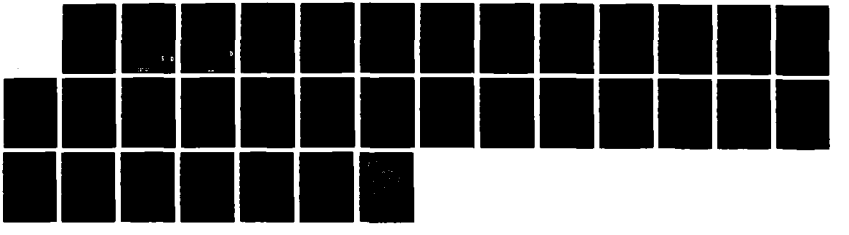
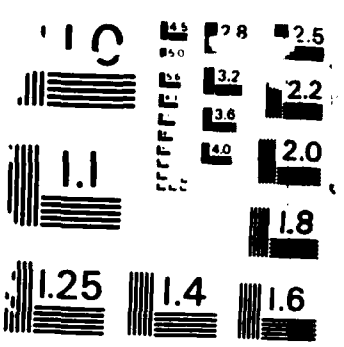
AD-A193 903 A FAST ALGORITHM FOR THE NUMERICAL EVALUATION OF CONFORMAL MAPPINGS(U) YALE UNIV NEW HAVEN CT DEPT OF COMPUTER SCIENCE S T O'DONNELL ET AL. JUL 87 1/1

UNCLASSIFIED YALEU/DCS/RR-554 N00014-86-K-0310 F/G 12/1 NL





TEST CHART

A Fast Algorithm for the Numerical Evaluation of Conformal Mappings

S. T. O'Donnell and V. Rokhlin†

Research Report YALEU/DCS/RR-554
July 1987

YALE UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

An algorithm is presented for the construction of conformal mappings from arbitrary simply-connected regions in the complex plane onto the unit disk. The algorithm is based on a combination of the Kerzman-Stein integral equation (see [11]) and the Fast Multipole Method for the evaluation of Cauchy-type integrals (see [15, 9, 3, 8]). Previously published methods of this type have an asymptotic CPU time estimate of the order $O(n^2)$, where n is the number of nodes in the discretization of the boundary of the region being mapped. The method we present has an estimate of the order $O(n)$, making it an approach of choice in many situations. The performance of the algorithm is illustrated by several numerical examples.

A Fast Algorithm for the Numerical Evaluation of Conformal Mappings

S. T. O'Donnell and V. Rokhlin†

Research Report YALEU/DCS/RR-554
July 1987

The authors were supported in part by ONR Grant N00014-86-K-0310.

Keywords: *Conformal Mapping, Szego kernel, Fast Multipole Method, Second Kind Integral Equation.*

†Dept. of Computer Science, Yale University, P.O. Box 2158, Yale Station, New Haven, Conn. 06520.

1. Introduction

Conformal mappings have been a popular analytical tool in the theory of partial differential equations for more than a century (see, for example, [7, 10, 13]). As a mathematical problem, the construction of a conformal mapping from a more or less arbitrary simply connected region onto a disk is solved satisfactorily by a whole range of methods (see [18, 10]). The connection between conformal mappings and Szegö and Bergman kernels has also been well known for a long time. For practical purposes, constructing a conformal mapping of a region Ω onto a disk is equivalent to evaluating a column of the Szegö kernel corresponding to Ω (see, for example, [7, 10, 17]).

Unfortunately, constructing conformal mappings (or evaluating Szegö kernels) numerically is by no means a trivial problem, and during the last several years the interest in it has increased significantly (see [18, 10]). Several algorithms have been proposed, and most of them encounter one of the following two difficulties. Most algorithms based on the Schwartz-Christoffel formula (or its variants) involve the iterative solution of a system of non-linear equations and suffer from the usual affliction of such methods. Namely, unless a good starting point is chosen, the method is likely to fail to converge. Algorithms based on linear first kind integral equations (see [14]) avoid this problem but lead to systems of linear algebraic equations with high condition numbers, with the attendant loss of precision, deterioration of convergence of iterative algorithms, etc.

The approach we take is based on the result originally published in [11], and is closely related to the subsequent work published in [19, 12]. Given a fairly general region Ω in C^1, a linear second kind integral equation is derived in [11], whose solution is one column of the Szegö kernel associated with Ω. The numerical solution of linear second kind integral equations leads to linear systems with asymptotically bounded condition numbers (see Section 3 below), and does not involve non-linear iterative schemes that might or might not converge. In [19, 12], the analytical apparatus of [11] is used to construct two versions of the algorithm for the numerical evaluation of conformal mappings of arbitrarily shaped simply-connected regions onto a disk. The version described in [12] requires order $O(n^3)$ operations to construct the mapping, where n is the number of nodes in the discretization of the boundary of the region, and the version described in [19] requires order $O(n^2)$ operations.

In this paper, we describe a modification of the algorithm of [19, 12] with an asymptotic operation count proportional to $O(n)$. The approach is based on discretizing the integral equation of [11] via the Nyström method, and solving the resulting system of linear algebraic equations by means of the conjugate residual algorithm, which is quite similar (but not identical) to the scheme used in [19]. Each iteration of the process requires that the matrix of the system be applied to a vector, which is normally an order n^2 procedure, since the matrix in question is dense. However, it turns out that the the Fast Multipole Method (see [15, 9, 3, 8]) can be used to speed up the process, resulting in an order $O(n)$ algorithm for the construction of a conformal mapping of an arbitrarily shaped region in C^1 onto a disk. A computer program implementing the procedure has been written, and we present numerical examples demonstrating that the computation times do indeed scale linearly with n, and that even large-scale problems result in acceptable computation times.

2. Mathematical Preliminaries

In this section we introduce the mathematical apparatus to be used in the remainder of the paper. D will denote the closed unit disk centered at the origin. Suppose that Ω is a simply connected bounded region in C^1 whose boundary $\partial\Omega$ is smooth, and a is a point in the interior of Ω. Suppose also that $\gamma : [0, L] \rightarrow R^2$ is a paramaterization of $\partial\Omega$ by its length. f_a will denote the conformal mapping $f_a : \Omega \rightarrow C^1$ such that $f_a(a) = 0$, $f_a'(a)$ is real and positive, and $f_a(\Omega) = D$. The existence and uniqueness of f_a are known as the Riemann mapping theorem and can be found,

for example, in [4].

In accordance with standard practice, we will denote by $L^2(\partial\Omega)$ the linear space of all square-integrable functions on $\partial\Omega$, and by $\mathcal{H}^2(\partial\Omega)$ the subspace of $L^2(\partial\Omega)$ consisting of all functions on $\partial\Omega$ that are restrictions of analytic functions on Ω.

The Cauchy projector associated with the region Ω is a mapping $\mathbf{H} : L^2(\partial\Omega) \to \mathcal{H}^2(\partial\Omega)$ defined by the formula

$$\mathbf{H}(u(w)) = \int_{\partial\Omega} H(w,z)u(z)d\sigma_z, \tag{2.1}$$

where

$$H(w,z) = \frac{1}{2\pi i}\frac{\gamma'(z)}{(z-w)} \tag{2.2}$$

is referred to as the Cauchy kernel, $\gamma'(z)$ is the counterclockwise unit tangent to $\partial\Omega$ at z, $d\sigma$ is the arc length on $\partial\Omega$, and $\gamma'(z)d\sigma_z = dz$. The Cauchy projector is an oblique one, and the orthogonal projector $\mathbf{S} : L^2(\partial\Omega) \to \mathcal{H}^2(\partial\Omega)$ is referred to as the Szegö projector. The Szegö projector is known to be a Cauchy type integral operator, i.e., it can be represented by the formula

$$\mathbf{S}(u(w)) = \int_{\partial\Omega} S(w,z)u(z)d\sigma_z, \tag{2.3}$$

and $S : \Omega \times \Omega \to C^1$ is usually referred to as the Szegö kernel.

The Szegö projector is related to the Cauchy projector by the formula

$$\mathbf{S} + (\mathbf{H}^* - \mathbf{H})\,\mathbf{S} = \mathbf{H}^*, \tag{2.4}$$

and therefore each column of the Szegö kernel satisfies the second kind integral equation

$$S(z,a) \; + \; \int_{t\in\partial\Omega} (\overline{H}(t,z) - H(z,t))\,S(t,a)\,d\sigma_t = \overline{H}(a,z), \tag{2.5}$$

with $z \in \partial\Omega$. The kernel of equation (2.5) is known as the Kerzman-Stein kernel, and is defined by the formula

$$\begin{aligned} A(z,w) &= \overline{H}(w,z) - H(z,w), \qquad for\ z \neq w \\ &= 0, \qquad for\ z = w. \end{aligned} \tag{2.6}$$

A is smooth, skew-Hermitian, and is identically zero if Ω is a disk.

Evaluation of Szegö kernels for regions in the complex plane is closely related to conformal mappings of such regions into the unit disk, and following are the classical formulae implementing this connection. For any $a \in \Omega\backslash\partial\Omega$,

$$S(a,a) = \int_{\partial\Omega} \overline{S(z,a)}\,S(z,a)d\sigma_z. \tag{2.7}$$

For any $z \in \Omega$,

$$f_a'(z) = 2\pi\,\frac{S^2(z,a)}{S(a,a)}. \tag{2.8}$$

For any $z \in \partial\Omega$,

$$f_a(z) = \frac{\gamma'(z)}{i}\,\frac{f_a'(z)}{|f_a'(z)|}. \tag{2.9}$$

Remark 2.1.

Obviously, given the function $\Psi(z) = S(a,z)$, the conformal mapping $f_a : \Omega \to D$ can be easily obtained either by quadratures using formulae (2.8) and (2.7) or directly by combining the expressions (2.8), (2.7) and (2.9). Therefore, the problem of finding f_a has been reduced to the problem of solving equation (2.5).

The proofs of all results in this section can be found in [11] and [12].

3. Numerical Preliminaries

In this section, the numerical techniques used to solve the integral equation (2.5) and compute the mapping f_a on the boundary of Ω are described. The Nyström algorithm is used to discretize the integral equation, and the conjugate residual algorithm is used to solve the resulting system of complex linear algebraic equations. Each iteration of the conjugate residual algorithm includes a matrix-vector multiplication, which in existing algorithms has a cost proportional to n^2. This is the only computation which forces the operation count in the conjugate residual algorithm to be proportional to n^2 rather than n. The Fast Multipole Method (see [15, 9, 3]) provides a technique for multiplying Hilbert matrices by arbitrary vectors in time proportional to n, and we use this method to rapidly apply the matrix to the vector in the conjugate residual algorithm (see Section 3.2). Once the equation (2.5) is solved numerically, we obtain the mapping f_a by combining formulae (2.7), (2.8) and (2.9).

3.1. The trapezoidal quadrature rule for periodic functions. We will define an n-point quadrature rule η on the interval $[0, L]$ as a finite sequence of pairs $\{x_i, w_i\}$, $i = 1, \ldots, n$, where $x_i \in [0, L]$ for all $i \in [1, n]$. For a function $\phi : [0, L] \to C^1$, we will view the sum

$$\eta(\phi) = \sum_{i=1}^{n} w_i \cdot \phi(x_i) \tag{3.1}$$

as an approximation to the integral

$$\int_0^L \phi(x)\, dx. \tag{3.2}$$

The family of quadrature formulae $\eta_n = \{x_{ni}, w_{ni}\}$, $i = 1, \ldots, n$, has a rate of convergence m ($m \geq 1$) for the function $\phi : [0, L] \to R^1$ if there exist $A > 0$ and $N > 0$ such that

$$|\eta_n(\phi) - \int_0^L \phi(x)\, dx| < \frac{A}{n^m} \tag{3.3}$$

for all $n > N$. The n-point trapezoidal quadrature rule is defined by the formulae

$$x_i = (i-1)\frac{L}{n} \qquad for \quad i = 1, \ldots, n, \tag{3.4}$$

$$w_1 = w_n = \frac{L}{2n} \qquad and \qquad w_i = L/N \qquad for \quad i \in [2, n-1].$$

Using the Euler-MacLaurin formula, it is easy to show that if the function $\phi : [0, L] \to R^1$ is periodic and has k continuous periodic derivatives ($k \geq 1$) then the trapezoidal rule for this function has order of convergence $k + 1$.

3.2. The Nyström algorithm. The Nyström algorithm associated with an n-point quadrature formula $\eta = \{z_i, w_i\}$ replaces the integral equation

$$\phi(z) + \int_0^L K(z,t) \cdot \phi(t) dt = g(z) \tag{3.5}$$

with the system of linear algebraic equations

$$\phi(z_i) + \sum_{j=1}^{n} w_j \cdot K(z_i, z_j) \cdot \phi(z_j) = g(z_i), \tag{3.6}$$

with $i = 1, \ldots, n$. We denote the matrix of the discretized system (3.6) by B_n, and view the solution $\phi_1, \ldots, \phi_n$ of (3.6) as an approximation to the solution ϕ of (3.5) at the nodes $z_1, \ldots, z_n$. If (3.5) has a unique solution, then for a wide class of quadrature formulae η_n and sufficiently large n, system (3.6) also has a unique solution. Furthermore, under fairly broad assumptions, the convergence rate of the Nyström algorithm is the same as the convergence rate of the quadrature formula which it is based on (see [2]).

Applying the Nyström algorithm based on the n-point trapezoidal quadrature rule to the equation (2.5), we obtain the system of equations

$$\phi_i + \frac{L}{n} \sum_{j=1}^{n} A(z_i, z_j) \cdot \phi_j = \overline{H}(a, z_i), \tag{3.7}$$

with $i = 1, \ldots, n$, where ϕ_j is viewed as an approximation to $S(z_j, a)$. The system of equations (3.7) can be written as the linear system

$$B_n \psi = \overline{H}_{n,a}, \tag{3.8}$$

where ψ is viewed as the approximation to one column of the Szegö kernel $(S(z_1, a), \ldots, S(z_n, a))$ and $\overline{H}_{n,a} = (\overline{H}(a, z_1), \ldots, \overline{H}(a, z_n))$.

By the following theorem (see, for example, [2]), the condition number of the matrix B_n is asymptotically bounded.

Theorem 3.1. *Suppose that $K : [0, L] \times [0, L] \to R^1$ is a c^2-function and that equation (3.5) has a unique solution. Suppose further that the system of linear equations (3.6) with corresponding coefficient matrix B_n has been obtained by applying the Nyström algorithm based on the trapezoidal quadrature rule to (3.5). Then*

$$\lim_{n \to \infty} k(B_n) = a, \tag{3.9}$$

where $0 < a < \infty$ is some real number and $k(B_n)$ denotes the condition number of the matrix B_n.

3.3. Representation of the boundary of Ω. The discretized equation (3.7) assumes that the nodes $z_1, \ldots, z_n$ are equispaced along the boundary $\partial\Omega$. In our implementation, we assume that the boundary of Ω is specified numerically by nodes $w_1, \ldots, w_m$ for some $m > 0$. To generate the equispaced nodes, we resample the curve by passing periodic splines under tension through the nodes $w_1, \ldots, w_m$, and generating nodes $z_1, \ldots, z_n$ which are equispaced in arc length along the splines.

3.4. Rapid application of the matrix B_n to arbitrary vectors. In this section we describe the application of the Fast Multipole Method (see [9, 3]) to the multiplication of the matrix B_n of equation (3.8) by the vector ψ in time proportional to n.

For points $z_1, \ldots, z_n \in C^1$, a Hilbert matrix H is defined by the formula

$$\begin{aligned} H(i,j) &= \frac{1}{z_i - z_j}, \qquad for\ i \neq j, \\ &= 0 \qquad for\ i = j. \end{aligned} \tag{3.10}$$

A description of the Fast Multipole Method (FMM) together with the proof of the following theorem can be found in [3].

Theorem 3.2. *Let A denote a Hilbert matrix of order n, and let x be an arbitrary vector. The Fast Multipole Method applies the matrix A to vector x in $O(n \cdot log(\frac{1}{\epsilon}))$ operations where ϵ is the precision of calculations.*

Suppose now that we would like to efficiently apply the matrix B_n of system (3.8) to a vector. Due to (2.2) and (2.6), the equations (3.7) can be rewritten in the form

$$\psi_i \; + \; \frac{L}{n} \frac{\overline{\gamma}'(z_i)}{\overline{2\pi i}} \sum_{j \neq i} \frac{1}{(\overline{z}_i - \overline{z}_j)} \cdot \psi_j \; + \; \frac{L}{n} \frac{1}{2\pi i} \sum_{j \neq i} \frac{1}{(z_i - z_j)} \, (\psi_j \cdot \gamma'(z_j)) = \overline{H}(a, z_i). \tag{3.11}$$

We denote the Hilbert matrix associated with the points $\overline{z}_1, \ldots, \overline{z}_n$ by U_n, and the Hilbert matrix associated with the points $z_1, \ldots, z_n$ by V_n. We denote the vector $(\psi_1 \cdot \gamma'(z_1), \ldots, \psi_n \cdot \gamma'(z_n))$ by χ. It is easy to see from formula (3.11) that

$$B_n \, \psi \; = \; \psi \; + \; c_1 \, U_n \, \psi \; + \; c_2 \, V_n \, \chi, \tag{3.12}$$

with

$$c_1 = \frac{L}{n} \frac{\overline{\gamma}'(z_i)}{\overline{2\pi i}} \qquad \textit{and} \qquad c_2 = \frac{L}{n} \frac{1}{2\pi i}. \tag{3.13}$$

From equation (3.12) it is clear that two applications of a Hilbert matrix to a vector are required to apply the matrix B_n to ψ. Consequently, the matrix B_n can be applied to the vector ψ in order $O(n)$ operations.

3.5. Conjugate residual algorithm. The conjugate residual method (CR) is an iterative method for solving a system of linear equations $Ax = f$ with $x, f \in C^n$ and A a normal $n \times n$-matrix (see [5]). We denote the i^{th} iterate by x_i and the residual at the i^{th} step by $r_i = y - Ax_i$. At the i^{th} step the CR algorithm minimizes the residual $||r_i||_2$ over the i-dimensional subspace of C^n spanned by the vectors $r_0, Ar_0, \ldots, A^{i-1}r_0$.

From equation (3.7), it is clear that the matrix B_n of system (3.8) is normal since it is the sum of the identity matrix and a skew-Hermitian matrix, and hence the conjugate residual algorithm can be used to solve (3.8) (see [5]). Moreover, if $E_2(x_i)$ denotes the L^2-error at the i^{th} iterate, it is known [6] that

$$E_2(x_i) \; \leq \; 2 \, [\frac{1 - 1/k(B_n)}{1 + 1/k(B_n)}]^i \, E_2(x_0). \tag{3.14}$$

Combining (3.14) with Theorem 3.1, it is easy to see that the convergence rate of the conjugate residual algorithm applied to (3.8) is independent of n.

4. Description of the algorithm

Following is a description of the algorithm for the numerical evaluation of the mapping $f_a : \partial\Omega \rightarrow \partial D$.

- Resample $\partial\Omega$ into equispaced nodes using periodic splines unter tension as the basis for interpolation.
- Apply the conjugate residual algorithm to solve the linear system $B_n\psi = \overline{H}_{n,a}$ which arises from the discretized system (3.7).
- In the conjugate residual routine, apply the matrix B_n to the vector using formulae (3.12) and (3.13) where the Fast Multipole Method is used to compute the Hilbert matrix-vector products $U_n\psi$ and $V_n\chi$.
- Use formula (2.8) to compute f_a' and a combination of formula (2.9) and recursive application of the modified trapezoidal rule (see Section 5) to compute f_a.

5. Details of the Implementation

For a wide class of regions Ω, the conformal mapping $f_a : \partial\Omega \to \partial D$ compresses much of the boundary of Ω into very small segments of the unit circle. This phenomenon is known as crowding, and is an inherent property of the conformal mapping. In areas of crowding, the derivatives f_a' can be very small, which has serious numerical implications. Where crowding is significant, the derivative $\frac{1}{f_a'}$ of the mapping $f_a^{-1} : \partial D \to \partial\Omega$ is sufficiently large that f_a^{-1} cannot be numerically computed from f_a and f_a'.

However, the numerical computation of f_a from f_a' is possible. Once an approximation to the Szegö kernel has been computed, the derivative f_a' and $f_a : \partial\Omega \to \partial D$ can generally be computed using equations (2.8) and (2.9). However, suppose that for some integer $i < n$, $z_i \in \partial\Omega$ is a point in a region of crowding. Then the analytic values of f_a' are very small relative to values of f_a' elsewhere on the boundary (see Section 6). Therefore, the numerical approximations of $S(z_i, a)$ and $f_a'(z_i)$ are likely to have a high relative error, rendering formula (2.9) useless (see Section 7).

Clearly, an alternate method of computing f_a is integration along the boundary $\partial\Omega$, and we use the modified trapezoidal rule with 4^{th} order corrections at the endpoints (see [1]) given by the formula

$$\begin{aligned}\int_{x_1}^{x_n} f(x)dx = &\sum_{j=1}^{n} f(x_i) \\ &- \frac{1}{2}f(x_1) - \frac{1}{24}(f(x_3) - 4f(x_2) + 3f(x_1)) \\ &- \frac{1}{2}f(x_n) - \frac{1}{24}(f(x_{n-2}) - 4f(x_{n-1}) + 3f(x_n)).\end{aligned} \tag{5.1}$$

By separately storing the right endpoint correction and the remaining sum in formula (5.1), the calculation of the integral from z_1 to z_{k+1} can make use of the calculation of the integral from z_1 to z_k, and therefore the modified trapezoidal rule can be implemented recursively, resulting in an order $O(n)$ procedure.

Other than in areas of crowding, the formula (2.9) can be expected to be more accurate than the trapezoidal rule. We combine the two methods, using formula (2.9) wherever the derivative is above a threshold value, and recursive integration via the modified trapezoidal rule otherwise.

6. Numerical Results

A FORTRAN program has been written implementing the algorithm of this paper. It has been tested on a variety of regions, and in this section, we present the results of six such numerical experiments. Tables 1 - 6 contain the quantitative information pertaining to the Examples 1 - 6 respectively. The first column of each table contains the number n of nodes in the discretization of the boundary of the region. The second column contains the CPU time the algorithm took on a VAX-8600 computer. The third column contains the number of iterations the conjugate residual process took to achieve 5-digit precision. The fourth and fifth columns contain the relative accuracies in L^2 and L^∞ norms respectively. Finally, the sixth column of each table contains the ratios

$$\frac{\min_{z\in\Gamma} \mid f'(z) \mid}{\max_{z\in\Gamma} \mid f'(z) \mid} \tag{6.1}$$

as determined by the calculation with the corresponding n.

Remark 6.1. Since no exact values are available for any of the conformal mappings constructed in the Examples 1-6 below, evaluating the errors and convergence rates presents a certain

difficulty. We have used a standard remedy, and estimated the errors (both L^2 and L^∞) by the difference between two successive discretizations of the same problem.

For each example, we present a set of figures depicting the equispaced nodes on the boundary of the original region (left column of Figures 1-6) and their images on the circle of radius 1 (right column of Figures 1-6). Following is a more detailed description of each of the examples.

Example 1. An ellipse with the aspect ratio 2:1. The algorithm has been applied to this region with $n = 16, 32, \ldots, 512$. The results for this set of experiments are summarized in Table 1, and are illustrated in Figure 1.

Example 2. An ellipse with the aspect ratio 3:1. The algorithm has been applied to this region with $n = 16, 32, \ldots, 512$. The results for this set of experiments are summarized in Table 2, and are illustrated in Figure 2.

Example 3. An ellipse with the aspect ratio 10:1. The algorithm has been applied to this region with $n = 16, 32, \ldots, 512$. The results for this set of experiments are summarized in Table 3, and are illustrated in Figure 3.

Example 4. A snake-shaped region. The number of nodes required to achieve reasonable accuracy is larger than that of the ellipse examples above. Results for the snake-shaped region are presented for $n = 128, 256, \ldots, 4096$. The data for this region are presented in Table 4 and the corresponding plots are presented in Figure 4.

Example 5. A spiral-shaped region. As in Example 4, many nodes along the boundary are required to achieve reasonable accuracy. In this example, results are presented for $n = 200, 400, 800, \ldots, 6400$. The data for this experiment are given in Table 5, and the corresponding plots are presented in Figure 5.

Example 6. A rounded square. The algorithm has been applied to this region with $n = 16, 32, \ldots, 512$. The data for this example are presented in Table 6, and the plots are presented in Figure 6.

The following observations can be made from the Tables 1 - 6 and the Figures 1 - 12.

1. The CPU time requirements of the algorithm grow linearly with n, though the growth is somewhat erratic.

2. The number of iterations of the conjugate residual process is virtually independent of the number of nodes in the discretization of the boundary of the region.

3. Even for regions of fairly complicated shapes, the number of iterations of the conjugatde residial process does not become excessive.

4. The algorithm does not cause substantial round-off errors, and single precision arithmetic can be used. This has been verified by repeating the calculations in double precision. This observation is typical for algorithms based on second kind integral equations (see, for example, [2], [15], and is in agreement with observations made in [19, 12].

5. Even for fairly large-scale calculations ($n > 4000$), the actual CPU times required by the algorithm are quite acceptable (about 10 minutes on a VAX-8600 for $n = 4098$, and less than 1 hour for $n = 20000$). Since the boundary of a very wide range of regions can be adequately discretized by 20000 nodes, the algorithm can be viewed as a nearly universal tool for mapping smooth regions in the complex plane onto the unit disk.

7. Crowding and Related Issues

Inspection of column 6 of the Tables 1 - 6 yields a disturbing observation. Namely, even for fairly uncomplicated regions (such as an ellipse with an aspect ratio 10:1) the ratio (6.1) tends to be extremely small. The effect is known in the literature under the name of crowding (see [20]), and has received much less attention than the authors feel it deserves. Consider, for example, Figures 4,5. It is obvious that nearly all of the boundary of the "spiral" is compressed onto an extremely small part of the circle. For example, there are 1600 nodes in the discretization of the boundary

of the "spiral" in the left column of Figure 5, yet only 42 images of these nodes can be discerned in the right column of the figure. Similar observations can be made about the "snake" in Figure 4 and about the ellipse with an aspect ratio 10:1 depicted in Figure 3. Even for the ellipse with an aspect ratio 3:1 depicted in Figure 2, the ratio (6.1) is of an order 10^{-2}.

One interpretation of the this observation is that while the mapping $f : \Omega \to D$ can be constructed numerically for a wide class of regions Ω, the mapping $f^{-1} : D \to \Omega$ is a numerically unmanageable object even for fairly mild Ω. This fact has serious repercussions for attempts to use the conformal mappings as a numerical (as opposed to analytical) tool for the solution of elliptic partial differential equations (PDEs) in general regions in two dimensions. Indeed, a standard prescription for using conformal mappings to solve an elliptic PDE (such as Laplace or Poisson equation) in a two-dimensional region is as follows.

a) Find a conformal mapping $f^{-1} : D \to \Omega$ and use it to construct an induced PDE on the disk D.

b) Find the solution ϕ_D of the induced PDE on D via an appropriate numerical technique.

c) Construct the mapping $f : \Omega \to D$ and evaluate the solution ϕ of the original PDE on Ω via the formula

$$\phi(x) = \phi_d(f(x)). \tag{7.1}$$

Obviously, if the mapping f^{-1} can not be constructed, step a) in the above procedure becomes impossible.

Remark 7.1

In fact, it is known that for a region Ω, the ratio (6.1) decays exponentially as a function of the aspect ratio of Ω (see, for example, [20]). It is also known that the mapping f^{-1} viewed as a function of the region Ω is not even continous (see [16]). Therefore, the poor numerical behaviour of f^{-1} is not surprising.

Remark 7.2

Obviously, if the region Ω has an aspect ratio close to 1 (i.e. it does not deviate from the disk too much), conformal mappings can be an effective numerical device for dealing with elliptic PDEs on Ω. For regions with high aspect ratios, it is natural to attempt to use model regions other than the disk (such as rectangles for which Fast Poisson Solvers and other efficient tools are available). While it is easy to see that no model region can solve the problem of crowding in general, it is clear that certain classes of regions can be handled in this manner.

8. Conclusions

The algorithm of the present paper appears to be the most efficient of presently available numerical tools for the construction of conformal mappings from complicated regions in the complex plane onto the unit disk. The algorithm is based on a combination of the approach of [11, 12, 19] with the Fast Multipole Method (FMM) described in [15, 9, 3], and has an asymptotic CPU time estimate $O(n)$, where n is the number of nodes in the discretization of the boundary of the region.

On the other hand, our experiments strongly indicate that the phenomenon known as the crowding makes the use of conformal mappings as a numerical tool difficult in some cases and impossible in others. It is unclear to the authors how this difficulty can be overcome.

The authors would like to thank Professor R.R. Coifman and Dr. L.F. Greengard for their interest and help.

References

[1] M. Abramowitz, I. Stegun ed., *Handbook of Mathematical Functions With Formulas, Graphs, and Mathematical Tables,* U.S. Government Printing Office. National Bureau of Standards Applied Mathematics Series 55.

[2] K.E. Atkinson, *A Survey of Numerical Methods for the Solution of Fredholm Integral Equations of the Second Kind,* SIAM, Philadelphia, Pa., 1976.

[3] J. Carrier, L. Greengard, V. Rokhlin, *A Fast Adaptive Multipole Algorithm for Particle Simulations,* Technical Report YALEU/DCS/RR-496, Yale Univ. Dept. of Computer Science, 1986.

[4] J.B. Conway, *Functions of One Complex Variable,* Springer-Verlag New York Inc., 1973.

[5] S.C. Eisenstat, H.C. Elman, M.H. Schultz, *Variational Iterative Methods for Nonsymmetric Systems of Linear Equations,* SIAM, J. Numer. Anal., 20, No. 2 (1983), pp. 345–357.

[6] H.C. Elman, *Iterative Methods for Large, Sparse, Nonsymmetric Systems of Linear Equations,* Ph.D. Thesis, Yale Univ. Dept. of Computer Science, 1982. Also available as Research Report 229.

[7] D. Gaier, *Konstruktive Methoden der konformen Abbildung,* Springer Heidelberg, 1964.

[8] L.F. Greengard, *The Rapid Evaluation of Potential Fields in Particle Systems,* Ph.D. Thesis, Yale Univ. Dept. of Computer Science, 1987. Also available as Research Report YALEU/DCS/RR-533.

[9] L. Greengard, V. Rokhlin, *A Fast Algorithm for Particle Simulations,* J. Comp. Phys., (to appear). Also available as Yale Univ. Dept. of Computer Science Research Report YALEU/DCS/RR-459.

[10] P. Henrici, *Applied and Computational Complex Analysis Vol. 3,* John Wiley, New York, 1986.

[11] N. Kerzman, E.M. Stein, *The Cauchy Kernel, the Szego Kernel, and the Riemann Mapping Function,* Math. Ann., 236 (1978), pp. 85–93.

[12] N. Kerzman, M.R. Trummer, *Numerical conformal mapping via the Szego kernel,* J. Comp. Appl. Math., 14 (1986), pp. 111–123.

[13] Z. Nehari, *Conformal Mapping,* Dover Publications, N.Y., 1952.

[14] L. R . ..el, *A fast method for solving certain integral equations of the first kind with application to conformal mapping,* J. of Comp. Appl. Math., 14 (1986), pp. 125–142.

[15] V. Rokhlin, *Rapid Solution of Integral Equations of Classical Potential Theory,* J. Comp. Phys., 60 (1985), pp. 187–207.

[16] S.W. Semmes, *The Cauchy Integral, Chord-Arc Curves, and Quasiconformal Mappings,* Am. Math. Soc., The Bieberbach Conjecture, Proceedings of the Symposium on the Occasion of the Proof, Mathematical surveys and monographs, ISSN 0076-5376 no.21 (1986), pp. 167–183.

[17] G. Szegö, *Orthogonal Polynomials,* Am. Math. Soc. Colloq. Publ., 23 (1939).

[18] L.N. Trefethen ed., *Numerical Conformal Mapping,* Elsevier Science Publishers B.V., 1986. Reprinted from J. Comp. Appl. Math..

[19] M.R. Trummer, *An Efficient Implementation of a Conformal Mapping Method Based on the Szego Kernel,* SIAM J. Numer. Anal., 23, No. 4 (1986), pp. 853–872.

[20] C. Zemach, *A conformal map formula for difficult cases,* J. Comp. Appl. Math., 14 (1984), pp. 207–215.

nodes	CPU time	iterations	L^2-error	L^∞-error	crowding
16	4.4	4	.59E-3	.17E-2	.089
32	4.6	4	.17E-4	.49E-4	.089
64	5.6	4	.19E-5	.69E-5	.089
128	8.5	4	.64E-6	.11E-5	.089
256	14.6	4	.19E-5	.39E-5	.089
512	22.2	4			.089

Table 1: A 2:1 Ellipse

nodes	CPU time	iterations	L^2-error	L^∞-error	crowding
16	11.5	5	.12E-1	.22E-1	.10E-1
32	11.8	5	.60E-3	.10E-2	.97E-2
64	13.2	5	.25E-4	.48E-4	.97E-2
128	17.0	5	.15E-5	.26E-5	.97E-2
256	25.0	5	.59E-6	.12E-5	.97E-2
512	35.6	5			.97E-2

Table 2: A 3:1 Ellipse

nodes	CPU time	iterations	L^2-error	L^∞-error	crowding
32	13.4	8	.32E+0	.37E+0	.60E-7
64	15.3	9	.33E-2	.39E-2	.11E-8
128	21.3	9	.20E-3	.20E-3	.10E-10
256	33.5	9	.12E-4	.12E-4	.95E-9
512	57.8	9	.82E-6	.85E-6	.94E-9
1024	121.7	9	.64E-6	.66E-6	.94E-9

Table 3: A 10:1 Ellipse

nodes	CPU time	iterations	L^2-error	L^∞-error	crowding
128	29.6	17	.32E+1	.11E+1	.20E-8
256	51.1	17	.70E+0	.71E+0	.17E-10
512	95.7	17	.10E-1	.10E-1	.30E-14
1024	203.1	17	.51E-3	.51E-3	.15E-15
2048	369.7	17	.54E-4	.54E-4	.98E-16
4096	652.1	17			.36E-15

Table 4: A Snake-shaped Region

nodes	CPU time	iterations	L^2-error	L^∞-error	crowding
200	42.9	18	.11E+1	.11E+1	.26E-13
400	81.8	17	.11E+0	.11E+0	.77E-14
800	127.3	16	.19E-2	.19E-2	.21E-14
1600	252.8	16	.12E-3	.12E-3	.13E-14
3200	521.6	16	.68E-4	.68E-4	.12E-14
6400	978.5	16			.11E-15

Table 5: A Spiral-shaped Region

nodes	CPU time	iterations	L^2-error	L^∞-error	crowding
16	6.9	6	.30E-1	.52E-1	.86E-1
32	7.9	6	.83E-2	.14E-1	.47E-1
64	9.5	6	.17E-2	.43E-2	.28E-1
128	13.7	6	.35E-3	.47E-3	.20E-1
256	18.8	6	.12E-4	.45E-4	.21E-1
512	33.0	6			.20E-1

Table 6: A Rounded Square

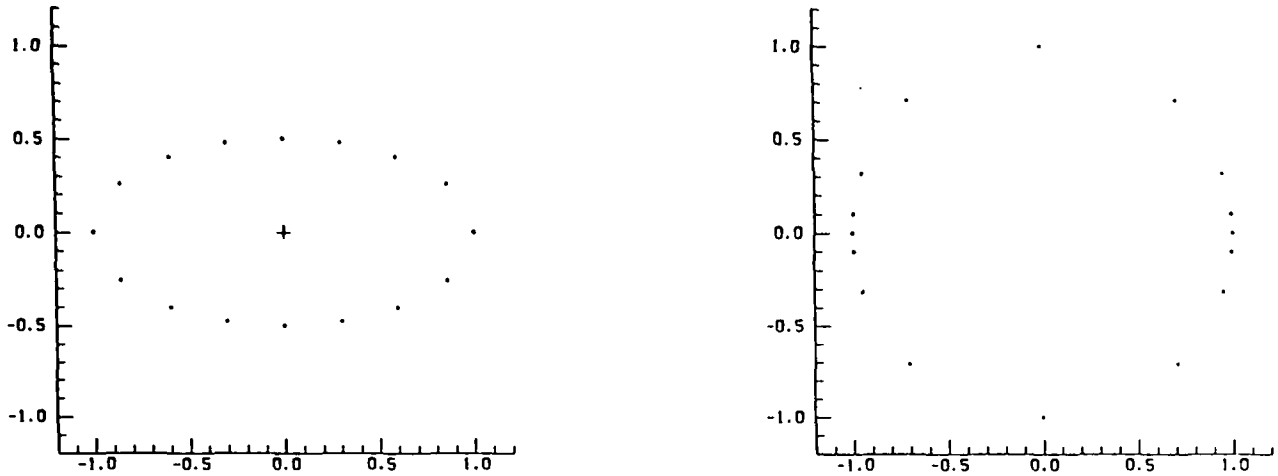


Figure 1a
A 2:1 Ellipse, n = 16.

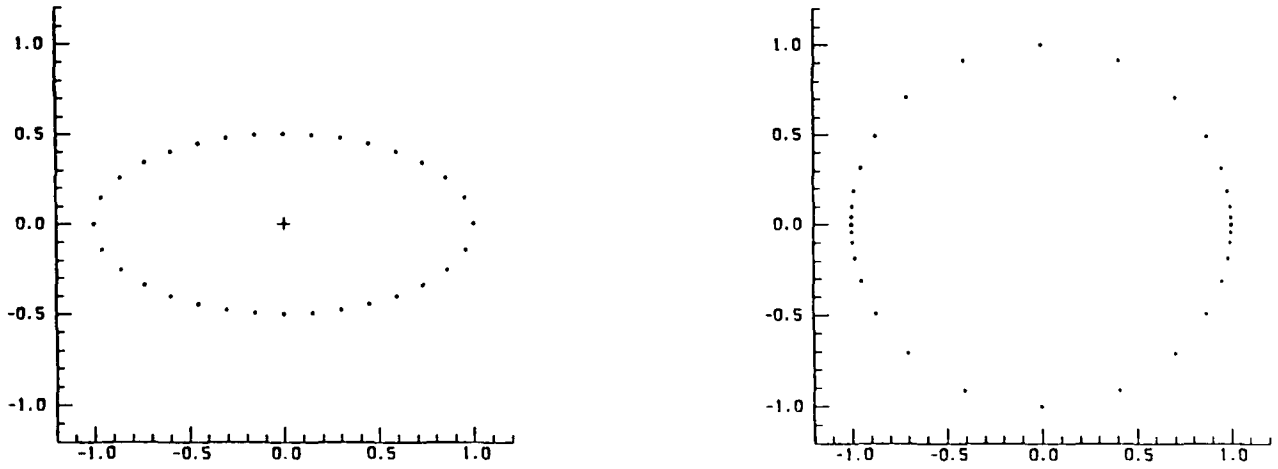


Figure 1b
A 2:1 Ellipse, n = 32.

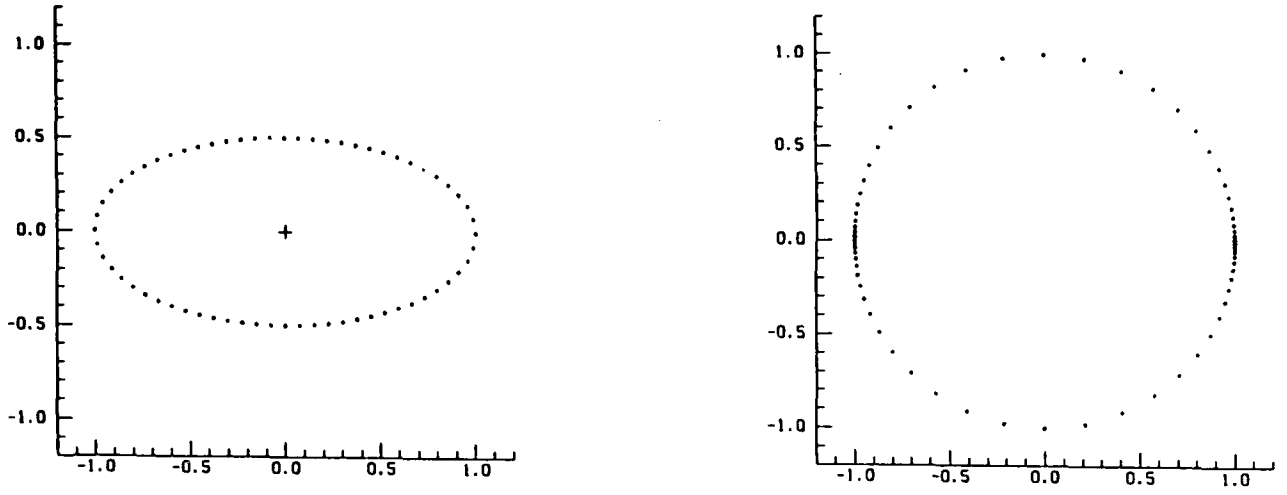


Figure 1c
A 2:1 Ellipse, n = 64.

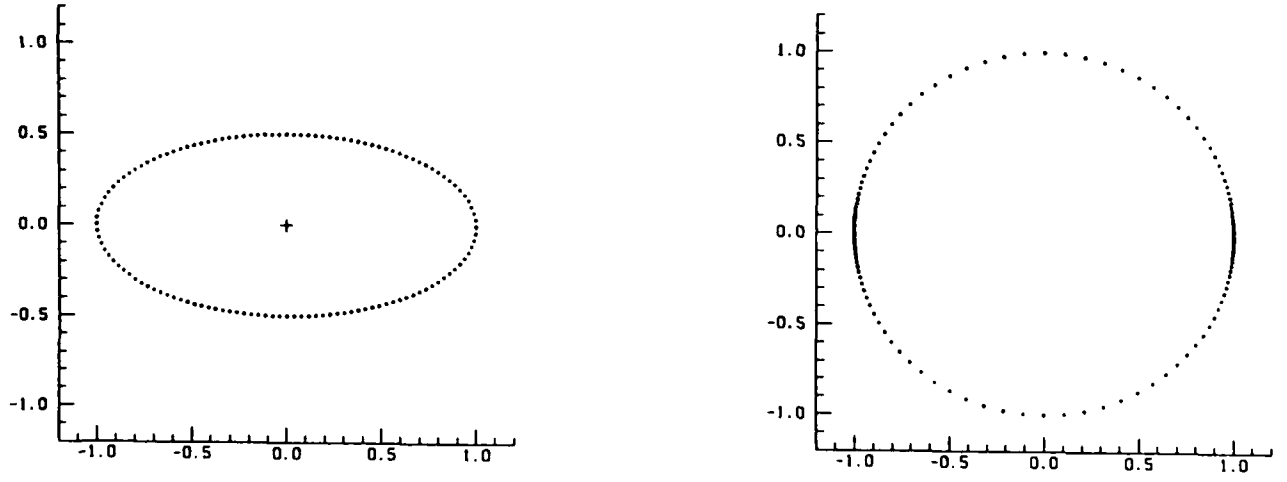


Figure 1d
A 2:1 Ellipse, n = 128.

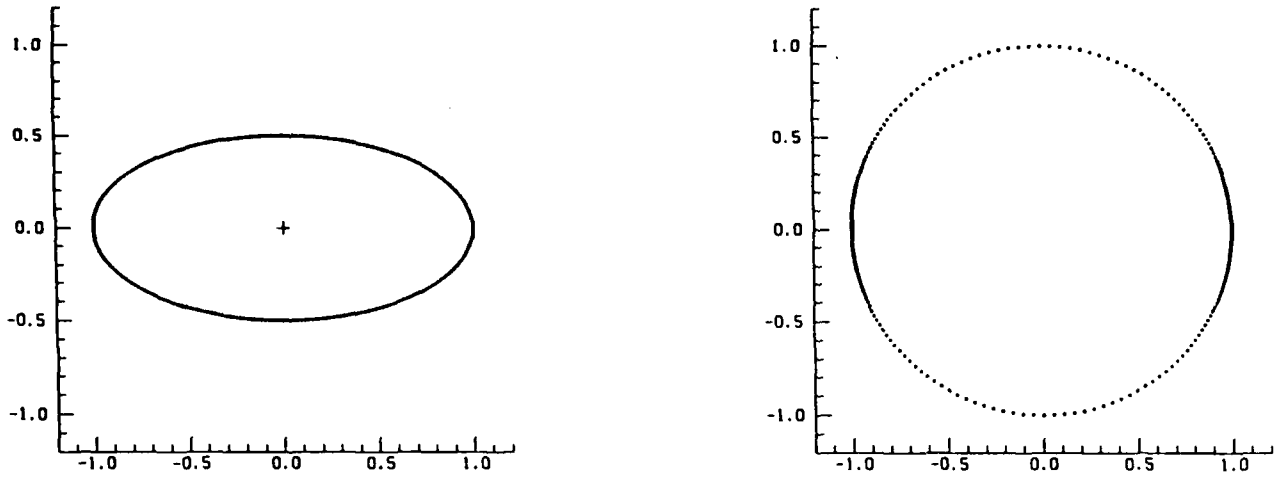


Figure 1e
A 2:1 Ellipse, n = 256.

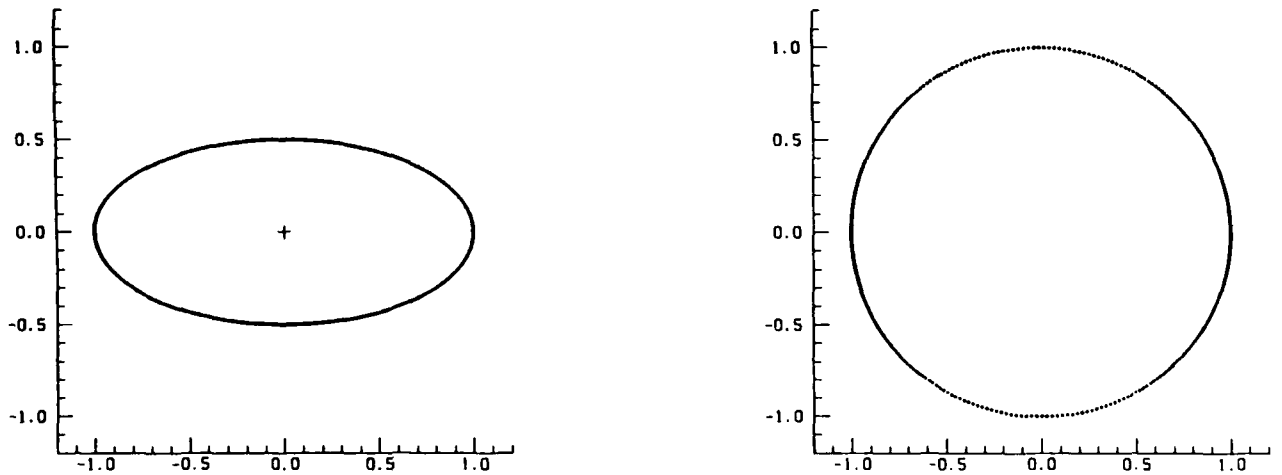


Figure 1f
A 2:1 Ellipse, n = 512.

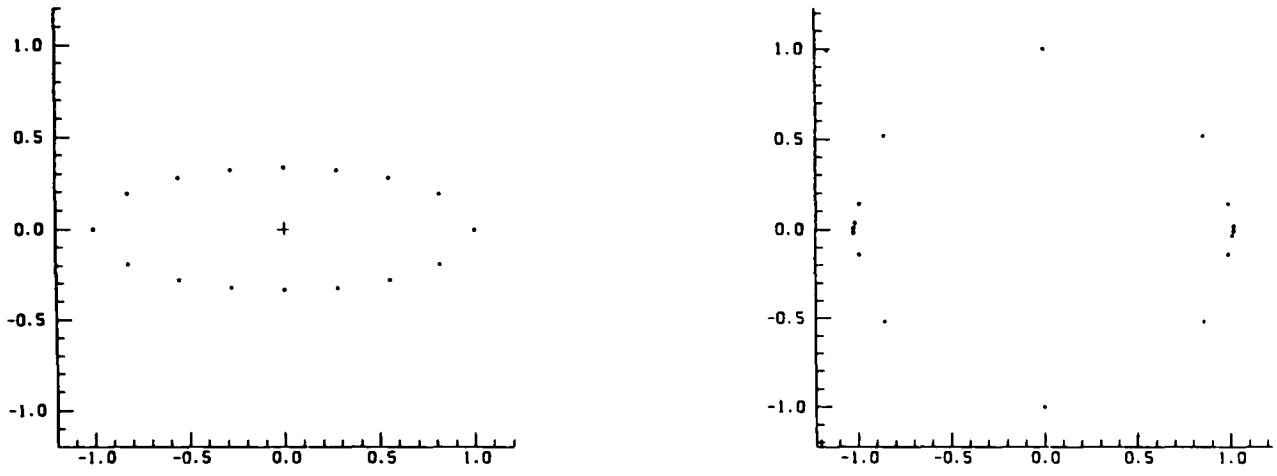


Figure 2a
A 3:1 Ellipse, n = 16.

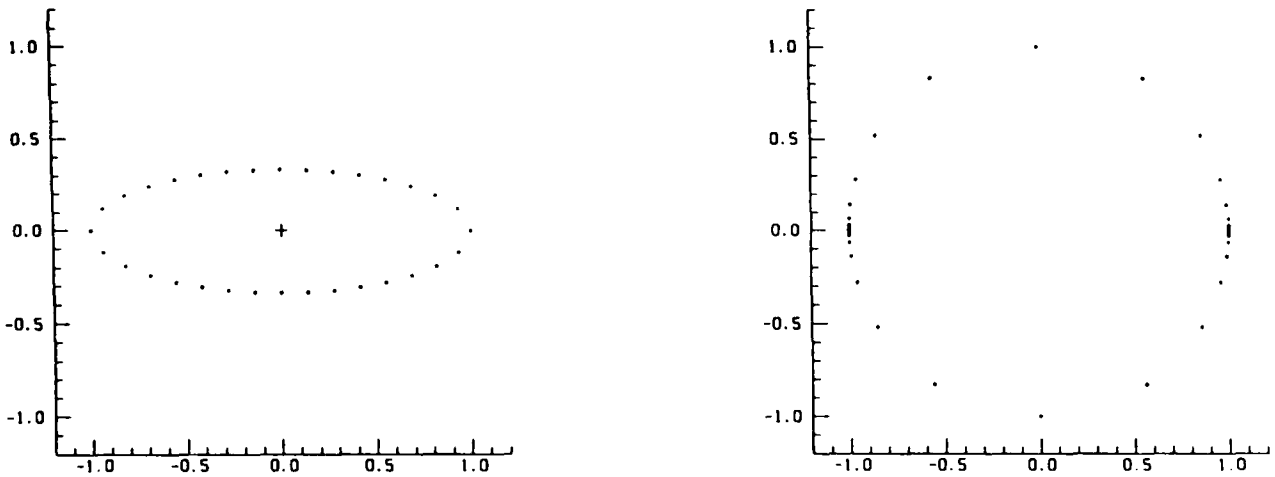


Figure 2b
A 3:1 Ellipse, n = 32.

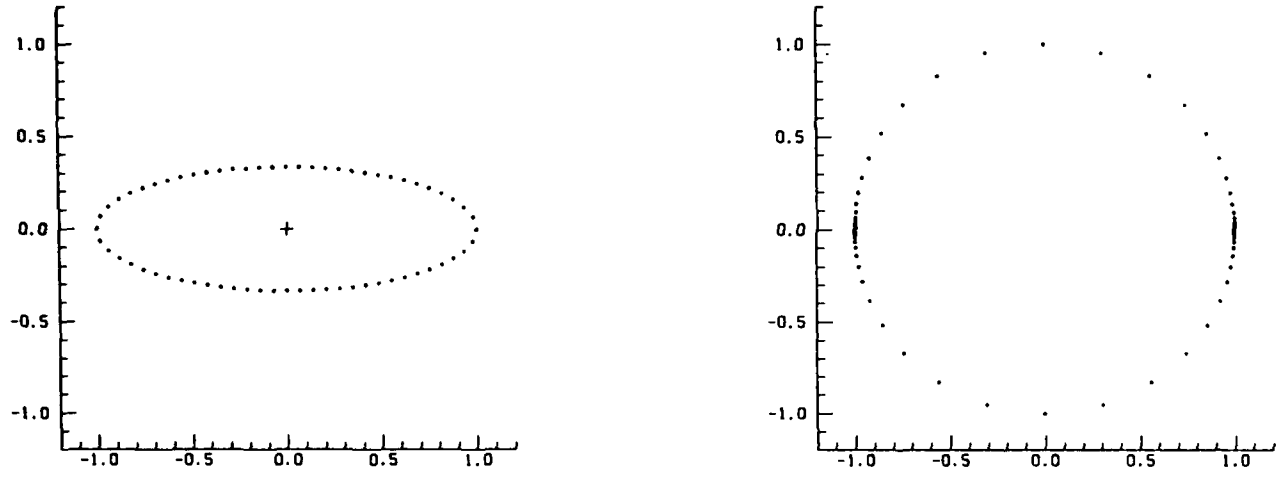


Figure 2c
A 3:1 Ellipse, n = 64.

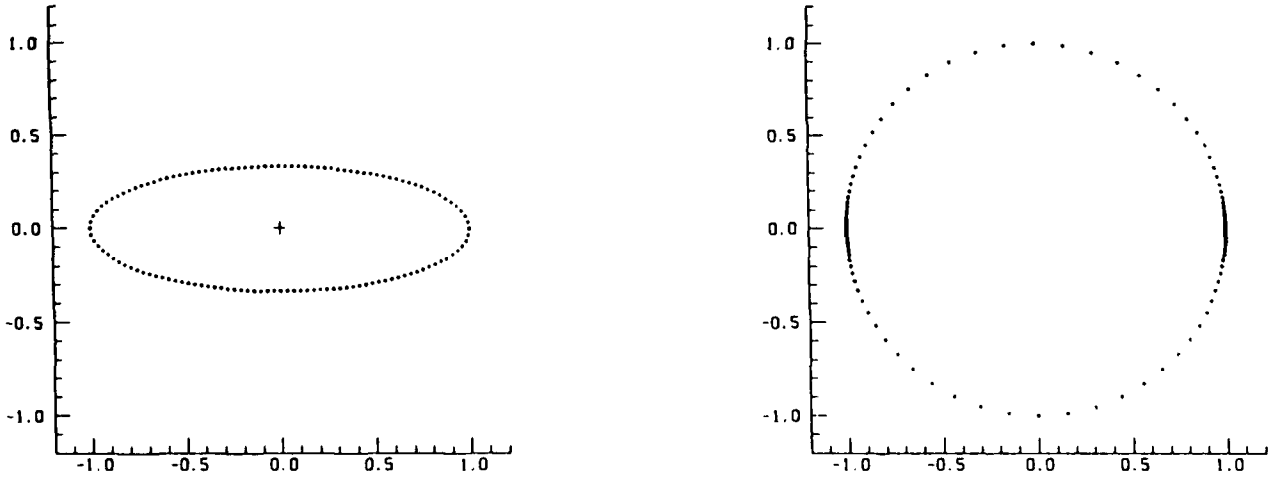


Figure 2d
A 3:1 Ellipse, n = 128

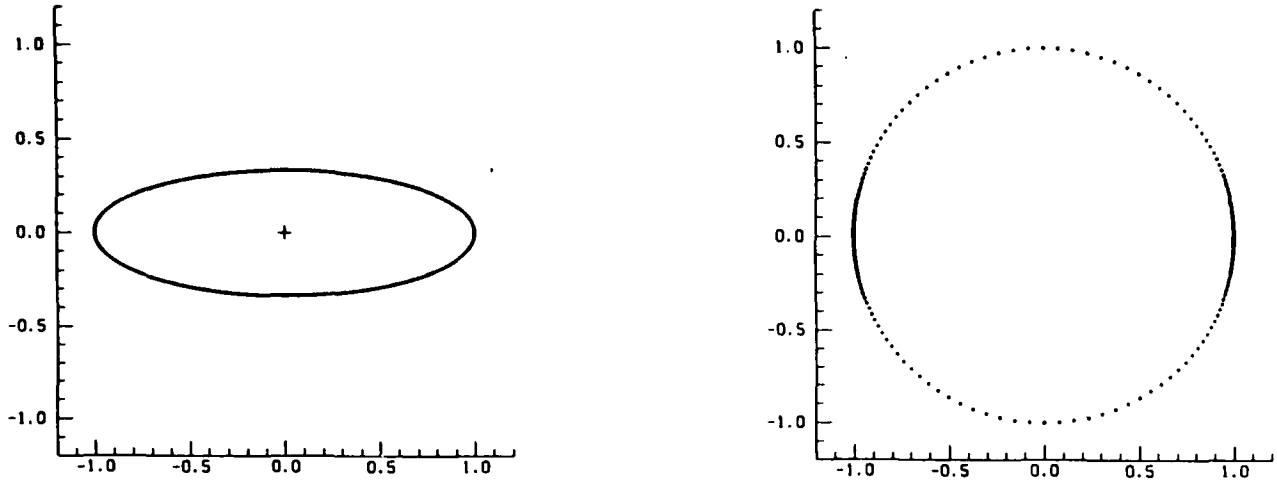


Figure 2e
A 3:1 Ellipse, n = 256.

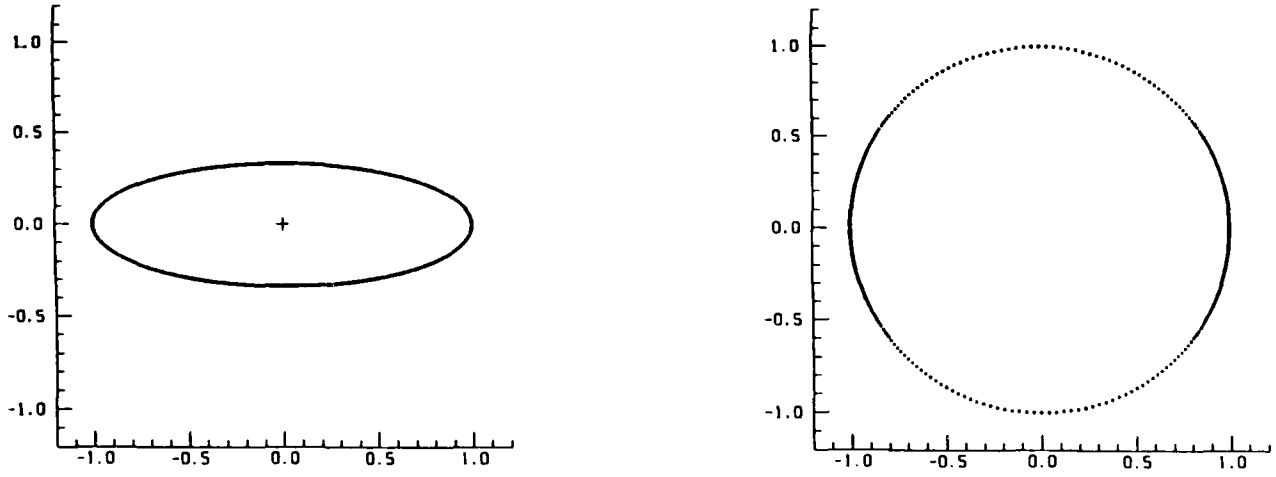


Figure 2f
A 3:1 Ellipse, n = 512

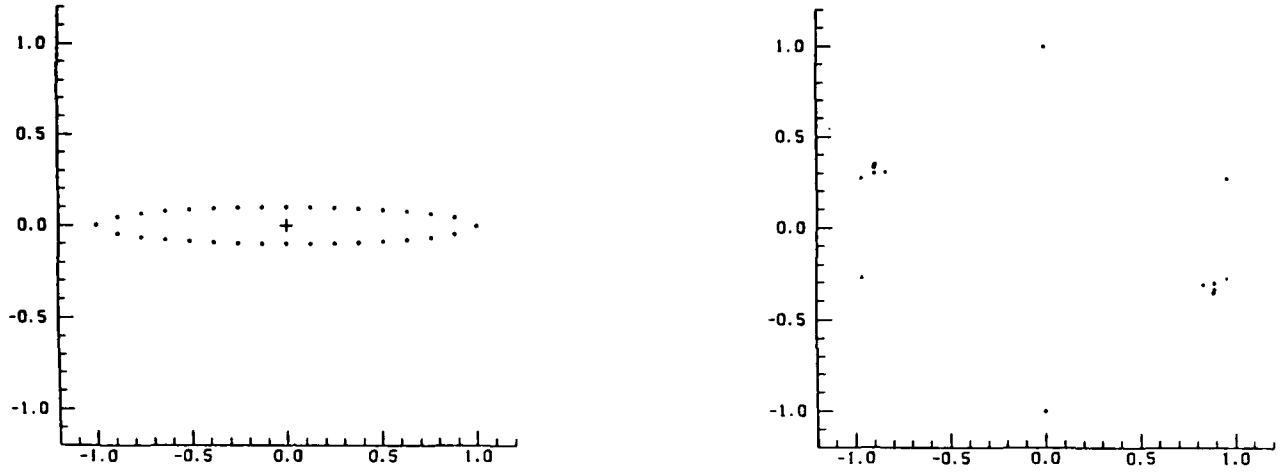


Figure 3a
A 10:1 Ellipse, n = 32.

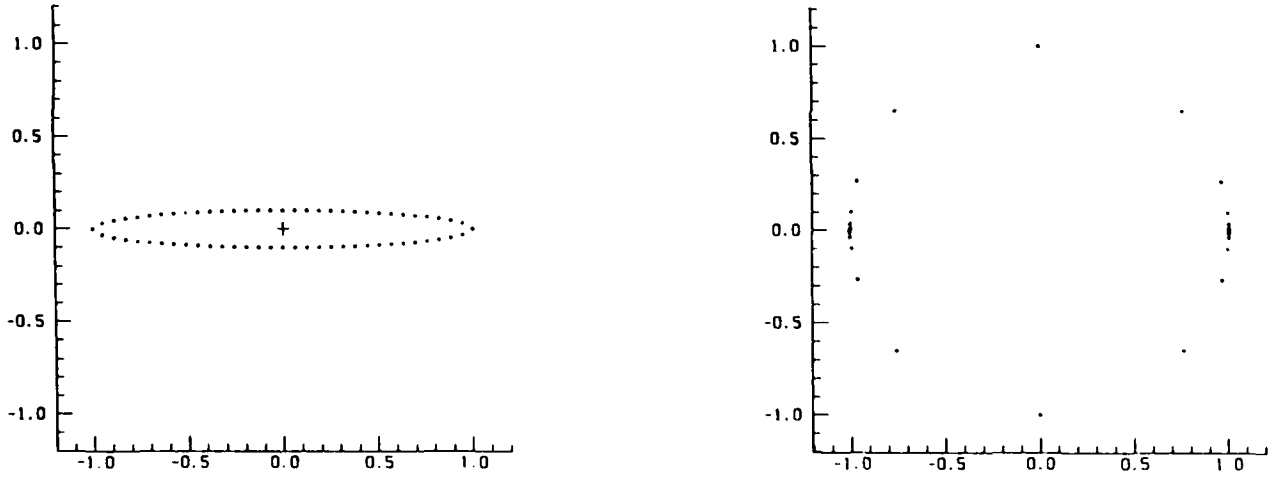


Figure 3b
A 10:1 Ellipse, n = 64.

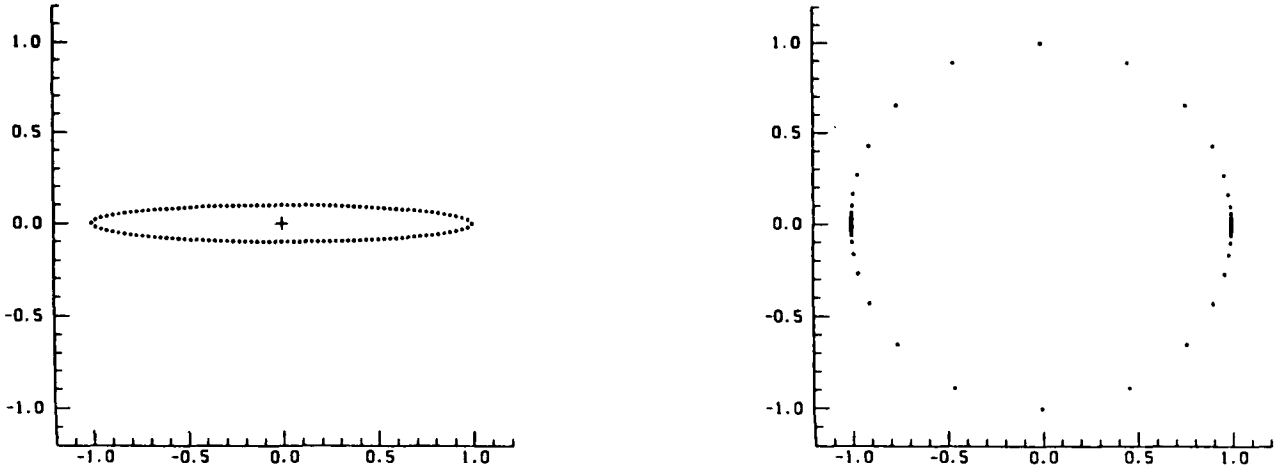


Figure 3c
A 10:1 Ellipse, n = 128.

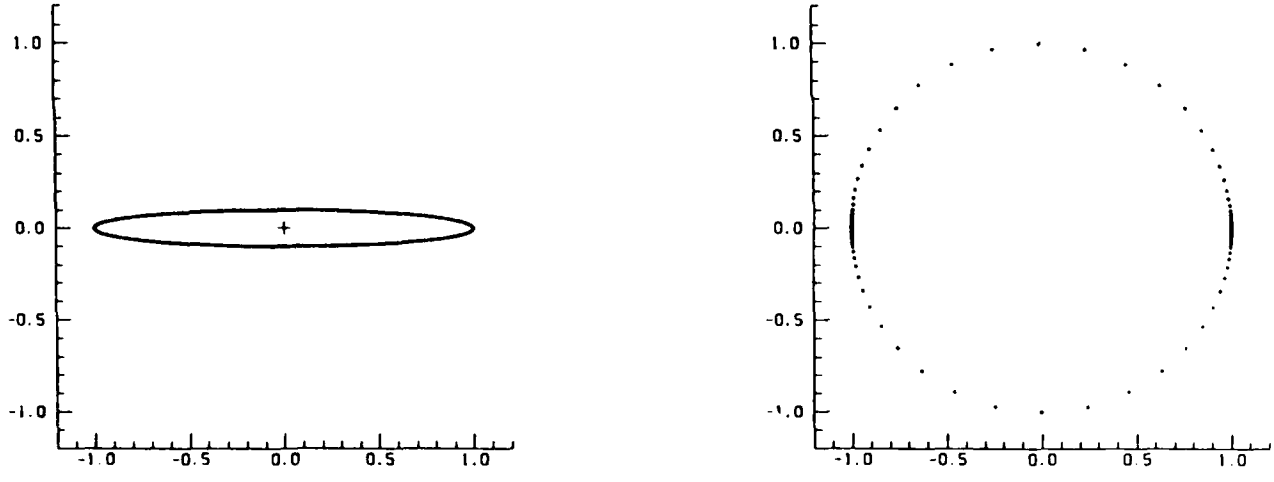


Figure 3d
A 10:1 Ellipse, n = 256.

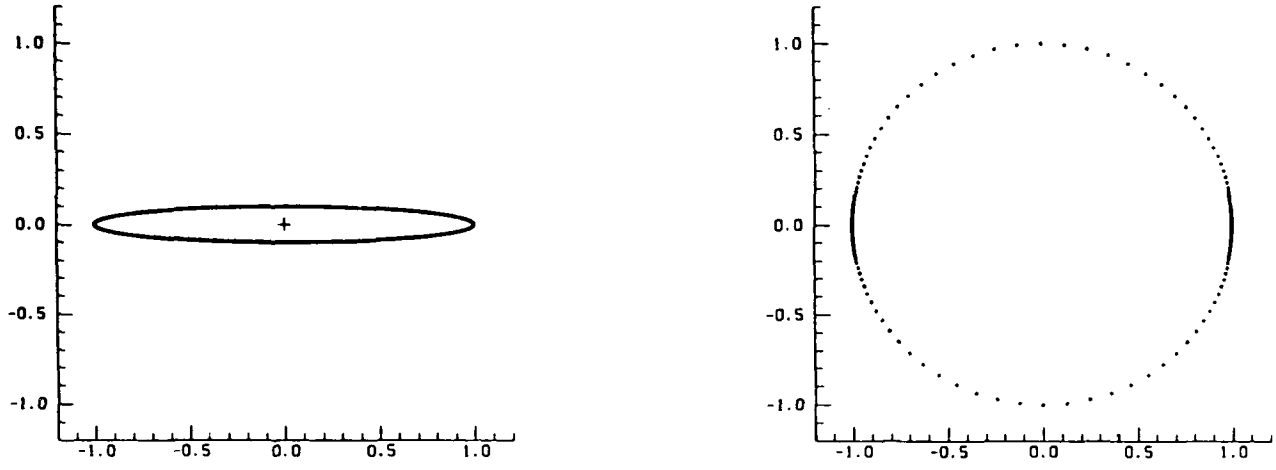


Figure 3e
A 10:1 Ellipse, n = 512.

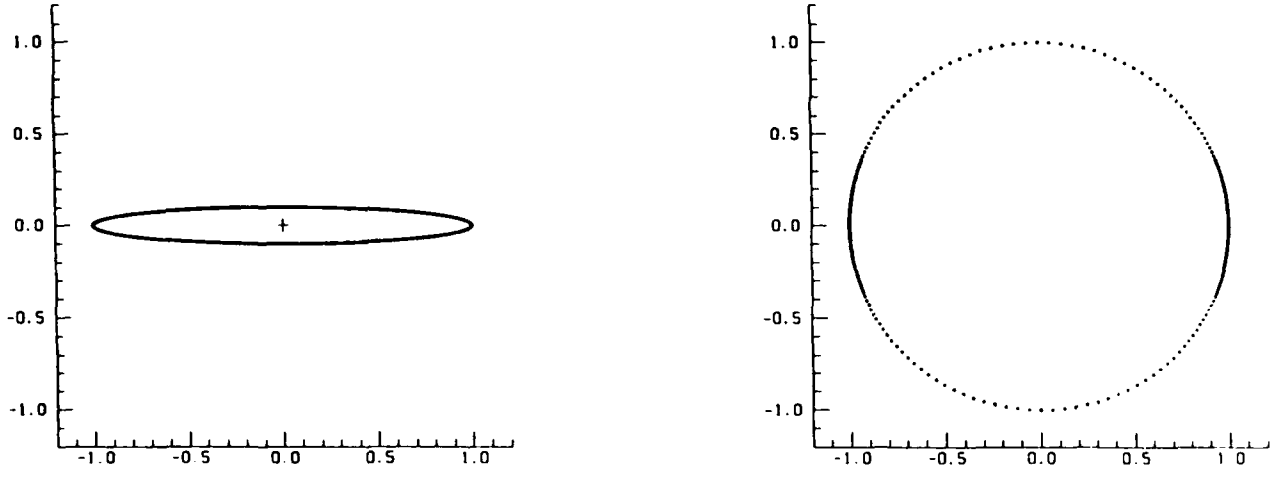


Figure 3f
A 10:1 Ellipse, n = 1024.

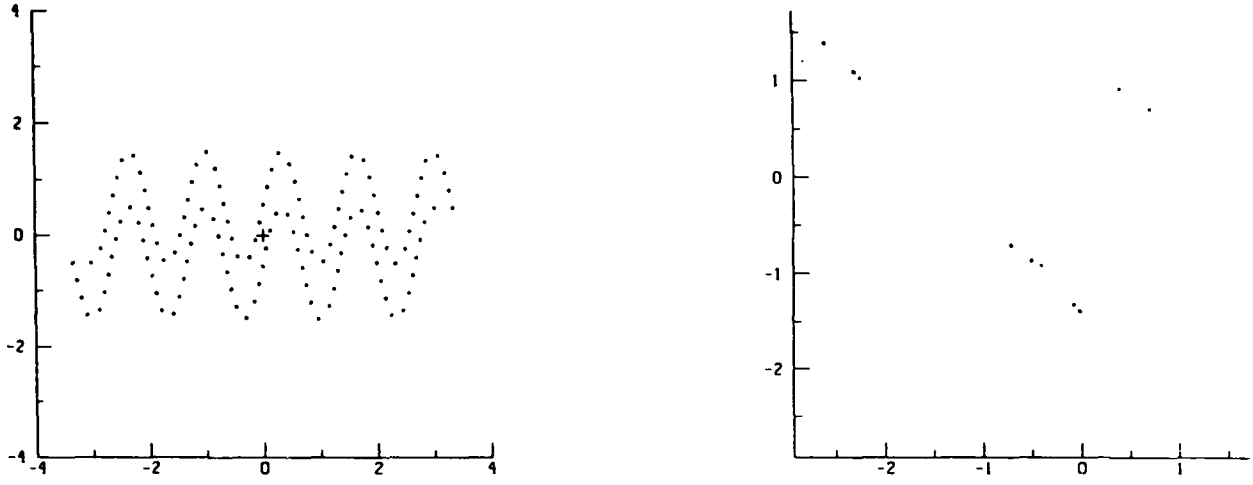


Figure 4a
A Snake-shaped Region, n = 128.

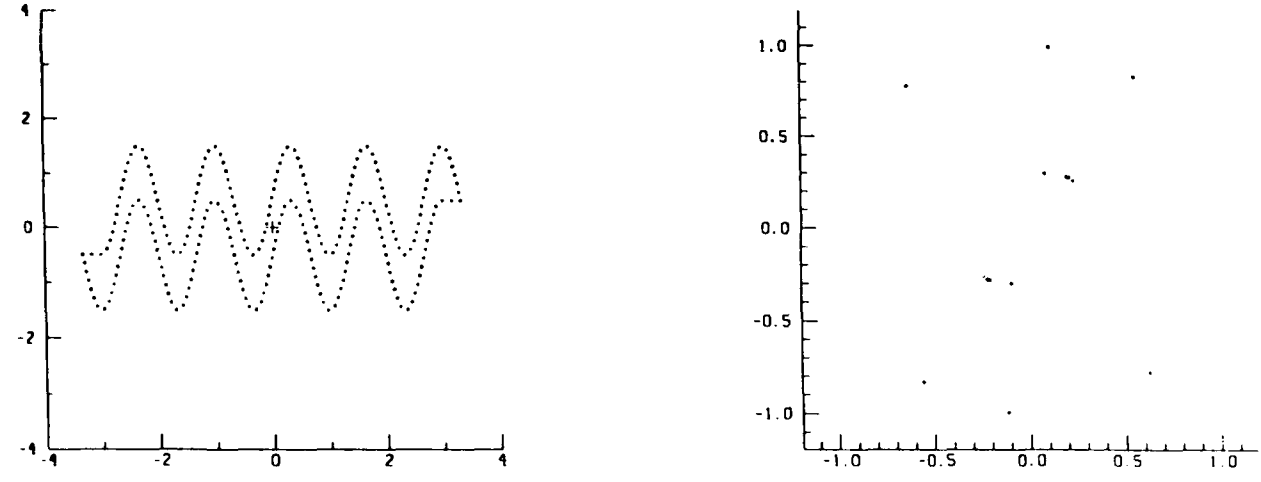


Figure 4b
A Snake-shaped Region, n = 256.

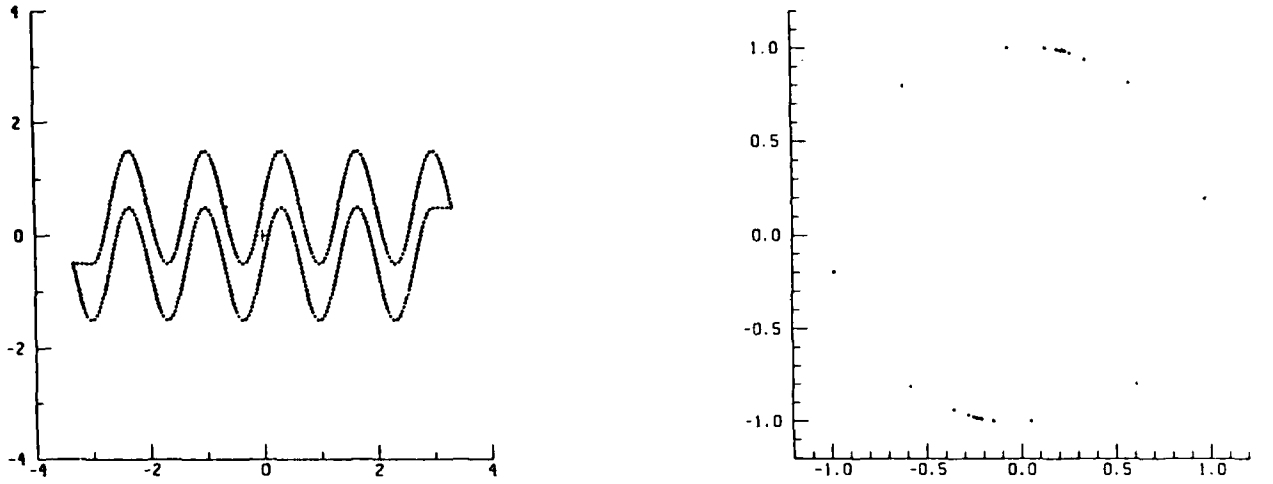


Figure 4c
A SnaKe-shaped Region, n = 512.

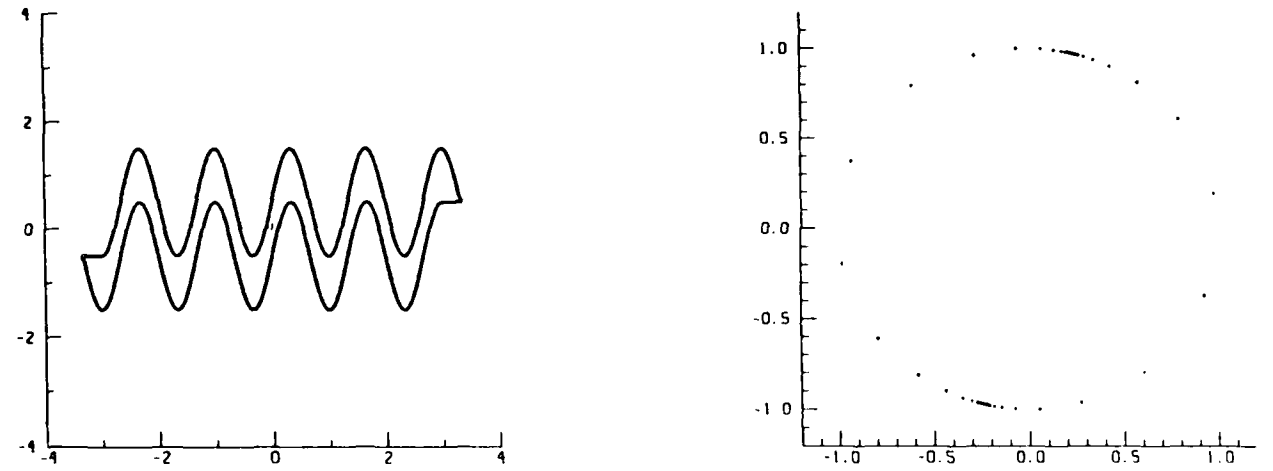


Figure 4d
A SnaKe-shaped Region, n = 1024.

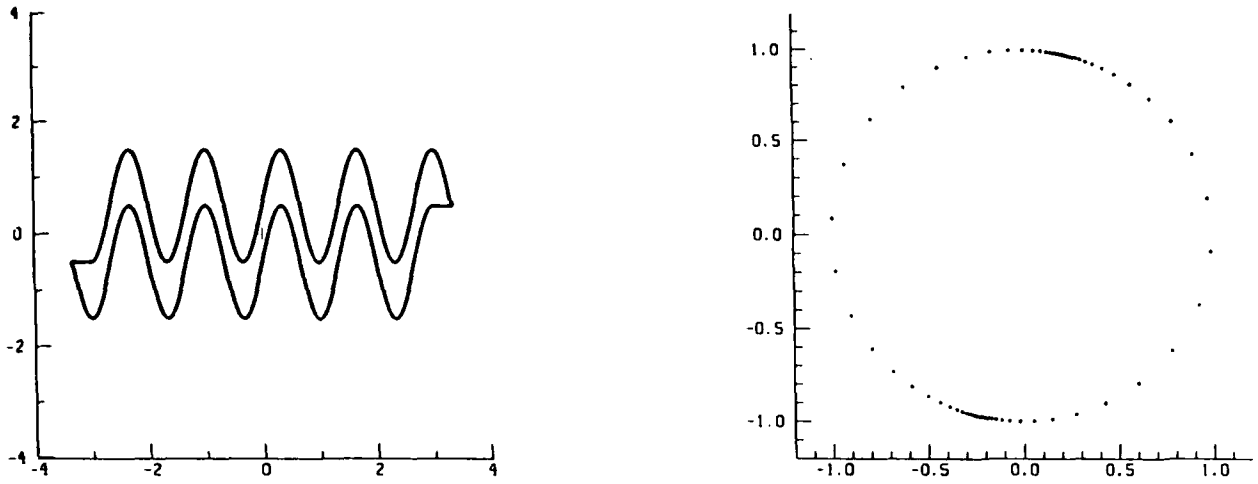


Figure 4e
A Snake-shaped Region, n = 2048.

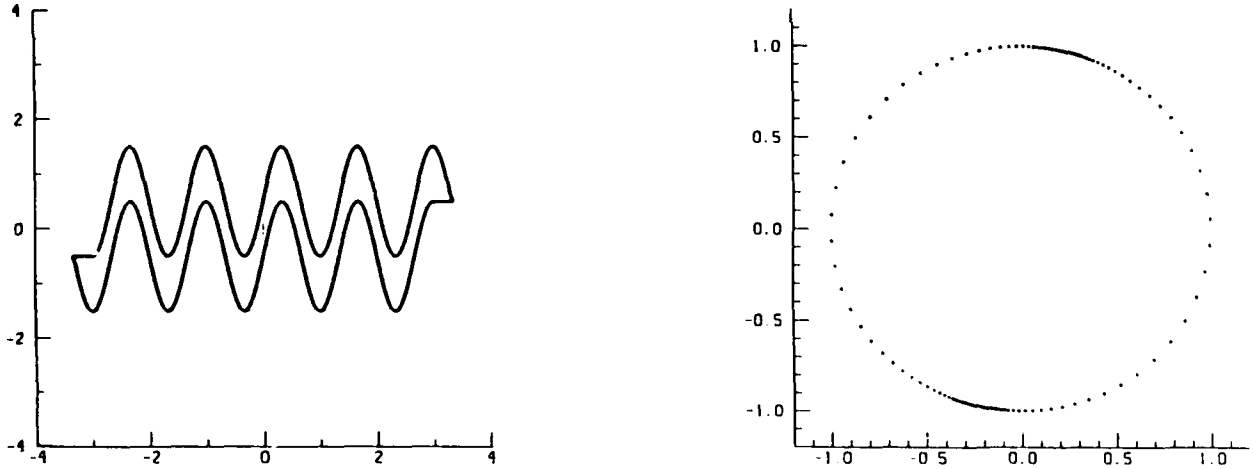


Figure 4f
A Snake-shaped Region, n = 4096.

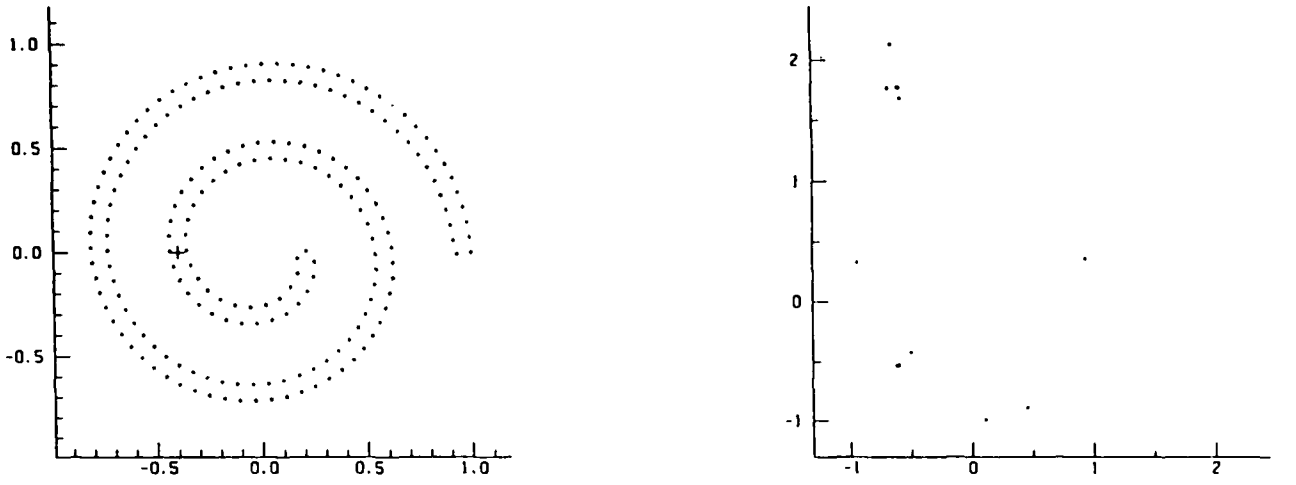


Figure 5a
A Spiral-shaped Region, n = 200.

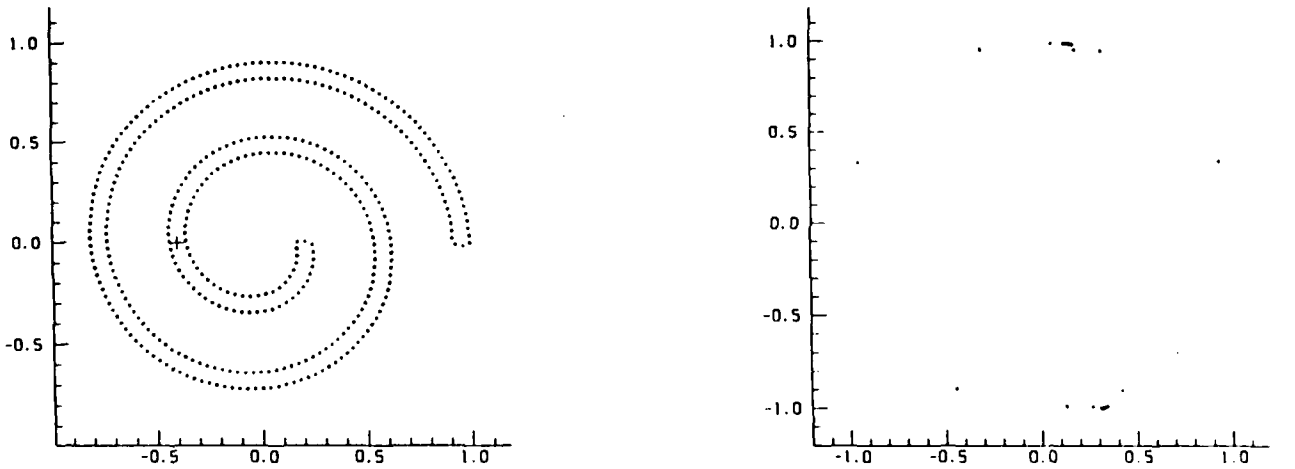


Figure 5b
A Spiral-shaped Region, n = 400.

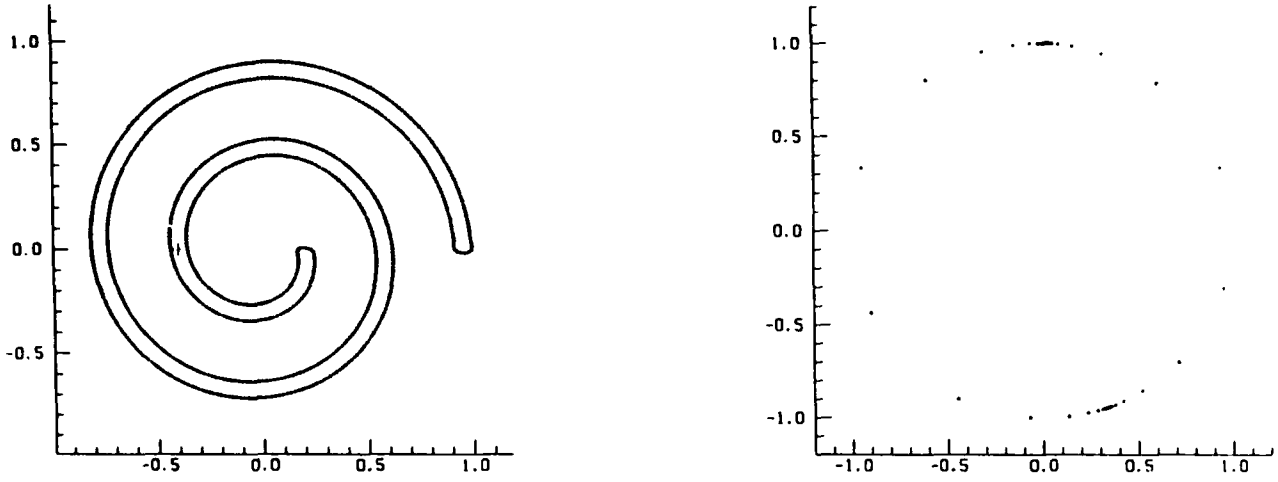


Figure 5c
A Spiral-shaped Region, n = 800.

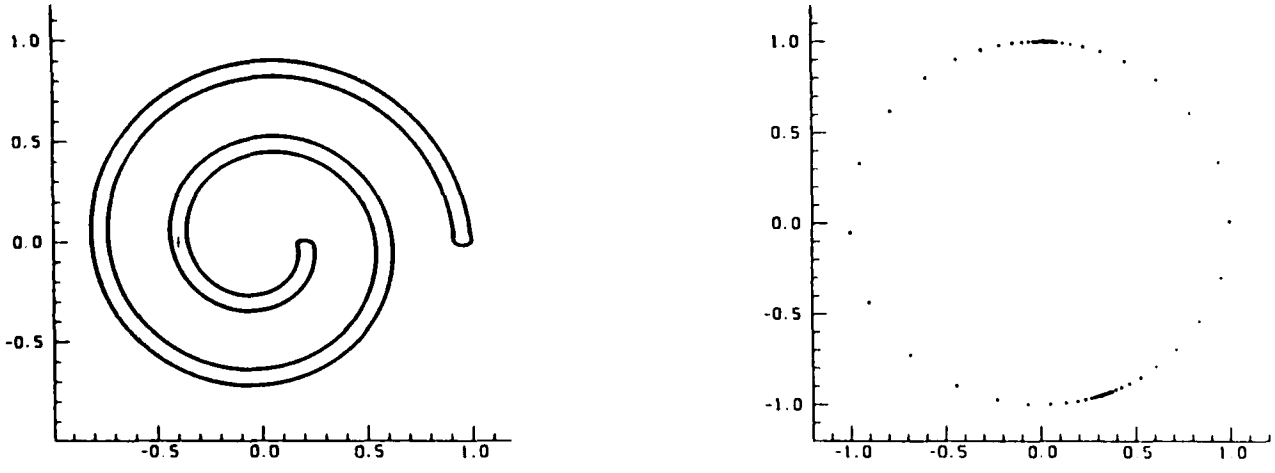


Figure 5d
A Spiral-shaped Region, n = 1600.

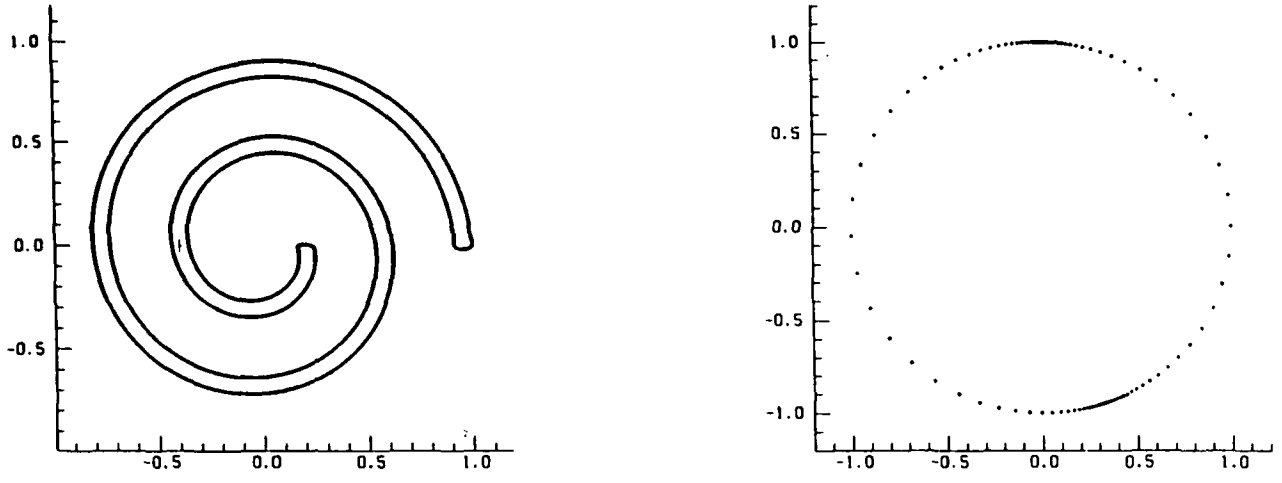


Figure 5e
A Spiral-shaped Region, n = 3200.

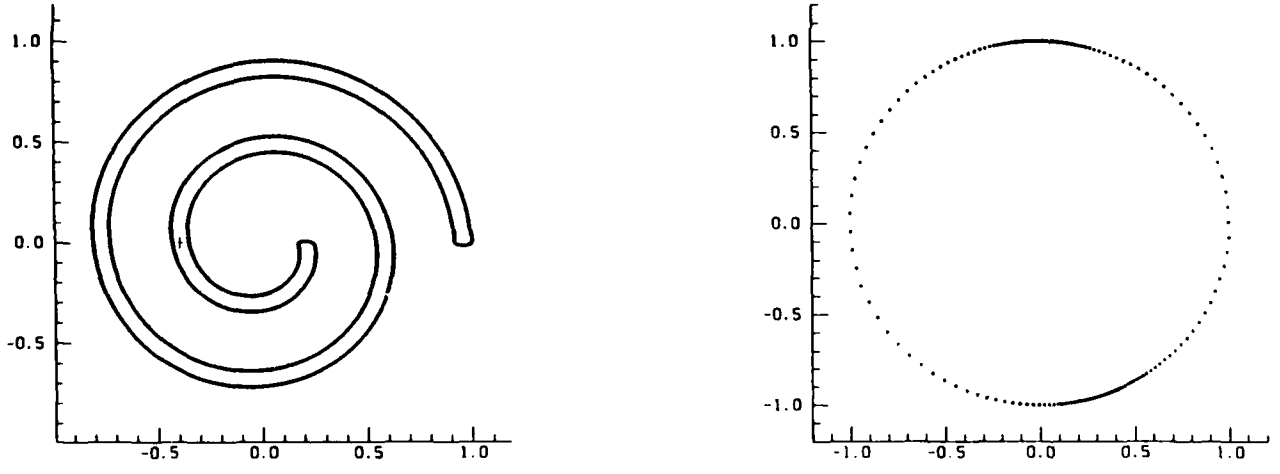


Figure 5f
A Spiral-shaped Region, n = 6400.

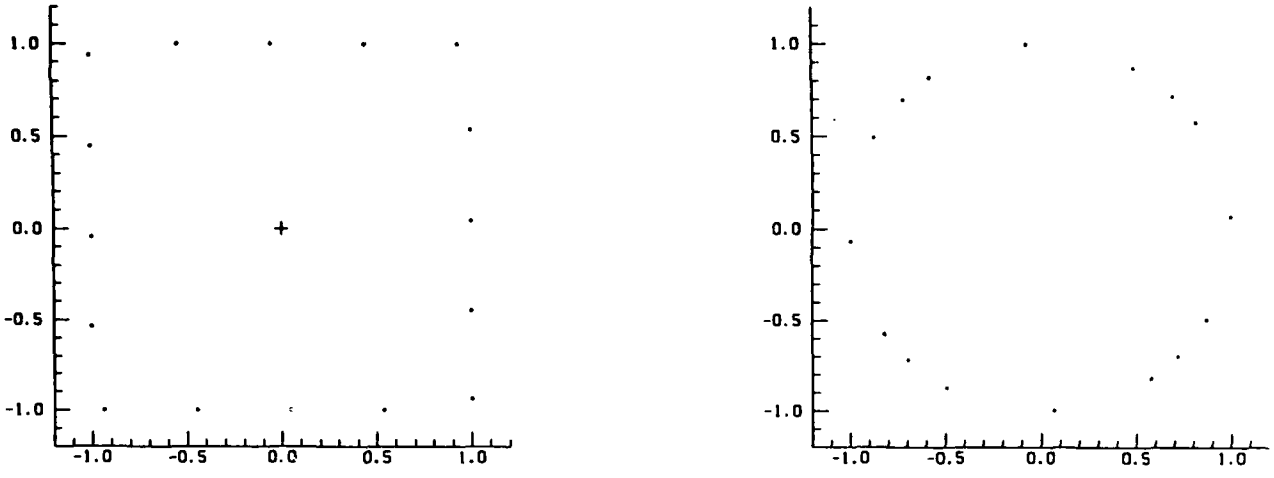


Figure 6a
A Rounded Square, n = 16.

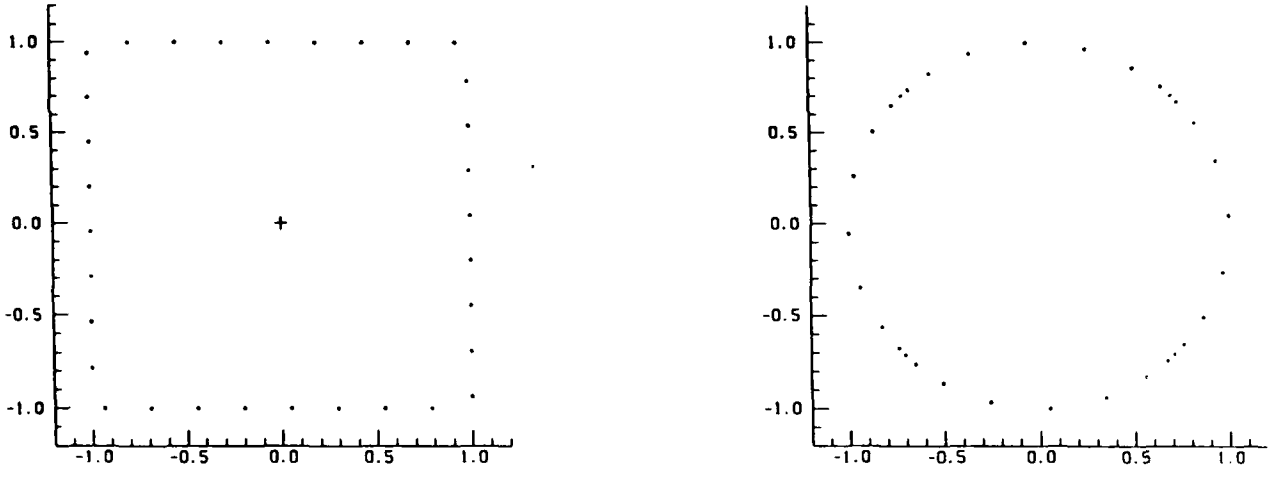


Figure 6b
A Rounded Square, n = 32.

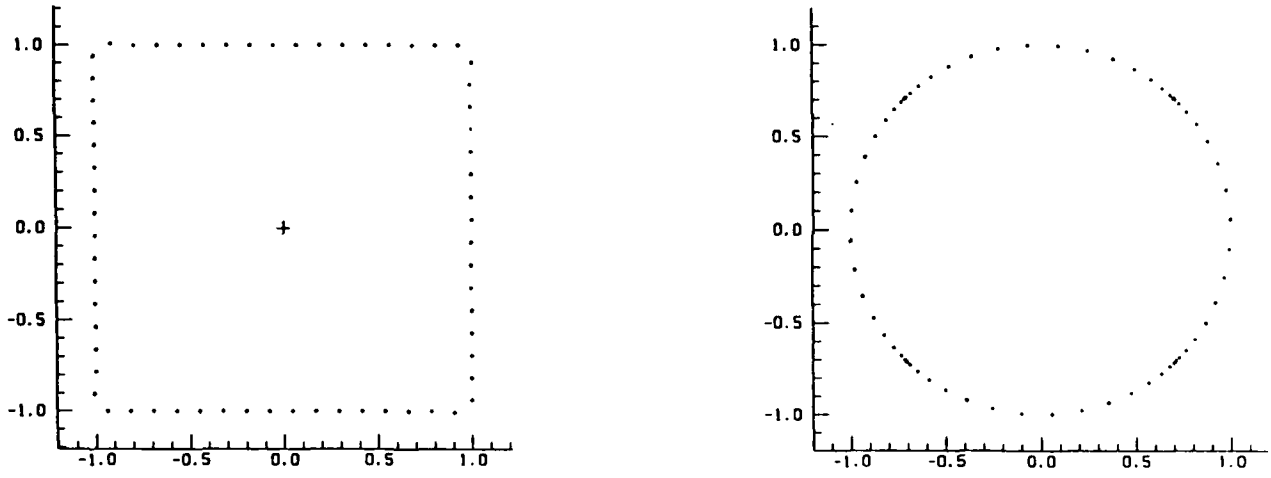


Figure 6c
A Rounded Square, n = 64.

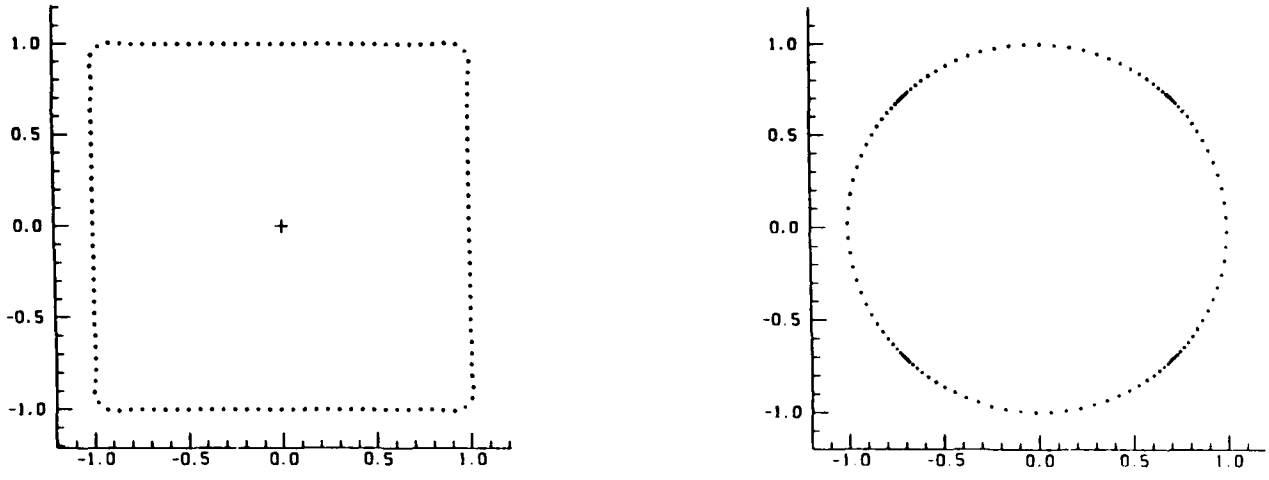


Figure 6d
A Rounded Square, n = 128.

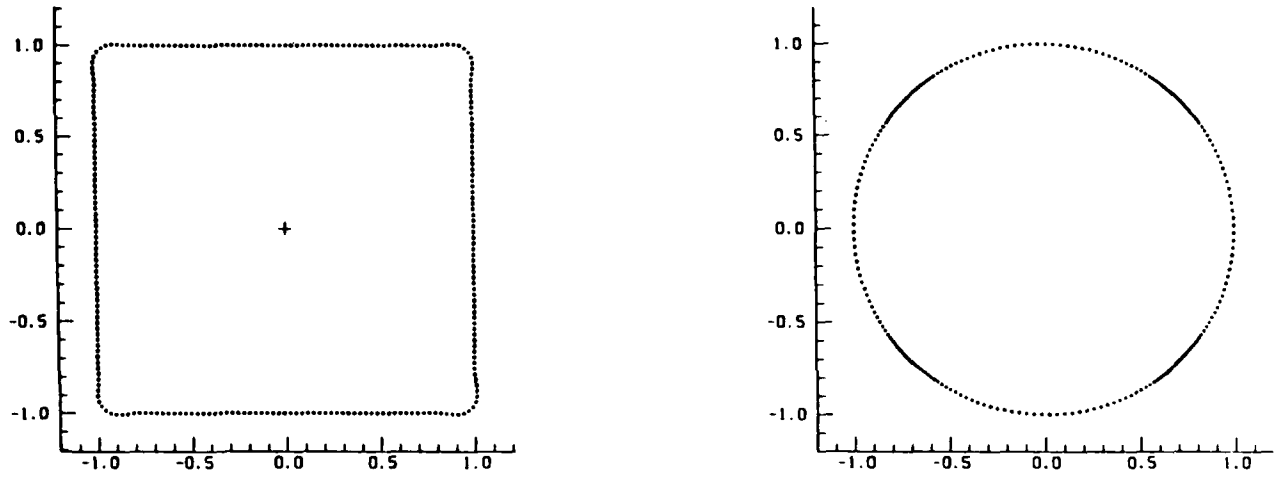


Figure 6e
A Rounded Square, n = 256.

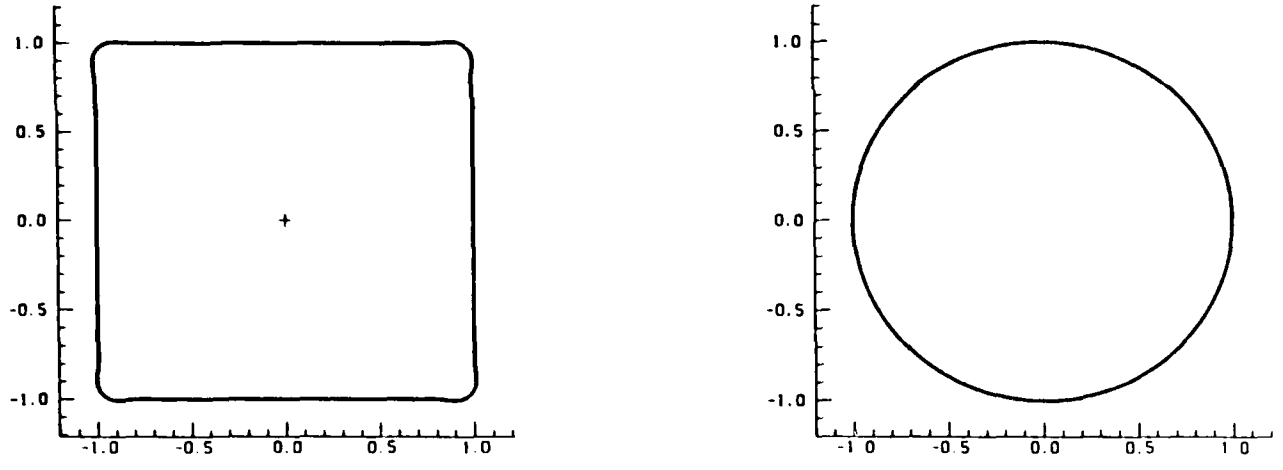


Figure 6f
A Rounded Square, n = 512.

END

DATE

FILMED

DTIC

JULY 88