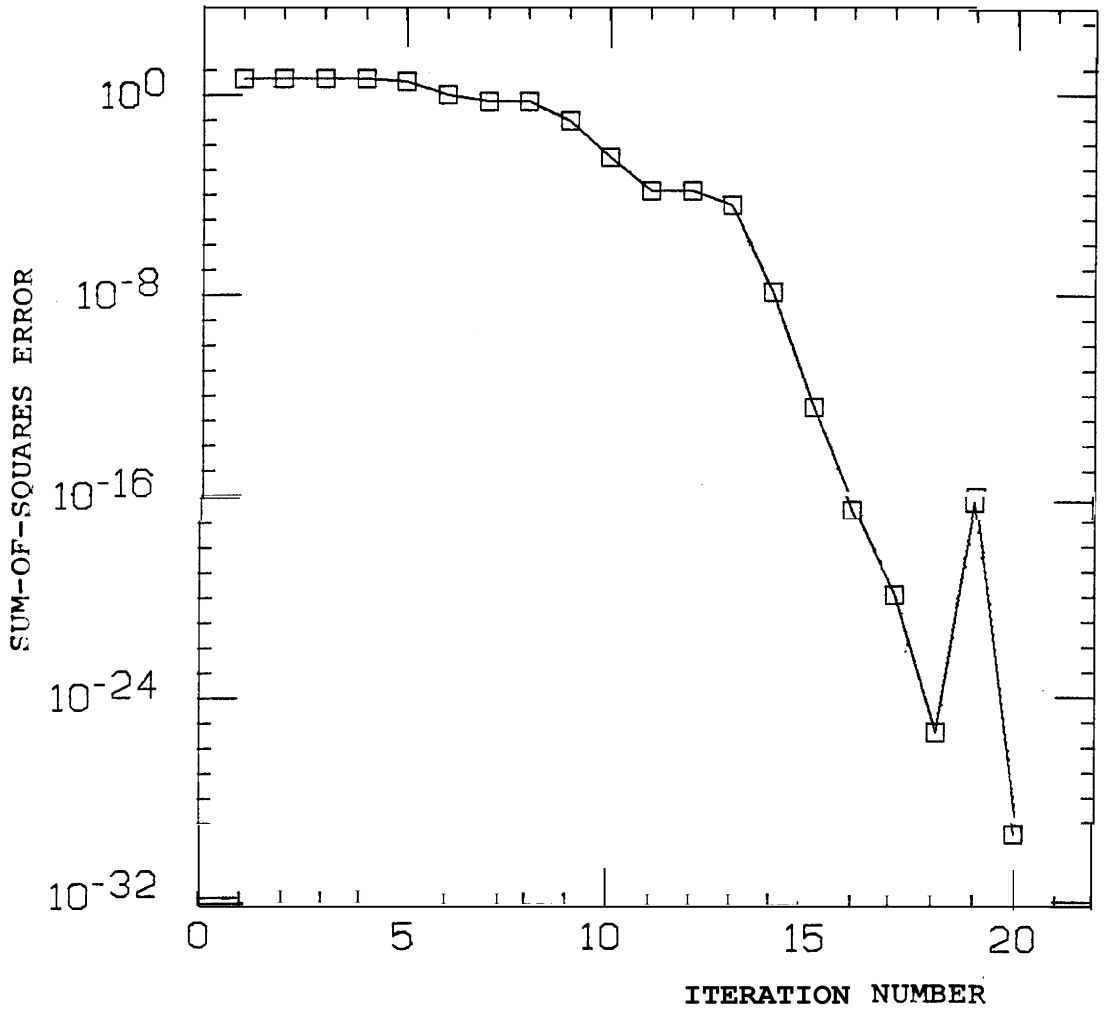


Figure 5.2 – Rate of convergence. Sum-of-squares error in the nonlinear system (2.4) as a function of iteration number, for the same problem a8 in Figure 5.1.

2. Sample Schwarz-Christoffel maps

Figures 5.3 and 5.4 show plots of computed Schwarz-Christoffel maps for representative problems. The polygons of Figure 5.3 are bounded and those of Figure 5.4 are unbounded, Observe that contour lines bend tightly around reentrant corners, revealing the large gradients there, while avoiding the backwater regions near outward-directed corners and vertices at infinity. Like the plots of Figure 5.1, these may be viewed as showing simultaneously the image polygon and the domain disk.

Figure 5.5 shows similar plots in which streamlines rather than contour lines have been plotted, so that the configuration may be thought of as portraying ideal irrotational fluid flow through a two-dimensional channel. To plot these streamlines an analytic transformation of the disk to an infinite channel with straight parallel sides was used in conjunction with the Schwarz-Christoffel transformation from the disk to the problem domain.

3. Laplace's equation

Conformal maps do not solve problems, but they may reduce hard problems to easier ones. How much work must be done to solve the easier problem will vary considerably with the application.

(1) In these circumstances, the original problem may be reduced to a model problem whose solution is known exactly. This is the case in the fluid flow problems of Figure 5.5, in which a crooked channel may be mapped to an infinite straight channel of constant width.

(2) If a problem of Laplace's equation with pure Dirichlet or Neumann boundary conditions can be mapped conformally to a disk, then Poisson's formula or Dini's formula [Kantorovich & Krylov, 1958] provide integral representations of the solution at each interior point. Such integrals may be evaluated readily on the computer to yield high accuracy solutions. The primary disadvantage of this approach is that a new integral must be evaluated for each point at which the solution is desired.

(3) If the solution will be required at many points in the domain, then it is probably more efficient to solve Laplace's equation by a trigonometric expansion of the form $a_0 + \sum_{k=1}^{m} r^k(a_k \sin k\theta +$

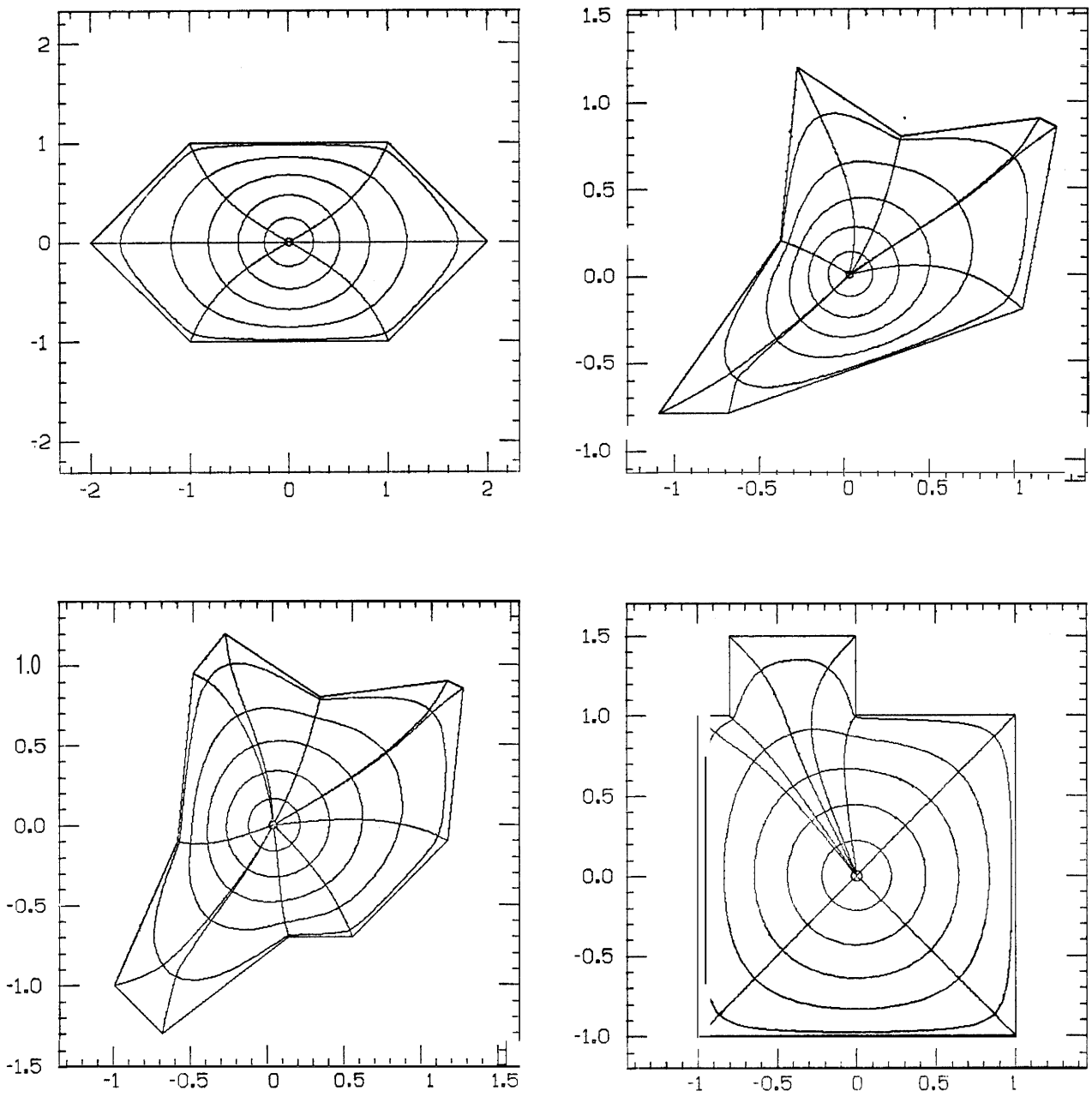


Figure 5.3 – Sample Schwarz-Christoffel transformations (bounded polygons). Contours within the polygons are images of concentric circles at radii .03, .2, .4, .6, .8, .97 in the unit disk, and of radii from the center of the disk to the prevertices z_k.

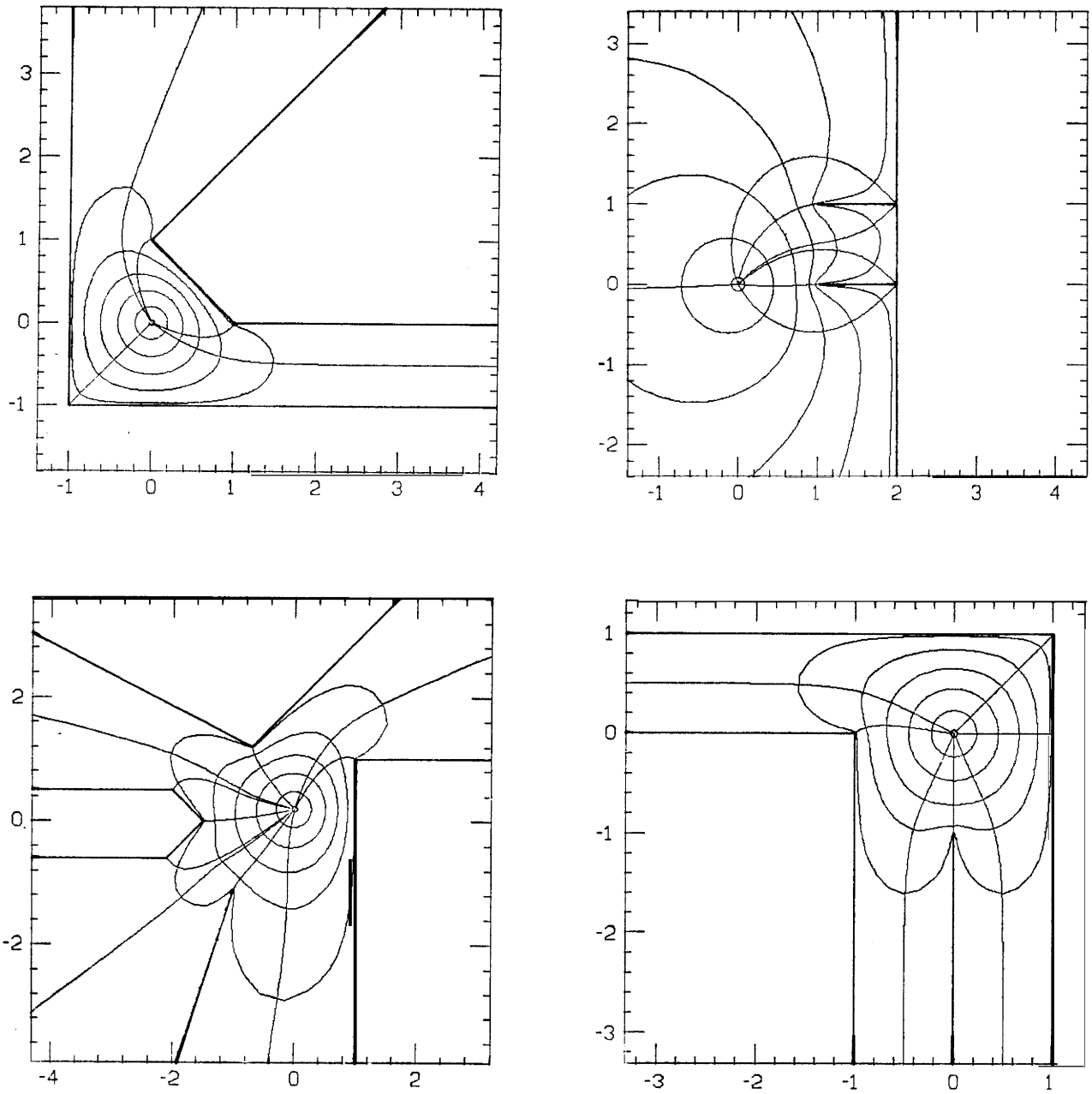


Figure 5.4 – Sample Schwarz-Christoffel transformations (unbounded polygons). Contours are as in Figure 5.3.

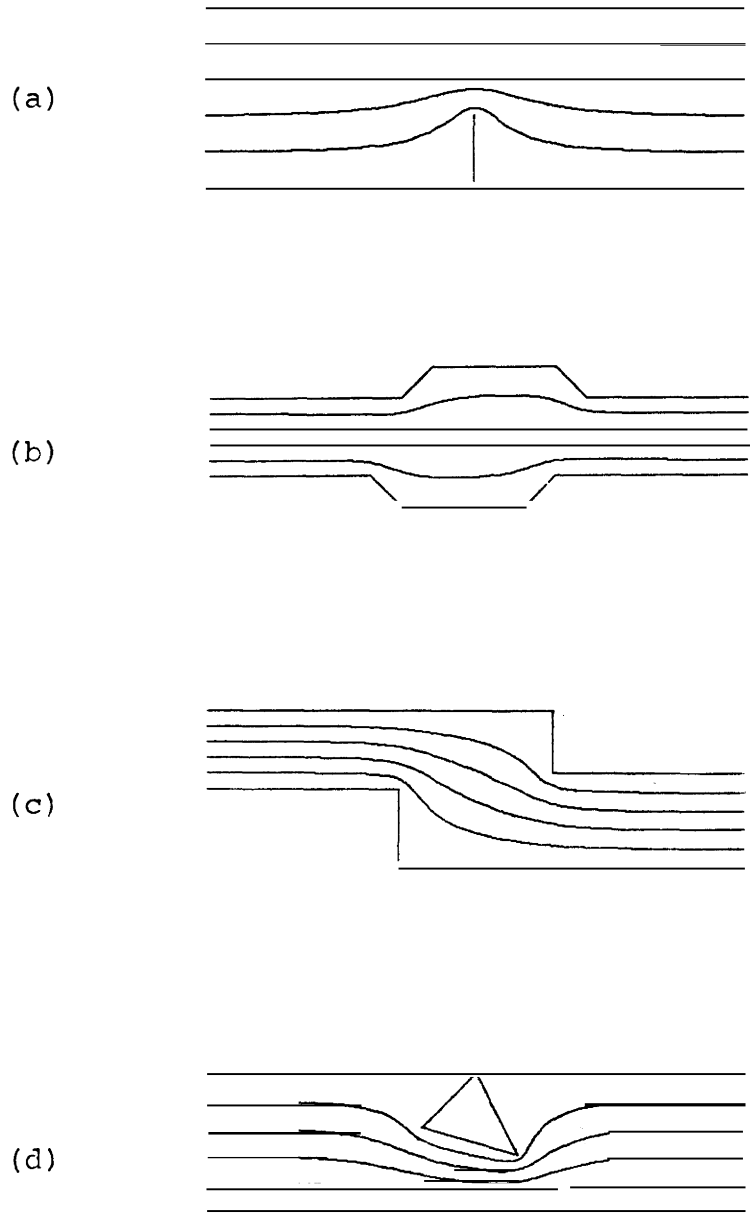


Figure 5.5 – Sample Schwarz-Christoff el transformations. Contours show streamlines for ideal irrotational, nonviscous Auid flow within each **channel.**

b_k co6 $k\theta$); coefficient6 a_k and b_k are selected 60 as to fit the boundary conditions closely. A disadvantage of this method is that convergence of the expansion may be slow if the boundary conditions are not smooth,

(4) Finally, if simpler methods fail, a solution in the model domain may be found by a finite-difference or finite-element technique. For problems of Poisson's equation or more complicated equations this will probably normally be necessary,

Figure 5.6 presents an example of type (1). We are given an infinite region bounded by one straight boundary fixed at potential $\varphi = 0$ and one jagged boundary fixed at $\varphi = 2$. We may think of this as an electrostatics problem. The central question to be answered computationally will be: what are the voltage φ and the electric field $E = -\nabla\varphi$ at a given point, either within the field or on the boundary?

We proceed by mapping the given region onto the disk by a Schwarz-Christoffel transformation, then analytically onto an infinite straight channel (as in the examples of Figure 5.5). In the straight channel φ and E are known trivially, and this information may be transferred to the problem domain through a knowledge of the conformal map that connects them and of its (complex) derivative. We omit the details, which are straightforward.

Figure 5.6b shows $|E|$ as a function of x on the upper and lower boundaries of the region. To see more of the behavior of the solution field near a reentrant corner, we also compute the field at three points near $3 + 1.5i$. These results are given in Figure 5.6c.

4. Poisson's equation

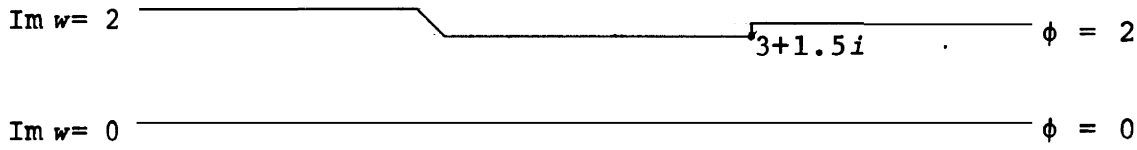
Consider the 'I-sided region shown in Figure 5.7a. We wish to solve Poisson'6 equation

$$\Delta\phi(x, y) = \frac{1}{5} \sin 2x(1 - 2(y + 1)^{\text{“}})$$

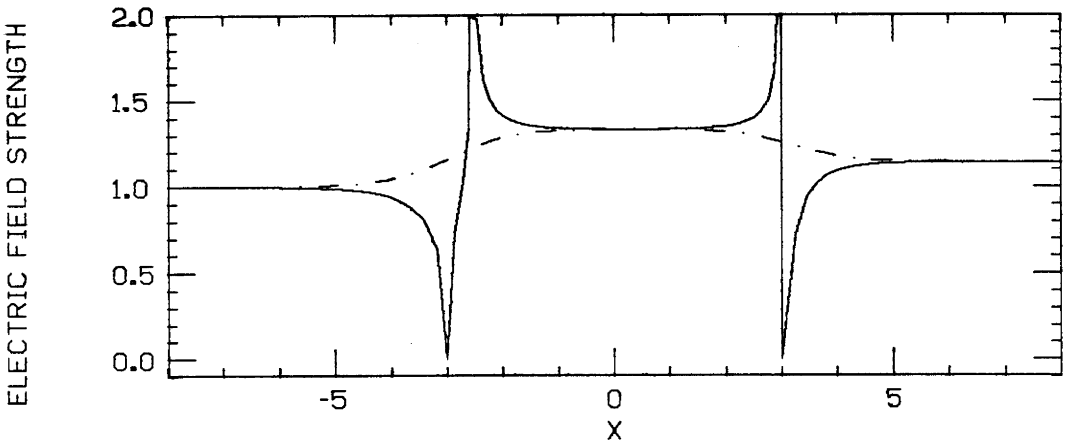
on this region subject to Dirichlet conditions

$$\phi(x, y) = \rho(x, y) = \frac{1}{10} \sin 2x(y + 1)^2$$

on the boundary. We proceed by mapping the domain to the disk and solving a transformed problem in the disk in polar coordinates by mean6 of



(a) Problem domain: region between two conducting sheets



(b) Field strength along the top boundary (solid line) and bottom boundary (broken line)

w	ϕ	$\lvert E \rvert$	$\arg E/\pi$
3.1 +1.4 i	1.7564	1.3082	-.3823
3.01 +1.49 i	1.9486	2.4403	-.2833
3.001+1.499i	1.9889	5.2137	-.2572
3.000+1.500i	2.0000	∞	-.2500

(c) Computed potential and field strength at three points near 3 + 1.5i

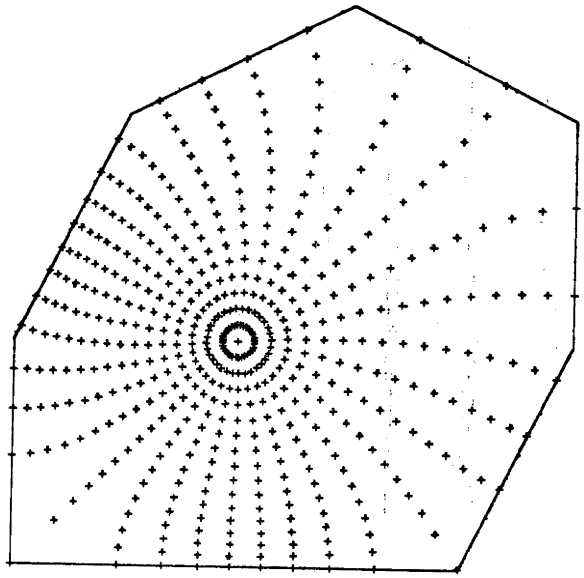
Figure 5.6 — Laplace equation example: electric potential and field between two infinite sheets.

a second-order fast finite difference solver (PWSPLR, by P. Swarztrauber and R. Sweet), $\rho(x, y)$ is the correct solution in the interior as well as on the boundary, so we can determine the accuracy of the numerical solution.

This is not as satisfactory a procedure as was available for Laplace equation problems, According to (1.2), the model problem here is Poisson's equation in the disk with an altered right hand side containing the factor $|f'(z)|^2$, where f is the composite map from the disk to the 7-gon. Two difficluties arise. The first is that to set up the transformed equation in the disk, $\rho(w_{ij})$ must be computed for every $w_{ij} = w(z_{ij})$ which is an image of a grid point in the disk, This is time consuming, one hundred times more so in this experiment than the fast solution of Poisson's equation once it is set up. Second, $|f'(z)|^2$ is singular (unbounded, in this example) at each prevertex z_k, and this appears to interfere with the second-order accuracy which we would like to observe. The table in Figure 5.7b attests to both of these problems.

5. Eigenfrequencies of the Laplace operator

Petter Bjørstad (Computer Science Dept., Stanford University) has recently combined the present Schwarz-Christoffel computation with a fast finite-difference scheme to successfully compute eigenvalues and eigenvectors of the Laplacian operator on polygonal regions. These results may be interpreted as giving the normal modes and frequencies of a thin membrane in two dimensions, or of a three-dimensional waveguide with constant cross-section. This work will be reported elsewhere.



(a) 7-sided problem domain, including image of 16×32 finite-difference grid in the unit disk

Grid ($r\times\theta$)	Transformation and setup time	Fast Poisson solver time	Max. error	RMS error
4×8	1.3 secs.	<.01 secs.	0.132	0.0309
8×16	2 secs.	.01 secs.	0.055	0.0085
16×32	5 secs.	.03 secs.	0.031	0.0037
32×64	16 secs.	.15 secs.	0.026	0.0012

(b) Computed results for four different grids. Time estimates are for an IBM 370/168.

Figure 5.7 — Poisson equation example. Problem is transplanted conformally to the unit disk and solved by finite differences.

VI. CONCLUSION

A program has been described which computes accurate Schwarz-Christoffel transformations from the unit disk to the interior of a simply connected polygon in the complex plane, which may be unbounded. Key features of the computation have been:

(1) Choice of the unit disk rather than the upper half plane as the model domain, for better numerical scaling (II.1)

(2) Use of complex contour integrals interior to the model domain rather than along the boundary, making possible the treatment of unbounded polygons (11.1)

(3) Use of compound Gauss-Jacobi quadrature in complex arithmetic to evaluate the Schwarz-Christoffel integral accurately (II.3,III. 1)

(4) Formulation of the parameter problem as a constrained nonlinear system in N — 1 variables (II.1)

(5) Elimination of constraints in the nonlinear system by a simple variable transformation (11.2)

(6) Solution of the system by a packaged nonlinear systems solver; no initial estimate required (IE.4)

(7) Computation of a reliable estimate of the accuracy of further computations, once the parameter problem has been solved (IV.1)

(8) Accurate evaluation of the inverse mapping in two steps by means of a packaged o.d.e. solver and a packaged complex rootfinder (III.2)

Previous efforts at computing Schwarz-Christoffel transformations numerically include [Cherednichenko & Zhelankina, 1975], [Hopkins & Roberts, 1978], [Howe, 1973], [Meyer, 1979], and [Vecheslavov & Kokoulin, 1973]. The present work differs from these in that it deals directly with

complex arithmetic throughout, taking the unit disk rather than the upper half plane as the model domain and evaluating complex contour integrals. This makes possible the computation of transformations involving general unbounded polygons. (Cherednichenko & Zhelankina [1975] also treat unbounded polygons, by a different method.) Two other important differences are the use of compound Gauss-Jacobi quadrature, and the application of a change of variables to eliminate constraints in the nonlinear system ((5), above). We believe that our program computes Schwarz-Christoffel transformations faster, more accurately, and for a wider range of problems than previous attempts.

A variety of directions for further work suggest themselves. Here are some of them:

(1) More attention should be paid to the problem of inverting the Schwarz-Christoffel map. The two-step method described in III.2 is only one of many possibilities.

(2) The program could easily be extended to construct maps onto the exterior of a polygon—that is, the interior of a polygon whose interior includes the point at infinity. This extension would be necessary for applications to airfoil problems.

(3) It should not be too great a step to raise the present program to the level of "software" by packaging it flexibly, portably, and robustly enough that naive users could apply it to physical problems.

(4) The program might be extended to handle the rounding of corners in Schwarz-Christoffel transformations [Henrici,1974]. What about mapping doubly or multiply connected polygonal regions, perhaps by means of an iterative technique which computes an S-C transformation at each step? What about applying S-C transformations to eliminate corners in the conformal mapping of curved domains?

Most important, further work is needed in the direction of applications to Laplace's equation, Poisson's equation, and related problems. Irregular or unbounded domains are generally troublesome to deal with by standard techniques, particularly when singularities in the form of reentrant corners are present. Schwarz-Christoffel transformations offer a means of getting around such difficulties in a natural way. Much more experience is needed here.

APPENDIX: PROGRAM LISTING

The boundaries of this program are not sharply defined, for the configuration changes according to what applications are being treated. The present listing includes only the core routines used to solve the parameter problem and to evaluate the Schwarz-Christoffel function and its inverse,

An experimental copy of the package may be obtained in machine-readable form from the author,

Control program:

- SC

Set-up:

- INIT initializes variables and reads input data
- QINIT computes quadrature nodes and weights

Solution of parameter problem:

- SCSOLV controls solution of parameter problem
  - YZTRAN transforms to an unconstrained system
  - SCFUN sets up the nonlinear system to be solved
  - SCOUTP prints output from SCSOLV
- TEST estimates accuracy of computed solution

Compound Gauss-Jacobi quadrature:

- ZQUAD divides the integral into two halves
  - ZQUAD1 evaluates the half-integral (compound)
  - DIST finds the distance to the nearest singularity
  - ZQSUM sums a Gauss-Jacobi quadrature rule

Forward and inverse S-C map:

- WSC evaluates map from disk to polygon
- ZSC evaluates map from polygon to disk
  - ZFODE computes initial estimate
  - ZNEWT inverts map by Newton's method

Miscellaneous routines:

- ZPROD evaluates N-fold Schwarz-Christoffel integrand
- FINITE returns "true" if the argument is finite
- ENTER begins timing of the current subroutine
- EXIT concludes timing of the current subroutine

Library routines not listed:

GAUSSQ (Golub&Welsch) computes Gauss-Jacobi **nodes** and wts (called by QINIT)

NS01A (Powell) solves the nonlinear system (called by SCSOLV)

ODE (Shampine & Gordon) solves the inverse **mapping problem** (called by ZSC)

```
C*******************************************************************
C* SC                                                    MAIN PROGRAM *
C*******************************************************************

C PROGRAM SC - "SCHWARZ-CHRISTOFFEL"
C  --------------------

C THIS PROGRAM COMPUTES THE SCHWARZ-CHRISTOFFEL TRANSFORMATION
C WHICH SENDS THE UNIT DISK TO THE INTERIOR OF THE POLYGON
C W(1),...,W(N).  THIS MAPPING IS OF THE FORM:
C
C                              Z     N
C     W   =   WC   +   C  *  INT PROD(1-Z/Z(K) )**BETAM (K) DZ .       (1)
C                            0    K=1

C TO SOLVE THE PROBLEM WE BEGIN BY FINDING THE ACCESSORY PARAMETERS
C -- VERTICES Z(K) AND CONSTANT C -- FOR THE MAP OF (1).  THIS IS DONE
C BY SUBROUTINE SCSOLV.

C THE IMAGE POLYGON MAY BE UNBOUNDED; PERMITTED ANGLES LIE IN THE
C RANGE -3.LE.BETAM(K).LE.1.  W(N) AND W(1) MUST BE FINITE.
C WE NORMALIZE BY THE CONDITIONS:
C
C     Z(N) = 1                                                        (2.1)
C     W(0.0) = WC  (A POINT IN THE INTERIOR OF THE POLYGON)           (2.2)

C NOTATION:

C   W(K) - VERTEX K OF THE IMAGE POLYGON
C   Z(K) - POINT ON THE UNIT DISK MAPPED TO W(K)
C BETAM (K) - NEGATIVE OF EXTERIOR ANGLE AT W(K) DIVIDED BY PI
C   N - NUMBER OF VERTICES W(K)
C   NM = N-1 - NUMBER OF UNKNOWN POINTS: Z(1),...,Z(N-1)
C   NPTSQ - NUMBER OF POINTS FOR GAUSS-JACOBI QUADRATURE
C  ZINF  - COMPLEX INFINITY

C LOCAL ROUTINES:

C   SC - MAIN PROGRAM
C  INIT  - INITIALIZES CONSTANTS AND DEFINES PROBLEM
C  QINIT  - COMPUTES QUADRATURE NODES AND WEIGHTS
C  SCSOLV  - COMPUTES ACCESSORY PARAMETERS FOR S-C MAP (1)
C     YZTRAN - TRANSFORMS UNKNOWNS FROM Y(K) TO Z(K)
C     SCFUN  - NONLINEAR SYSTEM OF EQUATIONS TO BE SOLVED BY SCSOLV
C     SCOUTP - PRINTS OUTPUT FROM SCSOLV
C  WSC  - COMPUTES W(Z)
C  zsc  - COMPUTES Z(W)
C   PLTCON - DRAWS PLOTS OF IMAGE POLYGON WITH CONTOURS
C   ZPROD - COMPUTES N-FOLD PRODUCT IN (1)
C  ZQUAD  - SUMS TO EVALUATE INTEGRAL BY GAUSS-JACOBI QUADRATURE
C   FINITE - RETURNS TRUE IF ARGUMENT IS FINITE

C LIBRARY ROUTINES REQUIRED:  MS01A,  GAUSSQ, ODE.

C L.N. TREFETHEN   JANUARY, 1978

      IMPLICIT REAL*8(A-B,D-H,O-V,X-Y), COMPLEX*16(C,W,Z)
      COMMON /SC/ WC,W(20),BETAM(20),C,Z(20),N,NM,NP
       COMMON /CONSTS/ PI,TWOPI,ZERO,ZINF,EPS
      REAL*8 CDABS

C SET UP PROBLEM:
      EPS = 1.D-8
      CALL INIT

C COMPUTE NODES AND WEIGHTS FOR PARAMETER PROBLEM:
      NPTSQ = 8
      CALL QINIT(NPTSQ)

C SOLVE PARAMETER PROBLEM:
      IPRINT = 1
      CALL SCSOLV(NM,IPRINT)
C
C TEST ACCURACY OF SOLUTION:
      CALL TEST
C
C DRAW CONTOUR PLOT OF SOLUTION:
      CALL PLTCON

  103 CONTINUE
      STOP 1
      END
//GO.SYSIN DD *
  7                                        N
      .0       .0                          UC
      2.       0.      99.
      2.       8.      -.5
   1.370    1.D70      -1.
    -.2      -2.      -.5
    -.2      -1.      99.
     .7    -2.5       99.
     .8    -2.7       99.
```

```
C*********************************************************************
C* INIT                                          PRIMARY SUBROUTINE * *
C*********************************************************************
C
      SUBROUTINE INIT
C
C INITIALIZES CONSTANTS IN /CONSTS/ AND PROBLEM DEFINITION
C PARAMETERS IN /SC/.  DATA FOR THE GECMETRY OF THE PROBLEM
C IS READ IN FROM UNIT 5.
C
      IMPLICIT REAL*8(A-B,D-H,O-V,X-Y), COMPLEX*16(C,W,Z)
      LOGICAL FINITE
      COMPLEX*16 DCMPLX
      COMMON /SC/ WC,W(20),BETAM(20),C,Z(20),N,NM,NP
      COMMON/CONSTS/  PI,TWOPI,ZERO,ZINF,EPS
      COMMON /GEOM/ KFIX(20),KRAT(20),NCOMP
      DATA SBNAME /'INIT'/
      CALL ENTER(SBNAME)
C
C SET CONSTANTS:
      PI = 3.14159 26535 89793 23 D0
      TWOPI = PI * 2.D0
      ZERO = (0.D0,0.D0)
      ZINF = (1.D70,1.D70)
C
C READ INPUT PARAMETERS:
      READ (5,201) N
      NM  =N-1
      NP = N+1
      Z(N)  = (1.D0,0.D0)
      READ (5,202) WC
      READ (5,203)  (W(K),BETAM(K),K=1,N)

C COMPUTE ANGLES AS REQUIRED  (WHERE VALUE INPUT IS 99.0):
      DO 10 K =   1,N
      IF (BETAM(K).NE.99.D0) GOTO 10
      KM = MOD(K+N-2,N) + 1
      KP = MOD(K,N)+1
      BETAM(K) = DIMAG(CDLOG((W(KM)-W(K))/(W(KP)-W(K))))/PI - 1.D0
      IF (BETAM(K).LE.-1.D0) BETAM(K) = BETAM(K) * 2.D0
   10 CONTINUE
C
C CHECK FOR VARIOUS INPUT  ERRORS:
      SUM = 0.D0
      DO 1 K = 1,N
    1 SUM = SUM * BETAM(K)
      IF (DABS(SUM+2.D0).LT.EPS) GOTO 2
      WRITE (6,30 1)
      STOP 2
    2 IF (FINITE(W( 1))) GOTO 3
      WRITE (6,302)
      STOP 2
    3 IF (FINITE(W(N))) GOTO 4
      WRITE (6,303)
      STOP 2
    4 IF (BETAM(NM).NE.0.D0) GOTO 5
      WRITE (6,304)
    5 IF (BETAM(NM).NE.1.D0) GOTO 20
      YRITE (6,305)
      STOP 2
C
C DETERMINE NUMBER OF BOUNDARY COMPONENTS, ETC.:
C   PASS 1:  ONE FIXED POINT FOR EACH INFINITE VERTEX:
   20 NCOMP = 0
      DO 21 K = 2,NM
        IF (FINITE(W(K))) GOTO 21
        NCOMP = NCOMP * 1
        KFIX(NCOMP) = K - 1
        IF (NCOMP.EQ. 1) KFIX(NCOMP) = 1
   21 CONTINUE
      IF (NCOMP.GT.0) GOTO 22
      NCOMP = 1
      KFIX(NCOMP) = 1
C   PASS 2: ONE RATIO FOR EACH LINE SEGMENT:
   22 CONTINUE
      NEQ= 2*NCOMP
      DO 23 K = 1,NM
        IF (NEQ.EQ.NM) GOTO 30
        IF (.NOT.FINITE(W(K)).OR..NOT.FINITE(W(K+1))) GOTO 23
        NEQ = NEQ + 1
        KRAT(NEQ) = K
   23 CONTINUE
C
   30 CALL EXIT
      RETURN
C
  201 FORMAT (I3)
  202 FORMAT (2F8.0)
  203 FORMAT (2D8.0,F8.0)
  301 FORMAT (/' *** ERROR IN INIT: ANGLES DO NOT ADD UP TO 2'/)
  302 FORMAT (/' *** ERROR IN INIT:  W(1) MUST BE FINITE'/)
  303 FORMAT (/' *** ERROR IN INIT:  W(N) MUST BE FINITE'/)
  304 FORMAT (/' *** WARNING IN INIT: W(N-1) NOT DETERMINED'/)
  305 FORMAT (/' * ** ERROR IN INIT:  W(N-1) NOT DETERMINED'/)
      END
```

```
C*********************************************************************
C* QINIT                                          PRIMARY SUBROUTINE * *
C*********************************************************************
      SUBROUTINE QINIT(NPTS)
C
C COMPUTES NODES AND WEIGHTS FOR GAUSS-JACOBI QUADRATURE
C
      IMPLICIT REAL*8(A-B,D-H,O-V,X-Y), COMPLEX*16(C,W,Z)
      LOGICAL FINITE
      COMMON /SC/ WC,W(20),BETAM(20),C,Z(20),N,NM,NP
      COMMON /QUAD/ QNODES(32,21),QWTS(32,21),NPTSQ
      DIMENSION QESCR(2), QSCR(32)
      DATA SBNAME /'QINIT'/
      CALL ENTER(SBNAME)
      WRITE (6,201) NPTS
C
      NPTSQ = NPTS
C
C  FOR EACH  FINITE  VERTEXW(K), COMPUTE NODES AND WEIGHTS FOR
C ONE-SIDED GAUSS-JACOBI QUADRATURE ALONG A CURVE BEGINNING AT Z(K):
      DO 1 K = 1,N
    1 IF (FINITE(W(K))) CALL GAUSSQ(5,NPTSQ,0.D0,BETAM(K),0,
     & QESCR,QSCR,QNODES(1,K),QWTS( 1,K))
C
C COMPUTE NODES AND WEIGHTS FOR PURE GAUSSIAN QUADRATURE:
      CALL GAUSSQ(5,NPTSQ,0.D0,0.D0,0,QESCR,QSCR,QNODES(1,NP),
     &QWTS  (1,NP))
C
      CALL EXIT
      RETURN

  231 FORMAT (' NPTS =',I5)
      END
```

```
C*********************************************************************
C* TEST                                           PRIMARY SUBROUTINE * *
C*********************************************************************
C
      SUBROUTINE TEST
C
C TESTS THE COMPUTED RAP FOR ACCURACY.
C
      IMPLICIT REAL*8(A-B,D-H,O-V,X-Y), COMPLEX*16(C,W,Z)
      REAL*8 CDABS
      LOGICAL FINITE
      COMMON /SC/WC,W (20) ,BETAM(20),C,Z(20),N,NM,NP
      COMMON/CONSIS/  PI,TWOPI,ZERO,ZINF,EPS
      DATA SBNAME /'TEST'/
      CALL ENTER(SBNAME)
C
C TEST LENGTH OF RADII:
      RADEMX = 0.D0
      DO  10 K = 2,N
      IF (FINITE(W(K))) RADE = CDABS(WC - WSC(ZERO,Z(K),W(K),K))
      IF (.NOT.FINITE(W(K))) RADB =
     & CDABS(WSC((. 1D0,. 1D0),Z (K-1), W (K-1) ,K-1)
     &  - WSC((. 1D0,. 1D0),Z(K+1) ,W(K+1),K+1))
      RADEMX = DMAX1(RADEMX,RADE)
   10 CONTINUE
      WRITE (6,20 1) RADEMX
C
      CALL EXIT
      RETURN
C
  201  FORMAT (/' RADEMX:',D12.4)
       END
```

```
C*********************************************************************
C* SCSOLV                                          PRIMARY SUBROUTINE **
C*********************************************************************
C
      SUBROUTINE SCSOLV(NM,IPRINT)
C
C THIS SUBROUTINE COMPUTES THE ACCESSORY PARAMETERS C AND Z(K).
C THE PROBLEM IS SOLVED BY PINDIEG THE
C SOLUTICN TO A SISTER OP N-1 YONLINEAB EQUATIONS IN THE N-1
C UNKNOWNS Y (1),...,Y(N-1), WHICH ABE RELATED TO TAE POINTS
C Z(K) BY THE FORMULA:
C
C       Y(K) = LOG ((TH(K)-TH(K-1))/(TH(K+1)-TH(K)))                (1)
C
C WHERE TH(K) DENOTES THE ARGUMENT OP Z(K).
C SUBROUTINE SCPUN DEPINES THIS SYSTEM OP EQUATIOBS.
C THE ORIGINAL PROBLEM IS SUBJECT TO TEE CONTRAINTS TH(K) < TH(K+1),
C BUT IHESE VANISH IN THE TRANSFORMATICNFROMZ TO Y.
C
C SEE MAIN PROGRAM POR FURTHER COMMENTS.
C
      IMPLICIT REAL*8(A-B,D-H,O-V,X-Y), COMPLEX*16(C,W,Z)
      COMMON /CONSTS/ PI,TWOPI,ZERO,ZINF,EPS
      DIMENSION AJINV(20,20), SCR(900), FVAL(19), Y(19)
      EXTERNAL SCFUN
      DATA SBNAME /'SCSOLV'/
      CALL ENTER(SBNAME)
C
C INITIAL GUESS (VERTICES EQUALLY SPACED AROUND CIRCLE):
      DO 3 K = 1,NM
    3 Y(K) = 0.D0
C
C NS01A COYTROL PARAMETERS:
      DSTEP = 1.D-8
      DMAX = 1.D1
      ACC = EPS
      MAXFUN = NM * 15
C
C SOLVE NONLINEAR SISTER WITH NS01A:
      CALL NS01A(NM,Y,FVAL,AJINV,DSTEP,DMAX,ACC,MAXFUN,IPRINT,SCR,SCFUN)
      CALL YZTRAN(Y)
C
C PRINT RESULTS:
      CALL SCOUTP

      CALL EXIT
      RETURN
C
      END
C*********************************************************************
C* YZTRAN                                SUBORDINATE(SCSOLV) SUBROUTINE * *
C*********************************************************************
C
      SUBROUTINE YZTRAN(Y)
C
C TRANSFORMS Y(K) TO Z(K).  SEE COMMENTS IN SUBROUTINE SCSOLV.
C
      IMPLICIT REAL*8(A-B,D-H,O-V,X-Y), COMPLEX*16(C,W,Z)
      COMPLEX*16 DCMPLX
      COMMON /SC/ WC,W(20),BETAM(20),C,Z(20),N,NM,NP
      COMMON /CONSTS/ PI,TWOPI,ZERO,ZINF,EPS
      DIMENSION Y(1)
C
      DTH = 1.D0
      THSUM = DTH
      DO 1 K = 1,NM
        DTA = DTH / DEXP(Y(K))
    1   THSUB = THSUB + DTH
C
      DTA = TWOPI / THSUM
      THSUM = DTH
      Z(1) = DCMPLX(DCOS(DTH),DSIN(DTH))
      DO 2 K = 2,NM
        DTR = DTH / DEXP(Y (K-1))
        THSUB = THSUM t DTH
    2   Z(K) = DCMPLX(DCOS(THSUM),DSIN(THSUM))
C
      RETURN
      END
```

```
C*************************************                ********************
C* SCFUN                                SUBORDINATE(SCSOLV) SUBROUTINE  **
C*************************************                ********************
      SUBROUTINE SCFUN(NDIM,Y,FVAL)
C
C THIS IS THE FUNCTION WHOSE ZERO MUST BE FOUND IN SCSOLV.
C
      IMPLICIT REAL*8(A-B,D-H,O-V,X-Y), COMPLEX*16(C,W,Z)
      REAL*8 CDABS
      LOGICAL FINITE
      DIMENSION FVAL(NDIM),Y(NDIM)
      COMMON /SC/ WC,W(20),BETAM(20),C,Z(20),N,NM,NP
      COMMON /CONSTS/ PI,TWOPI,ZERO,ZINF,EPS
      COMMON /GEOM/ KFIX(20),KRAT(20),NCOMP
C
C TRANSFORM Y(K) TO Z(K):
      CALL YZTRAN(Y)

C SET UP: COMPUTE INTEGRAL FROM 0 TO Z(N):
      WDENOM = ZQUAD(ZERO,0,Z(N),N)
      C = (W(N)-WC) / WDENOM
C
C CASE 1: W(K) AND W(K+1) FINITE:
C (COMPUTE INTEGRAL ALONG CHORD Z(K)-Z(K+1)):
      NFIRST = 2*NCOMP + 1
      IF (NFIRST.GT.NM) GOTO 11
      DO 10 NEQ = NFIRST,NM
        KL = KRAT(NEQ)
        KR = KL+1
        ZINT = ZQUAD(Z(KL),KL,Z(KR),KR)
        FVAL(NEQ) = CDABS(W(KR)-W(KL)) - CDABS(C*ZINT)
   13 CONTINUE
C
C CASE 2: W(K+1) INFINITE:
C (COMPUTE CONTOUR INTEGRAL ALONG RADIUS 0-Z(K)):
   11 DO 23 NVERT = 1,NCOMP
        KR = KFIX(NVERT)
        ZINT = ZQUAD(ZERO,0,Z(KR),KR)
        ZPVAL = W(KR) - WC - C*ZINT
        FVAL(2*NVERT-1) = DREAL(ZFVAL)
        FVAL(2*NVERT) = DIMAG(ZFVAL)
   20 CONTINUE
      RETURN
C
      END
C*************************************************************************
C* SCOUTP                              SUBORDINATE(SCSOLV) SUBROUTINE  **
C*************************************************************************
C
      SUBROUTINE SCOUTP
C
C PRINTS RESULTS (VARIABLES IN COMMON BLOCK /SC/)
C
      IMPLICIT REAL*8(A-B,D-H,O-V,X-Y), COMPLEX*16(C,W,Z)
      LOGICAL FINITE
      COMMON /SC/ WC,W(20),BETAM(20),C,Z(20),N,NM,NP
      COMMON /CONSTS/ PI,TWOPI,ZERO,ZINF,EPS
C
      WRITE (6,102)
      DO 1 K = 1,N
      THDPI = DIMAG(CDLOG(Z(K))) / PI
      IF (THDPI.LE.0.D0) THDPI = THDPI + 2.D0
      IF (FINITE(W(K))) WRITE (6,103) K,W(K),THDPI,BETAM(K),Z(K)
    1 IF (.NOT.FINITE(W(K))) WRITE (6,104) K,THDPI,BETAM(K),Z(K)
      WRITE (6,105) WC,C
      RETURN
C
  102 FORMAT (//' RESULTS:'//
     &  '  I',10X,'W(K)',13X,'TH(K)/PI',11X,'BETAM(K)',
     &  18X,'Z(K)'/
     &  ' ---',9X,'----',13X,'---------',11X,'--------',
     &  18X,'----'//)
  103 FORMAT (I3,'      (',F6.3,',',F6.3,')',F20.14,F14.5,
     &  3X,'(',F15.12,',',F15.12,')')
  104 FORMAT (x3,'           INFINITY    ',F20.14,F14.5,
     &  3X,'(',F15.12,',',F15.12,')')
  105 FORMAT (//' WC = (',D22.15,',',D22.15,')'/
     &        '  C = (',D22.15,',',D22.15,')'/)
      END
```

```
C*********************************************************************
C* ZQUAD                                          SECONDARY SUBROUTINE * *
C*********************************************************************

      FUNCTION ZQUAD(ZA,KA,ZB,KB)

C COMPUTES THE COMPLEX LINE INTEGRAL OF ZPROD FROM ZA TO ZB ALONG A
C STRAIGHT LINE SEGMENT WITHIN THE UNIT DISK.  FUNCTION  ZQUAD1  IS  .
C CALLED TWICE, ONCE FOR EACH HALF OF THIS INTEGRAL.
C
      IMPLICIT REAL*8(A-B,D-H,C-V,X-Y), COMPLEX*16(C,W,Z)
C
      ZMID = (ZA + ZB) / 2.D0
      ZQUAD = ZQUAD1(ZA,ZMID,KA) - ZQUAD1(ZB,ZMID,KB)
      RETURN
      END
C*********************************************************************
C* ZQUAD1                                  SUBORDINATE(ZQUAD) SUBROUTINE **
C*********************************************************************
C
      FUNCTION ZQUAD1(ZA,ZB,KA)
C
C COMPUTES THE COMPLEX LINE INTEGRAL OF ZPROD FROM ZA TO ZB AL016 A
C STRAIGHT LINE SEGMENT WITHIN THE UNIT DISK.  COMPOUND ONE-SIDED
C GAUSS-JACOBI QUADRATURE IS USED, USING FUNCTION DIST TO DETERMINE
C THE DISTANCE TO THE NEAREST SINGULARITY Z(K).

      IMPLICIT REAL*8(A-B,D-H,O-V,X-Y), COMPLEX*16(C,W,Z)
      COMMON /CONSTS/ PI,TWOPI,ZERO,ZINF,EPS
      REAL*8 CDABS
      DATA RESPRM /2.D0/
C
C CHECK FOR ZERO-LENGTH INTEGRAND:
      IF (CDABS (ZA-ZB).GT.0.D0) GOTO 1
      ZQUAD1 = ZERO
      RETURN
C
C STEP 1: ONE-SIDED GAUSS-JACOBI QUADRATURE FOR LEFT ENDPOINT:
    1 R = DMIN1(1.D0,DIST(ZA,KA) *RESPRM/CDABS(ZA-ZB))
      ZAA = ZA + R*(ZB-ZA)
      ZQUAD1 = ZQSUM(ZA,ZAA,KA)
C
C STEP 2: ADJOIN INTERVALS OF PURE GAUSSIAN QUADRATURE IF NECESSARY:
   10 IF (R.EQ. 1.D0) RETURN
      R = DMIN1(1.D0,DIST(ZAA,0)*RESPRM/CDABS(ZAA-ZB))
      ZBB = ZAA + R*(ZB-ZAA)
      ZQUAD1 = ZQUAD1 + ZQSUM(ZAA,ZBB,0)
      ZAA = ZBB
      GOTO 10
      END
C*********************************************************************
C* DIST                                    SUBORDINATE (ZQUAD) SUBROUTINE **
C*********************************************************************
C
      FUNCTION DIST(ZZ,KS)
C
C DETERMINES THE DISTANCE FROM ZZ TO THE NEAREST SINGULARITY Z(K)
C OTHER THAN Z(KS).
C
      IMPLICIT REAL*8(A-B,D-H,O-V,X-Y), COMPLEX*16(C,W,Z)
      COMMON /SC/ WC,W(20),BETAM(20),C,Z(20),N,NM,NP
      REAL*8 CDABS
C
      DIST = 99.D0
      DO 1 K = 1,N
      IF (K.EQ.KS) GOTO 1
      DIST = DMIN1(DIST,CDABS(ZZ-Z(K)))
    1 CONTINUE
      RETURN
      END
C*********************************************************************
C* ZQSUM                                   SUBORDINATE(ZQUAD) SUBROUTINE **
C*********************************************************************
C
      FUNCTION ZQSUM(ZA,ZB,KA)
C
C COMPUTES THE INTEGRAL OF ZPROD FROM ZA TO ZB BY APPLYING A
C ONE-SIDED GUASS-JACOBI FORMULA WITH POSSIBLE SINGULARITY AT ZA.
C
      IMPLICIT REAL*8(A-B,D-H,O-V,X-Y), COMPLEX*16(C,W,Z)
      COMMON /SC/ WC,W(20),BETAM(20),C,Z(20),N,NM,NP
      COMMON /CONSTS/ PI,TWOPI,ZERO,ZINF,EPS
      COMMON /QUAD/ QNODES(672),QWTS(672),NPTSQ
      REAL*8 CDABS
C
      ZS = ZERO
      ZH = (ZB-ZA) / 2.D0
      ZC = (ZA+ZB) / 2.D0
      K = KA
      IF (K.EQ.0) K = NP
      I1 = 32*(K-1) + 1
      I2 = I1 + NPTSQ - 1
      DO 1 I = I1,I2
    1 ZS = ZS + QWTS(I) *ZPROD(ZC+ZH*QNODES(I),KA)
      ZQSUM = ZS*ZH
      IF (CDABS(ZH) .NE.0.D0.AND.K.NE.NP)
     & ZQSUM = ZQSUM*CDABS(ZH)**BETAM(K)
      RETURN
      END
```

```
C*********************************************************************
C* WSC                                          PRIMARY SUBROUTINE  * *
C*********************************************************************

      FUNCTION WSC(ZZ,Z0,W0,KZ0)
C
C INTEGRATES FROM Z0 TO ZZ TO COMPUTE W VALUE CORRESPONDING TO ZZ
C
      IMPLICIT REAL*8(A-B,D-H,O-V,X-Y), COMPLEX*16(C,W,Z)
      COMMON /SC/ WC,W(20),BETAM(20),C,Z(20),N,NM,NP
C
      WSC = W0 + C * ZQUAD(Z0,KZ0,ZZ,0)
C
      RETURN
      END




C*********************************************************************
C* ZSC                                          PRIMARY SUBROUTINE **
C*********************************************************************
C
      FUNCTION ZSC(WW,Z0,W0,KZ0)
C
C COMPUTES Z(WW).  FIRST ODE IS CALLED TO GET AN INITIAL ESTIMATE;
C THEN ZNEWT IS CALLED TO GET THE FINAL ANSWER.
C
      IMPLICIT REAL*8(A-B,D-H,O-V,X-Y), COMPLEX*16(C,W,Z)
      DIMENSION SCR(142), ISCR(5)
      EXTERNAL ZFODE
      COMMON /SC/ WC,W(20),BETAM(20),C,Z(20),N,NM,NP
      COMMON /CONSTS/ PI,TWOPI,ZERO,ZINF,EPS
      COMMON /ZSCCOM/ CDWDT
C
C GET INITIAL GUESS Z1 VIA ODE:
      Z1 = ZERO
      T = 0.D0
      IFLAG = -1
      RELERR = 0.D0
      ABSERR = 5.D-3
      CDWDT  =(WW-WC)/C
      CALL ODE(ZFODE,2,Z1,T,1.D0,RELERR,ABSERR,IFLAG,SCR,ISCR)
      IF (IFLAG.NE.2) WRITE (6,201) IFLAG
C
C REFINE ANSWER VIA ZNEWT:
      CALL ZNEWT (Z1,WW,EPS,KZ0)
      ZSC = Z1
C
  23  FORMAT ('/ *** NONSTANDARD RETURN FROM ODE IN ZSC: IFLAG =',I2/)
      RETURN
      END
C*********************************************************************
C* ZFODE                                   SUBORDINATE(ZSC) SUBROUTINE * *
C*********************************************************************
C
      SUBROUTINE ZFODE(T,ZZ,ZDZDT)

C COMPUTES THE FUNCTION ZDZDT NEEDED BY ODE IN ZSC.

      IMPLICIT REAL*8(A-B,D-H,O-V,X-Y), COMPLEX*16(C,W,Z)
      COMMON /ZSCCOM/ CDWDT
C
      ZDZDT = CDWDT / ZPROD(ZZ,0)

      RETURN
      END
C*********************************************************************
C* ZNEWT                                   SUBORDINATE(ZSC) SUBROUTINE  **
C*********************************************************************
C
      SUBROUTINE ZNEWT(ZROOT,WW,EPS,KZ0)
C
C IMPLEMENTS NEWTON'S METHOD TO SOLVE FOR EQUATION
C W(ZROOT) = WW FOR ZROOT.

      IMPLICIT REAL*8(A-B,D-H,O-V,X-Y), COMPLEX*16(C,W,Z)
      COMMON /SC/WC,W(20),BETAM(20),C,Z(20),N,NM,NP
C
      DO 1 ITER = 1,10
      ZROOT0 = ZROOT
        IF (KZ0.EQ.0) ZFNWT =WW - WSC(ZROOT0,(0.D0,0.D0),WC,0)
        IF (KZ0.NE.0) ZFNWT = WW - WSC(ZROOT0,Z(KZ0),W(KZ0),KZ0)
      ZROOT = ZROOT0 + ZFNWT/(C*ZPROD(ZROOT0,0))
      IF (CDABS(ZFNWT).LT.EPS) RETURN
    1 CONTINUE
      WRITE (6,201)
      RETURN
C
  201 FORMAT (/' * ** ERROR IN ZNEWT: UC CONVERGENCE II 10 ITERATIONS')
      END
```

```
C*********************************************************************
C* ZPROD                                          SECONDARY SUBROUTINE **
C*********************************************************************
C
      FUNCTION ZPROD(ZZ,KS)
C
C COMPUTES THE INTEGRAND
C
C            N
C          PROD   (1-ZZ/Z(K))**BETAM(K)   ,
C           K=1
C
C TAKING ARGUMENT ONLY (NOT MODULUS) FOR TERM K = KS.
C
      IMPLICIT REAL*8(A-B,D-H,O-V,X-Y), COMPLEX*16(C,W,Z)
      REAL*8 CDABS
      COMMON /SC/ WC,W(20),BETAM(20),C,Z(20),N,NM,NP
C
      ZSUM = (0.D0,0.D0)
      DO 1 K = 1,N
        ZTMP = (1.D0,0.D0) - ZZ/Z(K)
        IF (K.EQ.KS) ZTMP = ZTMP / CDABS(ZTMP)
    1   zsum = ZSUM + BETAM(K)*CDLOG(ZTMP)
      ZPROD = CDEXP(ZSUM)
      RETURN
      END
```

```
C*********************************************************************
C* FINITE                                         SECONDARY SUBROUTINE * *
C*********************************************************************
C
      FUNCTION FINITE(Z)
C
C RETURNS TRUE IF AND ONLY IF Z IS NOT INFINITE
C
      IMPLICIT REAL*8 (A-B,D-H,O-V,X-Y), COMPLEX*16(C,W,Z)
      LOGICAL FINITE
      COMMON /CONSTS/ PI,TWOPI,ZERO,ZINF,EPS
C
      FINITE = DREAL(Z).NE.DREAL(ZINF)
      RETURN
      END
```

```
C*********************************************************************
C* ENTER                                          SECONDARY SUBROUTINE * *
C*********************************************************************

      SUBROUTINE ENTER(SBNAME)
C
C STARTS TIMING TIME SPENT IN SUBROUTINE WITH NAME SBNAME.
C
      IMPLICIT REAL*8(A-B,D-H,O-V,X-Y),  COMPLEX*16(C,W,Z)
      COMMON /TIME/ TENTER
C
      CALL LEFT1A(TENTER)
      WRITE (6,201) SBNAME
      RETURN
C
  201 FORMAT (//1X,80('X'),' ENTERING ',A8)
      END
C*********************************************************************
C* EXIT                                           SECONDARY SUBROUTINE **
C*********************************************************************
C
      SUBROUTINE EXIT
C
C PRINTS TIME SPENT IN SUBROUTINE.
C
      IMPLICIT REAL*8(A-B,D-H,O-V,X-Y), COMPLEX*16(C,W,Z)
      COMMON /TIME/ TENTER
C
      CALL LEFT1A(TEXIT)
      TIME = TENTER - TEXIT
      WRITE (6,201) TIME
      RETURN
C
  201 FORMAT (1X,80('X'),' TIME ELAPSED:',F7.3,' SECS.'/)
      END
```

REFERENCES

Cherednichenko, LA. and Zhelankina, I.K., "'Determination of the constants that occur in the Christoffel-Schwarz integral," Izv. Vyssh. Ucebn. Zaved. Elektromehanika (19'75) 10, 1037-1040 (Russian). Untranslated; see Math. Reviews 53, #13535.

Davis, P. J., and Rabinowitz, P., Methods of Numerical Integration, Academic Press, 1975.

Gaier, D., Konstruktive Methoden der konformen Abbildung, Springer-Verlag, 1964,

Golub, G. II., and Welsch, J. II., "Calculation of Gaussian Quadrature Rules," Math. Comp. 23 (1969), pp. 221-230.

Hopkins, T.R. and Roberts, D.E., "Kufarev's Method for the Numerical Determination of the Schwartz-Christoffel Parameters," University of Kent (1978).

Howe, D., "The Application of Numerical Methods to the Conformal Transformation of Polygonal Boundaries," J. Inst. Maths. Applics. 12 (1973), 125-136.

Henrici, P., Applied and Computational Complex Analysis I, Wiley, 1974,

Kantorovich, L.V. and Krylov, V. I., Approximate Methods of Higher Analysis, P. Noordhoff Ltd. (The Netherlands), 1958.

Meyer, Eva-Suzanne, "Praktische Verfahren zur konformen Abbildung von Geradenpolygonen," Dissertation Universität Hannover, to appear (1979).

Powell, M.J.D., "A Fortran subroutine for solving systems of non-linear algebraic equations," Tech. Report AERE-R. 5947, Harwell, England (1968).

Vecheslavov, V. V. and Kokoulin, V.I., "Determination of the Parameters of the Conformal Mapping of Simply Connected Polygonal Regions," Zh. vychisl. Mat. mat. Fiz. 13, 4 (1973), 865-872 (Russian). Translated in U.S.S.R. Comp. Math, and Math. Phys. 13 (1973), no. 4, 57-65 (1974).

ACKNOWLEDGMENTS

This work was suggested and guided by Prof. Peter Henrici, without whom it would not have been possible.

Computations were performed at the Stanford Linear Accelerator Center, to whom thanks are due both for computer time and for the use of its excellent library of numerical software, Library routines used have been GAUSSQ (G.H. Golub & J. H. Welsch), NS01A (M. J. Powell), ODE (L. F. Shampine & M. K. Gordon), DCADRE (C. de Boor / IMSL), and PWSPLR (P. Swarztraubcr & R. Sweet). This report was printed at the Stanford Artificial Intelligence lab by Donald Knuth's mathematical typesetting system, TEX.

The author has benefited from discussions with Petter Bjørstad, William M. Coughran, Jr., Gene Golub, Eric Grosse, Randy LeVeque, and Cleve Molcr .

This research was supported in part by Office of Naval Research Contract N00014-75-C-1132 and in part by a National Science Foundation Graduate Fellowship.