UCLA
COMPUTATIONAL AND APPLIED MATHEMATICS

Interpolation on Arbitrary Regions in the Complex Plane

J. Ma

December 1994

CAM Report 94-43

Department of Mathematics
University of California, Los Angeles
Los Angeles, CA. 90024-1555

A numerical algorithm is presented for the construction of stable interpolation points on arbitrary Jordan curves in the complex plane, originally introduced by Walsh over six decades ago. The classical results by Walsh describe the distribution of the interpolation points on any given curve, upon which the Lagrange interpolation is stable and converges rapidly for analytic functions within the region bounded by that curve. The algorithm of this paper is based on the solution of an integral equation, very similar to those of potential theory, by the combination of the Conjugate Gradient Algorithm and the Fast Multipole Method. The performance of the algorithm is illustrated with several numerical examples

Interpolation on Arbitrary Regions in the Complex Plane

J. Ma[†]
Department of Mathematics
University of California, Los Angeles
Los Angeles, CA 90024

[†] Research supported in part by DARPA under Contract F49620/91/C/0084, and in part by ONR under Grants N00014-89-J-1527 and N00014-92-J-1890.
Approved for public release: distribution is unlimited.
Keywords: *Interpolation, Distribution of interpolation points, Fast Algorithms.*

1 Introduction

The theory of interpolation in one dimension has been thoroughly studied and understood, and can be easily extended to two dimensions when domains are rectangular. In practice, interpolation in the complex plane has been normally performed on local grids, and can be a tedious procedure globally. It is often desirable to have simple and efficient interpolation algorithms when a function is to be interpolated from a curve in the plane to the region bounded by that curve, such as in the evaluation of electrostatic or magnetostatic field near a boundary, or the velocity field near a vortex sheet in fluid mechanics (see [2, 13, 16]).

In fact, there exist classical results by Walsh (see [17]), describing distributions of the interpolation points on curves, upon which the Lagrange interpolation is stable and converges rapidly. However, the results by Walsh do not provide a numerical procedure for the determination of the nodes of such interpolation formulae.

In this paper, we present a numerical algorithm for the construction of such interpolation points based on the solution of an integral equation, very similar to those of potential theory, by the combination of the Conjugate Gradient Algorithm and the Fast Multipole Method. The algorithm requires $O(N^{\frac{3}{2}})$ operations asymptotically, where N is the number of nodes in the discretization of the given curve.

The paper is organized as follows. In Section 2, we restate the relevant mathematical results from [17], and in Section 3, we summarize the numerical techniques to be used in this paper. The actual numerical algorithm is presented in Section 4, and the performance of the algorithm is demonstrated with numerical examples in Section 5.

2 Mathematical Preliminaries

In this section, we summarize several classical results from [17] to be used in the rest of this paper. Throughout the paper, we will assume that Γ is a Jordan curve in the complex plane $\mathbb{C}$, and is parameterized by its length $\gamma : [0, L] \to R^2$. We will also assume that $\Omega \subset \mathbb{C}$ is bounded by Γ, and D is the complement of Ω so that $D = \mathbb{C} \backslash \Omega$.

2.1 Green's Function

For a fixed $w \in D$, a function $G(z, w) : D \to R^1$ is called the Green's function of D with the pole at w if it satisfies the following three conditions.

1. The function $G(z, w)$ is harmonic in $D \backslash w$

2. For any $z \in \Gamma$,

$$G(z, w) = 0. \tag{1}$$

3. The function $g : D \to R^1$ defined by the formula

$$g(z) = \begin{cases} G(z,w) - \log|z-w| & \text{for } w \neq \infty, \\ G(z,\infty) - \log|z| & \text{for } w = \infty, \end{cases} \tag{2}$$

is harmonic in a neighborhood of w.

For convenience, the Green's function with the pole at infinity $G(z, \infty)$ will be denoted either by $G(z)$ or $G(x, y)$ with $z = (x, y)$.

Lemmas 2.1 and 2.2 below are well-known and their proofs can be found in [17].

Lemma 2.1 *Suppose that G is the Green's function for the domain D with the pole at infinity, and n is the interior normal derivative of D. Then*

$$\frac{\partial G}{\partial n}(x, y) > 0 \tag{3}$$

for any $(x, y) \in \Gamma$.

Consequently, the function $u : [0, L] \to R^1$ defined by the formula

$$u(t) = \int_0^t \frac{\partial G}{\partial n} ds \tag{4}$$

is a monotonically increasing positive function on Γ.

Lemma 2.2 *Suppose that G is the Green's function for the domain D with the pole at infinity, and n is the interior normal derivative of D. Suppose further that (x_0, y_0) is an interior point in Ω, and for any $(x, y) \in D$,*

$$r = \sqrt{(x - x_0)^2 + (y - y_0)^2}. \tag{5}$$

Then

$$\begin{aligned} G(x,y) &= \log r - C + O(\frac{1}{r}) && \text{for } (x,y) \to \infty, && (6) \\ \frac{\partial G}{\partial n}(x,y) &= \frac{1}{r} + O(\frac{1}{r^2}) && \text{for } (x,y) \to \infty, && (7) \end{aligned}$$

where C is a constant.

2.2 Approximation by Polynomials

Theorem 2.1 below states that any function analytic in $\overline{\Omega}$ can be well approximated by polynomials, and the approximation possesses the so-called maximum convergence (see below). The proof of the theorem can be found in [17].

Theorem 2.1 *Suppose that the function $\Phi : D \to \mathbb{C}$ maps the domain D conformally onto the exterior of the unit circle so that $\Phi(\infty) = \infty$. Suppose also that for any positive real $r > 0$, the locus C_r is defined by*

$$C_r = \{z \in \mathbb{C} \textit{ such that } |\Phi(z)| = r\}. \tag{8}$$

Then for any analytic function $f : \overline{\Omega} \to \mathbb{C}$, there exists a real number $1 < \rho \leq \infty$ such that f is analytic at every point interior to the locus C_ρ defined by (8). For any real $R < \rho$, there exists a sequence of polynomials $\{p_n\}$ with $n = 0, 1, 2, \cdots$ such that for each n, the polynomial p_n is of degree n, and for all $z \in \overline{\Omega}$,

$$|f(z) - p_n(z)| \leq \frac{M}{R^n} \tag{9}$$

where M is a constant dependent of R and independent of n and z.

Furthermore, for any $R > \rho$, there exist no polynomials p_n such that (9) is valid for all $z \in \overline{\Omega}$.

The sequence of functions p_n satisfying (9) is said to converge maximally to function f on domain Ω.

2.3 Interpolation by Polynomials

It is well-known that for a given set of distinct points $z_0, z_1, \cdots, z_n$ in $\mathbb{C}$ and a set of values $f_0, f_1, \cdots, f_n$, there exists a unique polynomial p_n of degree n such that for all $i = 0, 1, \cdots, n$,

$$p_n(z_i) = f_i. \tag{10}$$

The interpolating polynomial p_n is given by the Lagrange interpolation formula

$$p_n(z) = \sum_{i=0}^{n} f_i \cdot L_i(z) \tag{11}$$

where L_i is the $i-$th Lagrange polynomial defined by the formula

$$L_i(z) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{z - z_j}{z_i - z_j} \tag{12}$$

with $i = 0, 1, \cdots, n$.

It is clear from Theorem 2.1 in Section 2.2 that any function f analytic in $\overline{\Omega}$ can be approximated by polynomials. In fact, f can be approximated by interpolating polynomials as well, and the distribution of interpolation points plays a crucial role in the convergence of the approximation (see Theorem 2.2 below).

We will first introduce the concept of uniform distribution of points. Suppose that $z_0, z_1, z_2, \cdots, z_n \in \Gamma$, and function $u : [0, L] \to R^1$ is positive and non-decreasing. Then the set of points z_i is called uniformly distributed on Γ with respect to u if the set of points z_i divides Γ into $(n+1)$ equal parts with respect to the increments of u.

Theorem 2.2 below is the principal analytical tool of this paper. It describes the distribution of stable interpolation points on any given curve in the complex plane. Theorem 2.3 states that uniformly distributed points on a given curve can be obtained via a conformal mapping. The proofs of Theorems 2.2 and 2.3 can be found in [17].

Theorem 2.2 *(Walsh [17], 1935) Suppose that f is a function analytic in $\overline{\Omega}$. Suppose also that $z_0^{(n)}, z_1^{(n)}, \cdots, z_n^{(n)}$ are $(n+1)$ points on Γ, and p_n denotes the interpolating polynomial of degree n to the function f on the $(n+1)$ points $\{z_i^{(n)}\}$.*

Then the sequence of the interpolating polynomials p_n converges uniformly to f on domain $\overline{\Omega}$ if and only if the set of interpolation points $\{z_i^{(n)}\}$ is uniformly distributed on Γ with respect to the function u defined by (4). Furthermore, if the sequence p_n converges to f uniformly, then it converges maximally.

The well-known Fekete points $z_0, z_1, \cdots, z_n$, which maximize the modulus of the Vandermonde determinant

$$V_n(w_0, w_1, \cdots, w_n) = \prod_{\substack{i,j=0 \\ i<j}}^{n} (w_i - w_j) \tag{13}$$

are uniformly distributed with respect to (4). Moreover,

$$|L_i(z)| \leq 1 \tag{14}$$

for any $z \in \Omega$, and $i = 0, 1, 2, \cdots, n$ (see [17], for example).

Theorem 2.3 *Suppose that Φ maps the domain D conformally onto the exterior of the unit circle so that $\Phi(\infty) = \infty$. Suppose also that $w_0, w_1, \cdots, w_n$ are equally spaced on the unit circle with respect to its arc length. Then points $z_i = \Phi^{-1}(w_i)$ for $i = 0, 1, 2, \cdots, n$ are uniformly distributed on the boundary Γ of domain D with respect to the function u defined by (4).*

2.4 An Integral Equation

In this subsection, we establish an integral equation for the computation of the normal derivative of the Green's function of the domain D on the boundary Γ, which in turn will enable us to find uniformly distributed interpolation points on Γ.

Theorem 2.4 *Suppose that G is the Green's function for the domain D with the pole at infinity, and n is the interior normal derivative of D. Then there exists a constant C such that*

$$\frac{1}{2\pi}\int_0^L \log|z-w|\sigma(w)ds_w = C, \tag{15}$$

and

$$\frac{1}{2\pi}\int_0^L \sigma(w)ds_w = 1, \tag{16}$$

for any $z \in \Gamma$, where ds is the arc length on Γ, and

$$\sigma = \frac{\partial G}{\partial n}. \tag{17}$$

Proof: Suppose that (x_0, y_0) is an interior point of Ω, and for $(x, y) \in D$,

$$r = \sqrt{(x-x_0)^2 + (y-y_0)^2}. \tag{18}$$

Suppose also that $C(R)$ is a circle of radius R with center (x_0, y_0) such that $\Omega \subset C(R)$, and E denotes the region bounded by both Γ and $C(R)$.

By the assumption that G is the Green's function for the domain D with the pole at infinity, we immediately have

$$G(x,y) = 0 \qquad \text{for all } (x,y) \in \Gamma. \tag{19}$$

Consequently, we have

$$\int_\Gamma \log r\frac{\partial G}{\partial n}ds = \int_\Gamma \left(\log r\frac{\partial G}{\partial n} - G\frac{\partial \log r}{\partial n}\right) ds. \tag{20}$$

Observing that functions $\log r$ and G are harmonic in the domain D, we apply the Green's formula (see [4], for example)

$$\int_E (u\Delta v - v\Delta u)\,dxdy = \int_{\partial E}\left(u\frac{\partial v}{\partial n} - v\frac{\partial u}{\partial n}\right) ds \tag{21}$$

to functions $\log r$ and G on the region $E \subset D$, and obtain

$$\int_\Gamma \left(\log r\frac{\partial G}{\partial n} - G\frac{\partial \log r}{\partial n}\right) ds = \int_{C(R)} \left(\log r\frac{\partial G}{\partial n} - G\frac{\partial \log r}{\partial n}\right) ds. \tag{22}$$

Combining (20) and (22), we get

$$\int_{\Gamma} \log r \frac{\partial G}{\partial n} ds = \int_{C(R)} \left(\log r \frac{\partial G}{\partial n} - G \frac{\partial \log r}{\partial n} \right) ds. \tag{23}$$

Combining (23) with (6) and (7) in Lemma 2.2, and letting $R \to \infty$, we obtain

$$\int_{\Gamma} \log r \frac{\partial G}{\partial n} ds = 2\pi C. \tag{24}$$

That is,

$$\frac{1}{2\pi} \int_{\Gamma} \log |z_0 - w| \frac{\partial G}{\partial n}(w) ds_w = C \tag{25}$$

for any $z_0 = (x_0, y_0) \in \Omega \backslash \Gamma$. Now, (15) follows from (25) and the fact that the logarithmic potential is continuous in R^2.

The proof of (16) is nearly identical to that of (15). The only difference is that the Green's formula (21) is applied to functions G and $\varphi \equiv 1$ rather than functions G and $\log r$. ■

Remark 2.1 *Due to Theorem 2.4, the function u defined by (4) assumes the form*

$$u(t) = \int_0^t \sigma(w) ds \tag{26}$$

where σ is the solution of the integral equation (15) satisfying the condition (16).

It is well-known that the integral equation (15) has a unique solution satisfying condition (16) (see [7], for example). For a given right-hand side C, the equation (15) has a unique solution provided that the capacity of Γ is not equal to 1 (see [7], for example). If the capacity of Γ equals to 1, the problem is easily remedied by simple scaling.

Remark 2.2 *Obviously, the right-hand side C of equation (15) can be chosen to be equal to 1, since the location of the uniformly distributed points on Γ is independent of C.*

3 Numerical Preliminaries

In this section, we describe numerical techniques to be used to solve the integral equation (15). The Nyström algorithm (or quadrature method) is used to discretize the integral equation, and the Conjugate Gradient Method is used to solve the resulting linear system. At each iteration of the Conjugate Gradient Algorithm, the Fast Multipole Method (see [5, 3]) is used to compute the matrix-vector product in $O(N)$ operations.

3.1 Corrected Trapezoidal Rules For Singular Periodic Functions

An n-point quadrature rule Q_n on interval $[a, b]$ is defined by the formula

$$Q_n(f) = \sum_{i=1}^{n} w_i f(x_i) \tag{27}$$

where $f : [a, b] \to R^1$ is an integrable function, and $\{x_i\} \in [a, b]$ and $\{w_i\} \in R^1$ are the quadrature nodes and weights respectively. A family of quadrature formulae $\{Q_n\}$ has a rate of convergence p $(p \geq 1)$ if there exist a constant $M > 0$ and an integer $N > 0$ such that for all $n > N$,

$$\left| Q_n(f) - \int_a^b f(x)dx \right| < \frac{M}{n^p}. \tag{28}$$

The quadrature formula (27) will be referred to as the n-point Trapezoidal quadrature rule if the quadrature nodes and weights are chosen so that for $i = 1, 2, \cdots, n$,

$$x_i = a + (i-1)\frac{b-a}{n}, \tag{29}$$

$$w_1 = w_n = \frac{b-a}{2n}, \tag{30}$$

and for $i = 2, \cdots, n-1$,

$$w_i = \frac{b-a}{n}. \tag{31}$$

The n-point trapezoidal quadrature rule will be denoted by T_n.

It is well-known that for a periodic function with p continuous derivatives, the trapezoidal rule T_n has an order of convergence $p + 1$. For singular functions, more sophisticated quadratures have to be used (see [14, 8], for example). Recently, high order corrected Trapezoidal rules have been introduced in [8] for singular functions. We will need the quadrature rules of [8] in a very special case, and below is a description of one such rule. We refer the reader to [8] for both the detailed theory of such rules, and for the description of many other situations where such quadrature rules are applicable.

Suppose that Γ is a Jordan curve parameterized by its length $\gamma : [0, L] \to R^2$. For notational convenience, we will extend the mapping γ into a periodic mapping $R^1 \to R^2$, by postulating that

$$\gamma(x + jL) = \gamma(x) \tag{32}$$

for any integer j and $x \in [0, L]$. Suppose further that for some integer n, the curve γ is discretized into n equispaced nodes, that is, points $x_i \in R^1$ are specified by the formulae

$$x_i = (i-1)h \tag{33}$$

with $h = L/n$, and $i = 0, \pm 1, \pm 2, \cdots$.

Suppose that function $g : \Gamma \to R^1$ is continuous and function $\phi : [0, L] \to R^1$ is defined by

$$\phi(x) = g(\gamma(x)) \tag{34}$$

for any $x \in [0, L]$. For notational convenience, we will also extend the function ϕ into a periodic function $R^1 \to R^1$ such that

$$\phi(x + jL) = \phi(x) \tag{35}$$

for any integer j and $x \in [0, L]$.

We will be considering integrals of the form

$$J(x_i) = \int_0^L \log|x_i - x|\,\phi(x)ds, \tag{36}$$

for all $i = 0, \pm 1, \pm 2, \cdots$, where ds is the arclength on Γ. The integrals (36) will be approximated by quadrature rules of the form

$$J(x_i) \approx T_n(f_i) + h\sum_{j=1}^{p} \alpha_j \left(\phi(x_{i+j}) + \phi(x_{-i+j})\right) + h\left(\log h + 2c_0 - 2\sum_{j=1}^{p} \alpha_j\right)\phi(x_i), \tag{37}$$

with appropriately chosen c_0 and α_j, where the function f_i is defined by the formula

$$f_i(x) = \log|x_i - x| \cdot \phi(x). \tag{38}$$

The following theorem can be found in [8].

Theorem 3.1 *Suppose that Γ is a Jordan curve parameterized by its length $\gamma : [0, L] \to R^2$, and function $\phi : [0, L] \to R^1$ is a function defined on the curve Γ and possesses at least $2p$ continuous derivatives. Then there exist coefficients α_1, α_2, $\cdots$, α_p, and a constant c_0 such that the quadrature rule (37) has at least $2p$ order of convergence. Furthermore,*

$$\sum_{k=1}^{p} \alpha_j^2 \le 1 \tag{39}$$

for all integer $p \ge 1$.

Remark 3.1 *In Theorem 3.1, the constant c_0 is independent of the order of corrections p, and for double precision, it assumes the following value*

$$c_0 \equiv -0.918938533204673. \tag{40}$$

We refer the reader to [8] for the detailed calculation of c_0,

For $p = 2, 3, 4, 5, 10, 15$, the coefficients α_j of the quadrature rule (37) are given in Tables 10-15.

3.2 The Nyström Algorithm

Given an n-point quadrature formula Q_n with nodes $\{x_i\}$ and weights $\{w_i\}$, the Nyström algorithm discretizes the integral equation

$$\int_a^b K(x,t) \cdot \phi(t)dt = f(x) \tag{41}$$

as a system of linear algebraic equations

$$\sum_{i=1}^{n} w_j K(x_i, x_j)\phi(x_j) = f(x_i) \tag{42}$$

with $i = 1, 2, \cdots, n$. If (41) has a unique solution, then for a reasonable choice of quadrature formulae and sufficiently large n, the system of equation (42) also has a unique solution. Furthermore, under fairly broad assumptions, the convergence rate of the Nyström algorithm is one order lower than the convergence rate of the quadrature formula if the condition number of (42) is proportional to n.

In order to apply the Nyström algorithm to the integral equation (15), we decompose the left-hand side of (15) into two parts

$$\int_0^L \log|z-w|\sigma(w)ds_w = I(z) + J(z), \tag{43}$$

where the integral operators I and J are defined by the formulae

$$I(z) = \int_0^L \log\left|\frac{z-w}{\gamma^{-1}(z) - \gamma^{-1}(w)}\right| \sigma(w)ds_w, \tag{44}$$

$$J(z) = \int_0^L \log\left|\gamma^{-1}(z) - \gamma^{-1}(w)\right| \sigma(w)ds_w. \tag{45}$$

It is easily observed that the integrand in (44) is a c^k–function of w for any $z \in \Gamma$ if the curve $\gamma \in c^k$. Thus, the integral operator I in (44) will be discretized by the trapezoidal quadrature rule. On the other hand, the integral operator J in (45) will be discretized by the corrected Trapezoidal rule (37). The discretization of integral equation (15) results in a system of the linear equations of the form

$$(A_n + B_n)\psi = b, \tag{46}$$

where $b = (1, 1, \cdots, 1)^T$, and for $i, j = 1, 2, \cdots, n$

$$A_n(i,j) = h\log|z_i - z_j|, \tag{47}$$

$$B_n(i,j) = h\alpha_k \qquad \text{for } |i-j| = k, \tag{48}$$

$$B_n(i,j) = 0 \qquad \text{for } |i-j| \neq k, \tag{49}$$

with $k = 0, 1, \cdots, p$, and $h = L/n$.

Theorem 3.2 below states that the coefficient matrix in equation (46) is negative definite, and the condition number of the coefficient matrix is proportional to the size of the matrix. Its proof can be found in [8].

Theorem 3.2 *Suppose that the domain Ω is contained in a unit circle, and $K_n = A_n + B_n$ is the coefficient matrix of equation (46). Then there exists an integer $N_0 > 1$ such that for all $n > N_0$, the matrix K_n is negative definite. Furthermore, there exists a constant c such that*

$$\kappa(K_n) = c \cdot n, \tag{50}$$

where $\kappa(K_n)$ is the condition number of K_n.

3.3 Application of Matrices A_n and B_n to Arbitrary Vectors

The Fast Multipole Method (see [3, 5]) provides an $O(n)$ scheme for the evaluation of gravitational and electrostatic potentials involving n particles. Hence, it can be used to apply the matrix A_n defined by (47) to any vector in $O(n)$ operations. On the other hand, the matrix B_n contains $(2p+1) \cdot n$ elements, and thus the application of B_n to any vector requires only $O(n)$ operations.

3.4 Conjugate Gradient Method

The conjugate gradient method (see [6, 15], for example) can be used as an iterative method for the solution of a system of linear equations of the type,

$$A\mathrm{x} = b \tag{51}$$

where $b \in R^n$, and the matrix $A \in R^{n \mathrm{x} n}$ is positive definite. Suppose that x_i is the approximate solution at the i–th iterate, then

$$||\mathrm{x} - \mathrm{x}_i||_A \leq 2||\mathrm{x} - \mathrm{x}_0||_A \left(\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1} \right)^i, \tag{52}$$

where $\kappa(A)$ is the condition number of the matrix A, and the A–norm of vector $\mathrm{y} \in R^n$ is defined by

$$||\mathrm{y}||_A = \sqrt{\mathrm{y}^T A \mathrm{y}}. \tag{53}$$

Combining (52) and Theorem 3.2, it is easy to observe that for any given precision ϵ, the conjugate gradient method requires $O(\sqrt{n})$ iterations to solve equation (46). On the other hand, each iteration can be achieved in $O(n)$ operations by the Fast Multipole

Method (see Section 3.3). Thus, the Conjugate Gradient algorithm combining with the Fast Multipole Method has an asymptotic time complexity $O(n^{\frac{3}{2}})$ for the solution of the integral equation (15).

4 Description of the Algorithm

Following is the description of the algorithm for the construction of uniformly distributed interpolation points on any given curve Γ.

1. Resample Γ into equispaced nodes with respect to the arc length by using periodic splines under tension as the basis for interpolation.
2. Apply the Conjugate Gradient Method to solve the linear system (46) resulted from the discretization of the integral equation (15). At each iteration of the algorithm,
   - apply the matrix A_n defined by (47) to the vector via the Fast Multipole Method,
   - apply the matrix B_n defined by (48) and (49) to the vector directly.
3. Find uniformly distributed points on Γ with respect to the function u of (26).
   - Integrate the solution of the linear system (46) to obtain u of (26).
   - Obtain the uniformly distributed points by applying the inverse interpolation technique.

Remark 4.1 *We use the 4-th order inverse interpolation to obtain the stable interpolation points after solving the linear system (46). For high precision, the inverse interpolation technique can be replaced by Newton's method.*

Remark 4.2 *Integral equations of the first kind are in general to be avoided whenever possible because of potential ill-conditioning.*

Although the condition number of the system (46) grows in proportion to its size n, it is safe numerically to use the first kind integral equation (15) in the construction of stable interpolation points due to the following two facts.

- *It can be easily observed that interpolation process is insensitive to small perturbations of stable interpolation points. Thus, it is often sufficient to obtain the solution of the integral equation (15) with low precision for the construction of stable interpolation points.*

- *Since (15) is a first kind integral equation, one would expect that the number of iterations required in the Conjugate Gradient Method would grow with the number of nodes N in the discretization. However, our numerical experiments in Section 5 indicate that the number of iterations required does not grows with N (see Tables 1-9). One obvious explanation is that the projection of the solution of (15) onto its high-frequency eigenfunctions is very small, so that equation (15) behaves almost like a second kind integral equation when the right-hand side of (15) is constant.*

Remark 4.3 *It would be natural to attempt to use Theorem 2.3 to find the uniformly distributed interpolation points via explicitly constructing the conformal mapping of the region D onto the exterior of the unit disk. A very satisfactory algorithm for the construction of such mappings can be found in [9, 12]. However, it is somewhat difficult to use this approach due to the so-called crowding phenomenon (see [12], for example). In our experiments, the approach of this paper turned out to be more stable than that based on the explicit construction of the conformal mapping.*

5 Numerical Results

We have implemented the numerical algorithm described in Section 4 for the construction of uniformly distributed interpolation points on a curve, and performed numerical experiments on a Sparc-2 workstation. Tables 1-9 contain results for several examples. The first column of each table contains the number of nodes in the discretization of the given curve. The second column contains the CPU time of the algorithm. The third column contains the number of iterations taken by the Conjugate Gradient process. The fourth column contains the L^2 norm of the residual of the numerical solution of (46), that is, $||(A_n+B_n)\psi-b||_2$. The fifth and sixth columns contain the absolute and relative errors of the solution in L^2 norm respectively. For each example, we also present a set of figures depicting the distribution of interpolation points.

Remark 5.1 *Following the standard practice, we have estimated the errors by the difference between two successive discretizations of the same problem.*

Example 5.1 *An ellipse with the aspect ratio* $2:1$. *The numerical results are summarized in Table 1 and illustrated in Figure 1.*

Example 5.2 *An ellipse with the aspect ratio* $3:1$. *The numerical results are summarized in Table 2 and illustrated in Figure 2.*

Example 5.3 *An ellipse with the aspect ratio* $10:1$. *The numerical results are summarized in Table 3 and illustrated in Figure 3.*

Example 5.4 *A smile-shaped curve. The numerical results are summarized in Table 4 and illustrated in Figure 4.*

Example 5.5 *A snake-shaped curve. The numerical results are summarized in Table 5 and illustrated in Figure 5.*

Example 5.6 *A spiral-shaped curve. The numerical results are summarized in Table 6 and illustrated in Figure 6.*

Example 5.7 *A star-shaped curve. The numerical results are summarized in Table 7 and illustrated in Figure 7.*

Example 5.8 *A tank-shaped curve. The numerical results are summarized in Tables 8,9 and illustrated in Figure 8.*

The following observations can be made from Tables 1-8.

1. The time requirement of the algorithms grows linearly with the number of nodes N in the discretization.
2. The number of iterations required in the Conjugate Gradient method is almost independent of the number of nodes in the discretization.

Remark 5.2 *Table 9 is given here to illustrate that the number of iteration is proportional to* $\log \epsilon$ *for any given precision* ϵ.

Remark 5.3 *Our experiments show that the interpolation based on the points obtained at very low precision produces very satisfactory results.*

Remark 5.4 *Our numerical experiments indicate that the norms of the Lagrange polynomials are at most slightly greater than one in these examples.*

6 Conclusions

An algorithm is presented for the construction of stable interpolation points on arbitrary curves in the complex plane, upon which the Lagrange interpolation is stable and converges rapidly within the region bounded by that curve. The algorithm is based on the solution of an integral equation by the combination of the Conjugate Gradient Algorithm and the Fast Multipole Method.

Our numerical experiments indicate that the CPU time of the algorithm is roughly proportional to N rather than the theoretical estimate $O(N^{\frac{3}{2}})$, and the number of iterations required in the Conjugate Gradient Method is almost independent of the number of nodes N in the discretization. Furthermore, the interpolation based on points obtained at low precision produces very satisfactory numerical results.

7 Acknowledgments

The author is indebted to Professor V. Rokhlin for his encouragement and many useful discussions, and would like to thank Professor L. Greengard for his interest and help.

References

[1] M. Abramowitz and I. A. Stegun, ed., *Handbook of mathematical functions*, National Bureau of Standards, Applied Mathematics Series 55, 1964.

[2] G. R. Baker and M. J. Shelley, *On the connection between thin vortex layers and vortex sheets*, J. Fluid Mech. (1990), Vol. 215, pp. 261-294.

[3] J. Carrier, L. Greengard and V. Rokhlin, *A Fast Adaptive Multipole Algorithm for Particle Simulations*, SIAM J. Sci. and Stat. Compu., 9(4), 1988.

[4] R. Courant and D. Hilbert, *Methods of Mathematical Physics*, Vol.I, John Wiley & Sons, New York, 1953.

[5] L. Greengard and V. Rokhlin, *A Fast Algorithm for Particle Simulations*, J. Compu. Phys., 73(1987), pp. 325-348.

[6] G. Golub and F. Van Loan, *Matrix Computation*, The Johns Hopkins University Press, Baltimore and London, 1989.

[7] P. Henrici, *Applied and Computational Complex Analysis*, Vol.III, John Wiley & Sons, New York, 1986.

[8] S. Kapur and V. Rokhlin, *High-Order Corrected Trapezoidal Quadrature Rules for Singular Functions*, Technical Report 1042, Yale University, Department of Computer Science, 1994.

[9] N. Kerzman, and R. Stein, *The Cauchy Kernel, the Szego Kernel, and the Riemann Mapping Function*, Math. Ann., 236(1978), pp. 85-93.

[10] P. Koosis, *The Logarithmic Integral*, Vol.2, Cambridge University Press, 1988.

[11] R. Kress, *Linear Integral Equations*, Springer-Verlag, Berlin, 1989.

[12] S. O'Donnell and V. Rokhlin, *A Fast Algorithm for the Numerical Evaluation of Conformal Mappings*, SIAM J. Sci. Stat. Compu., Vol.10, No.3, May, 1989.

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

[14] V. Rokhlin, End-Point Corrected Trapezoidal Quadrature Rules for Singular Functions, *Computers Math. Applic.* Vol.20, No.7, pp.51-62, 1990.

[15] J. Stoer and R. Bulirsch, *Introduction to Numerical Analysis*, Springer-Verlag, 1983.

[16] D. Yu and G. Tryggvason, *The free-surface signature of unsteady two-dimensional vortex flows*, J. Fluid Mech. (1990), vol 218, pp.547-572.

[17] J. L. Walsh, *Interpolation and Approximation by Rational Functions in the Complex Domain*, American Mathematical Society Colloquium Publications, vol. 20, 1935.

Nodes	CPU time (sec.)	Iterations	Residual	Absolute error	Relative error
32	0.5	8	0.90E-06	0.19E-03	0.46E-04
64	1.0	8	0.67E-06	0.52E-07	0.88E-08
128	2.2	8	0.47E-06	0.13E-08	0.16E-09
256	4.4	8	0.33E-06	0.62E-09	0.53E-10
512	8.5	8	0.24E-06	0.55E-09	0.33E-10
1024	17.5	8	0.17E-06		

Table 1: A 2:1 Ellipse

Nodes	CPU time (sec.)	Iterations	Residual	Absolute error	Relative error
32	0.5	8	0.67E-06	0.59E-02	0.18E-02
64	0.9	9	0.24E-06	0.45E-03	0.95E-04
128	2.1	8	0.81E-06	0.13E-08	0.19E-09
256	4.6	8	0.57E-06	0.53E-09	0.56E-10
512	9.0	8	0.40E-06	0.21E-09	0.16E-10
1024	17.5	8	0.29E-06		

Table 2: A 3:1 Ellipse

Nodes	CPU time (sec.)	Iterations	Residual	Absolute error	Relative error
64	2.2	20	0.47E-08	0.26E+00	0.64E-01
128	6.8	26	0.92E-08	0.29E-01	0.50E-02
256	16.9	29	0.71E-08	0.31E-03	0.39E-04
512	37.2	28	0.94E-08	0.61E-04	0.54E-05
1024	59.8	27	0.79E-08	0.72E-07	0.45E-08
2048	107.3	27	0.56E-08		

Table 3: A 10:1 Ellipse

Nodes	CPU time (sec.)	Iterations	Residual	Absolute error	Relative error
64	2.3	29	0.65E-10	0.50E+00	0.12E+00
128	10.0	37	0.91E-10	0.95E-01	0.16E-01
256	27.4	48	0.63E-10	0.25E-02	0.29E-03
512	67.4	51	0.95E-10	0.20E-05	0.16E-06
1024	135.8	50	0.16E-09	0.43E-07	0.25E-08
2048	255.7	50	0.16E-09		

Table 4: A Smile-shaped Curve

Nodes	CPU time (sec.)	Iterations	Residual	Absolute error	Relative error
128	7.2	31	0.87E-09	0.18E+00	0.28E-01
256	30.6	40	0.48E-09	0.73E-02	0.79E-03
512	69.3	43	0.65E-09	0.10E-04	0.79E-06
1024	140.0	42	0.74E-09	0.97E-05	0.52E-06
2048	239.8	40	0.91E-09	0.18E-06	0.69E-08
4096	461.8	40	0.63E-09		

Table 5: A Snake-shaped Curve

Nodes	CPU time (sec.)	Iterations	Residual	Absolute error	Relative error
128	11.5	37	0.53E-09	0.19E+00	0.14E-01
256	27.8	41	0.94E-09	0.74E-02	0.38E-03
512	64.7	46	0.96E-09	0.19E-04	0.70E-06
1024	134.1	45	0.91E-09	0.54E-05	0.14E-06
2048	267.3	44	0.84E-09	0.53E-06	0.97E-08
4096	508.1	43	0.59E-09		

Table 6: A Spiral-shaped Curve

Nodes	CPU time (sec.)	Iterations	Residual	Absolute error	Relative error
64	1.7	23	0.56E-09	0.95E+00	0.12E+00
128	6.8	29	0.66E-09	0.21E+00	0.18E-01
256	21.7	40	0.32E-09	0.69E-02	0.43E-03
512	44.0	40	0.79E-09	0.80E-05	0.35E-06
1024	90.8	40	0.56E-09	0.18E-07	0.55E-09
2048	177.9	40	0.40E-09		

Table 7: A Star-shaped Curve

Nodes	CPU time (sec.)	Iterations	Residual	Absolute error	Relative error
64	2.1	28	0.63E-09	0.33E+00	0.85E-01
128	9.1	35	0.67E-09	0.95E-01	0.17E-01
256	26.5	45	0.38E-09	0.48E-02	0.61E-03
512	57.2	45	0.85E-09	0.54E-05	0.49E-06
1024	108.0	45	0.72E-09	0.19E-06	0.12E-07
2048	213.5	45	0.42E-09		

Table 8: A Tank-shaped Curve

Nodes	CPU time (sec.)	Iterations	Residual	Absolute error	Relative error
64	1.6	21	0.64E-07	0.33E+00	0.85E-01
128	6.4	25	0.71E-07	0.95E-01	0.17E-01
256	17.8	30	0.93E-07	0.46E-02	0.59E-03
512	39.2	31	0.64E-07	0.44E-03	0.40E-04
1024	66.0	28	0.93E-07	0.82E-03	0.52E-04
2048	126.7	27	0.72E-07		

Table 9: A Tank-shaped Curve

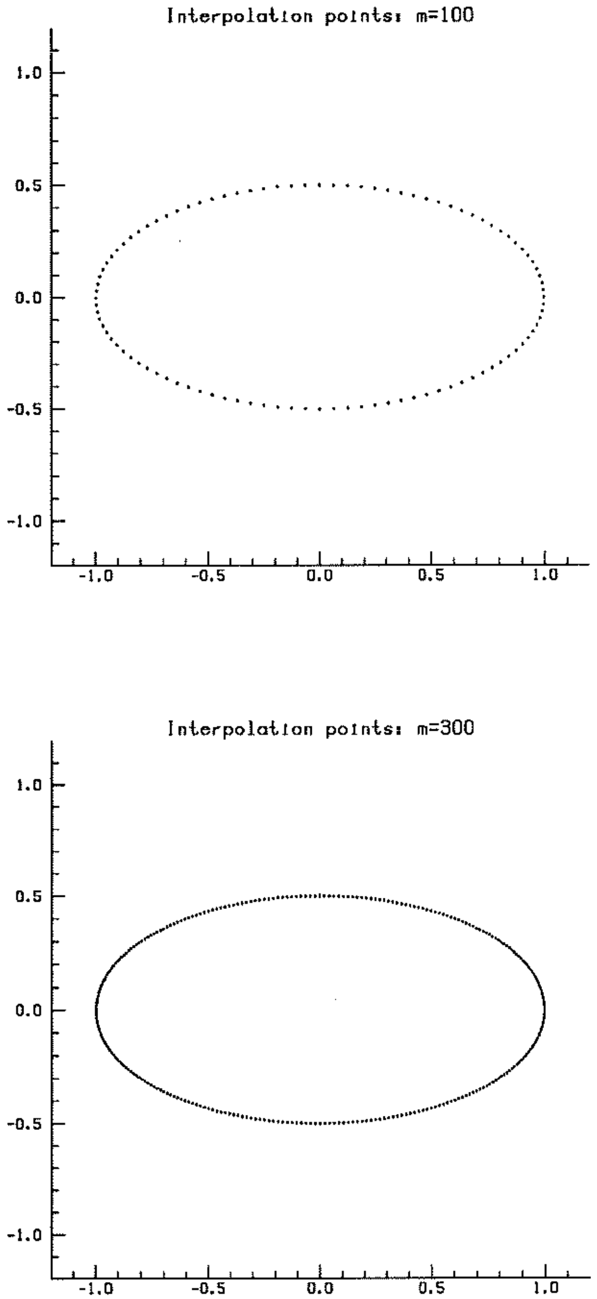


Figure 1: Interpolation points on a 2:1 ellipse

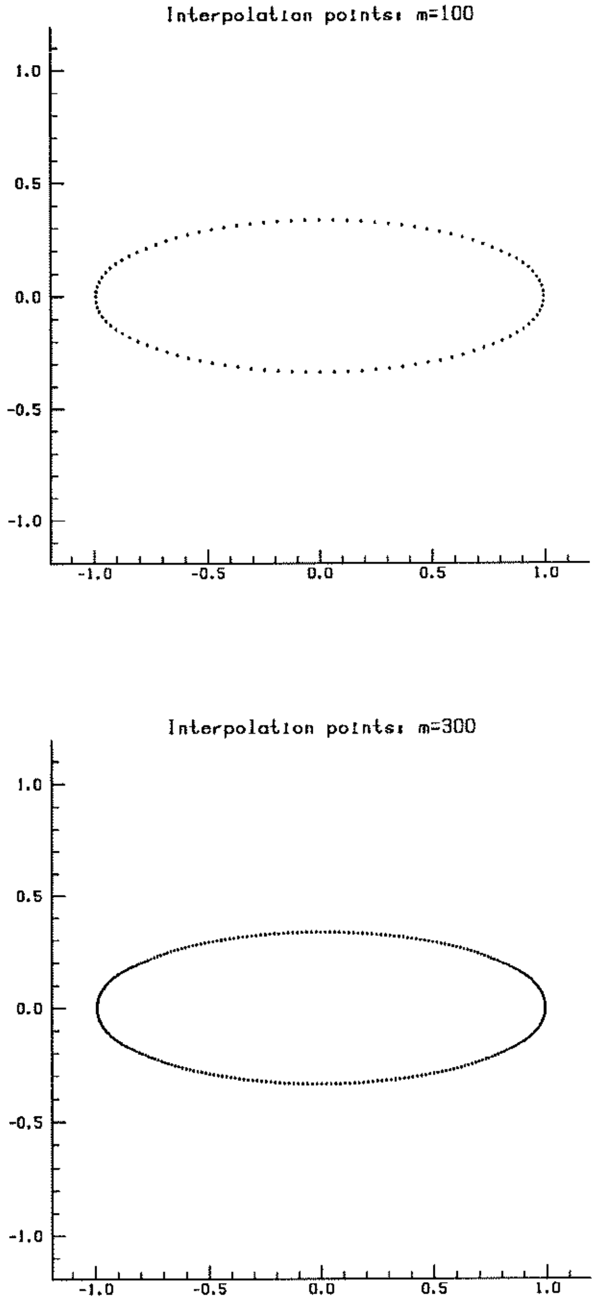


Figure 2: Interpolation points on a 3:1 ellipse

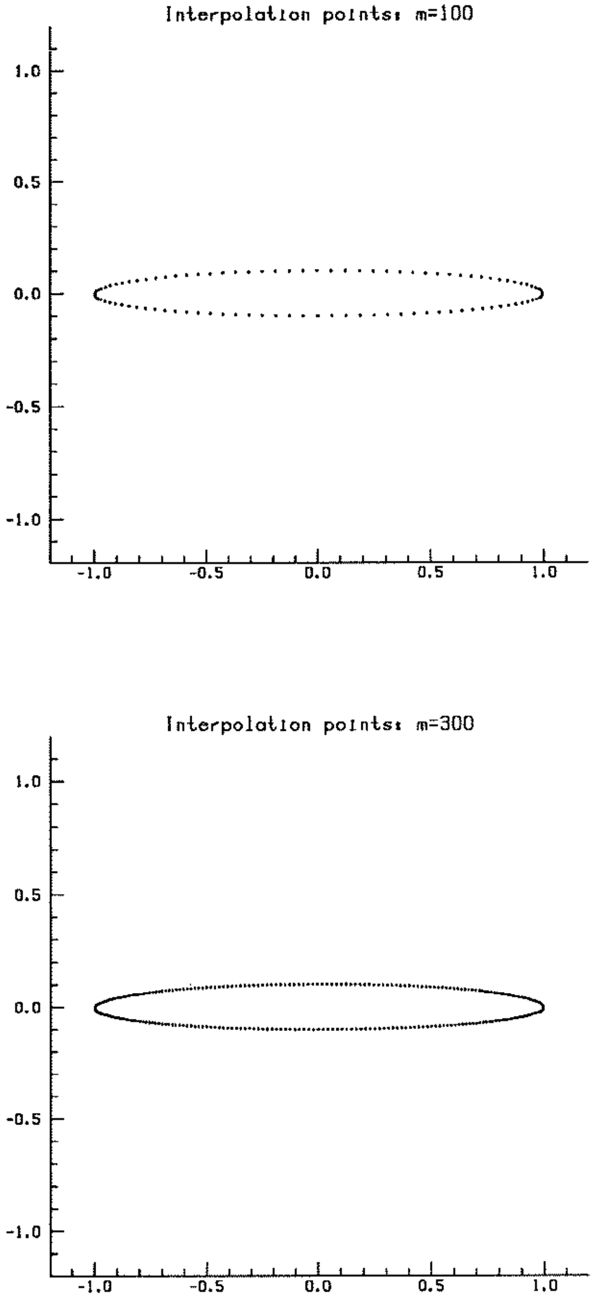
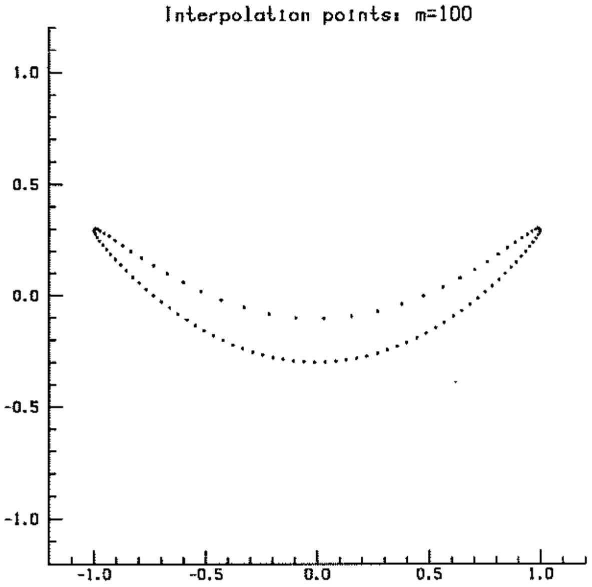


Figure 3: Interpolation points on a 10:1 ellipse



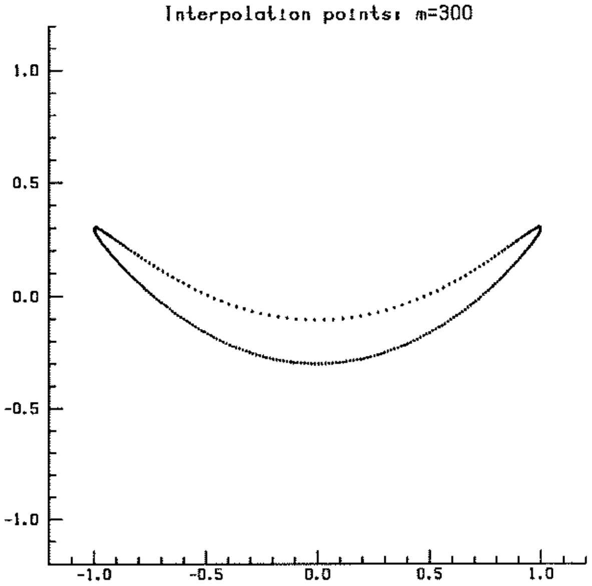


Figure 4: Interpolation points on a smile-shaped curve

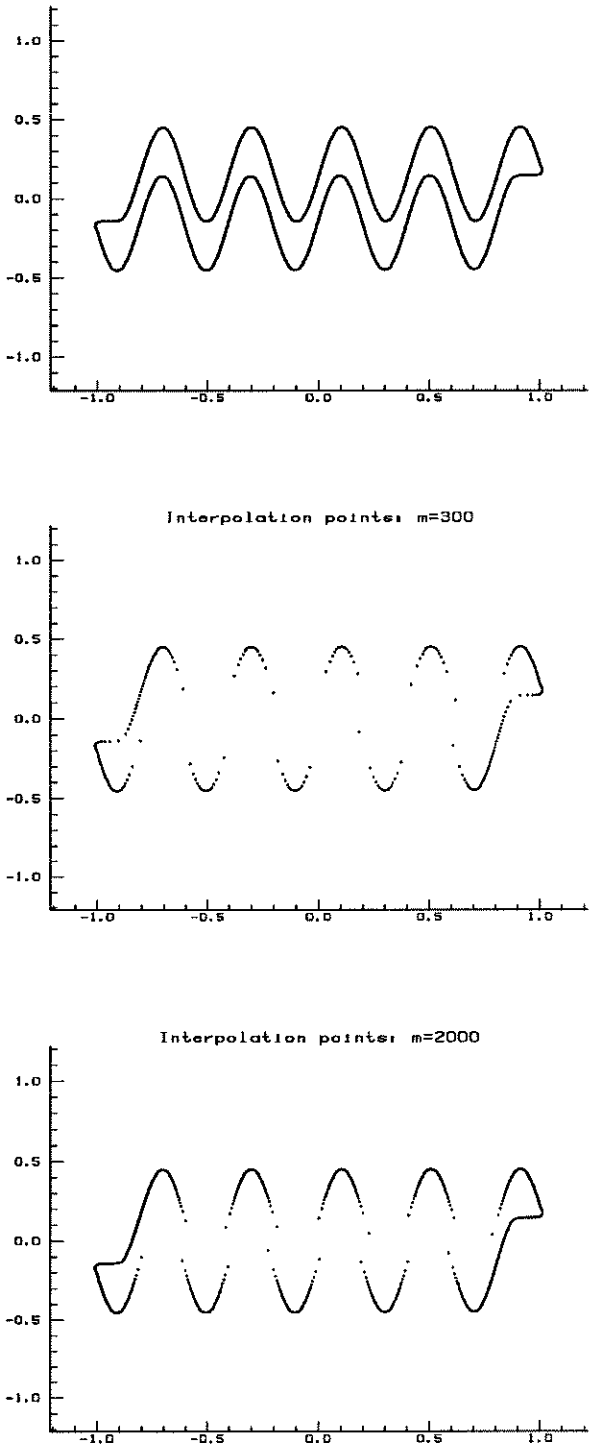
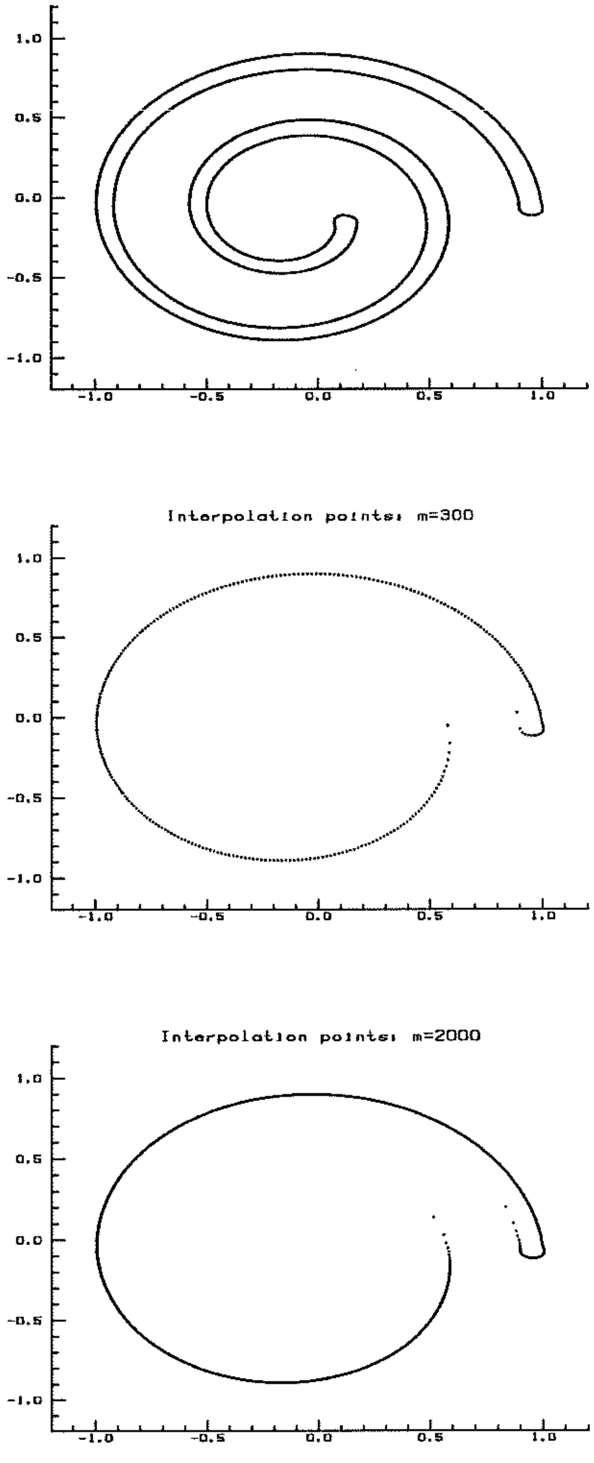


Figure 5: Interpolation points on a snake-shaped curve



Interpolation points: m=300

Interpolation points: m=2000

Figure 6: Interpolation points on a spiral-shaped curve

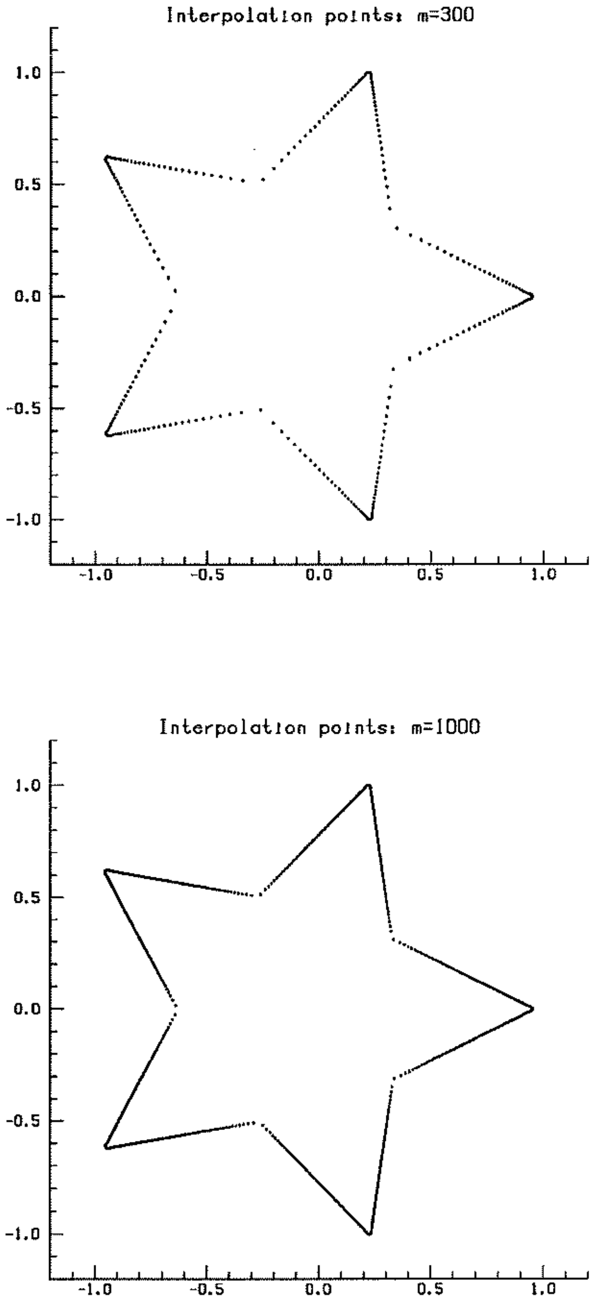


Figure 7: Interpolation points on a star-shaped curve

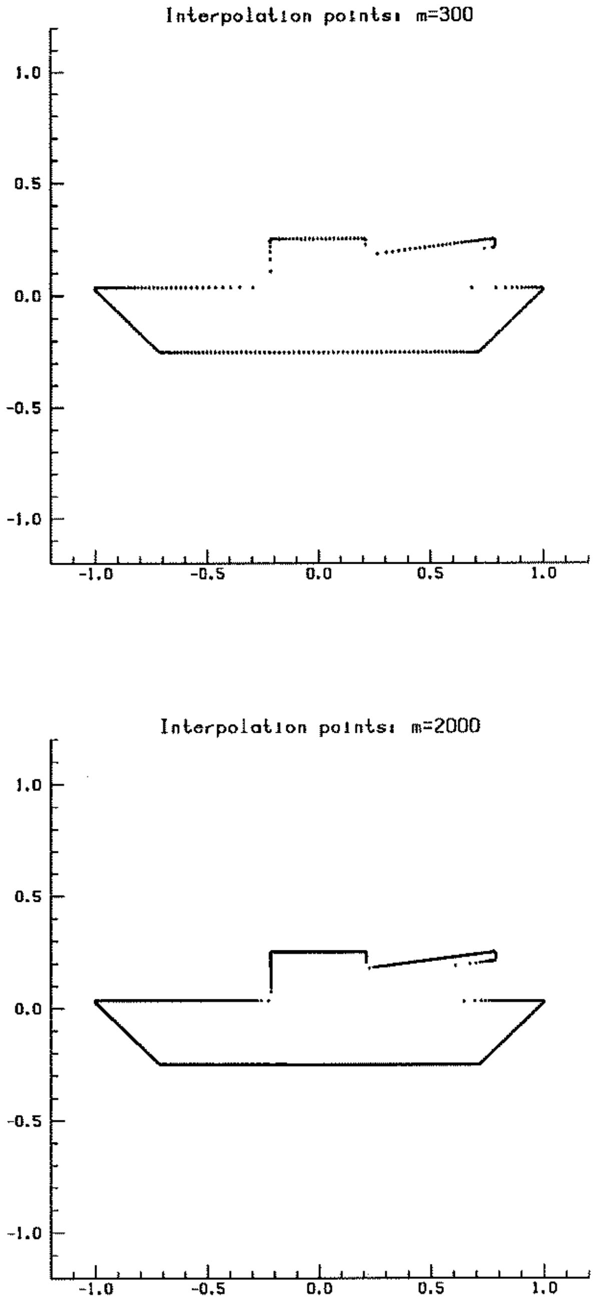


Figure 8: Interpolation points on a tank-shaped curve

j	α_j
1	-.4325921322794724E-01
2	0.3202689042388491E-02

Table 10: Coefficients for the Corrected Trapezoidal Rule (35): p=2

j	α_j
1	-.5024307342079858E-01
2	0.5996233119529027E-02
3	-.4655906795234226E-03

Table 11: Coefficients for the Corrected Trapezoidal Rule (35): p=3

j	α_j
1	-.5462714010179898E-01
2	0.8188266460029230E-02
3	-.1091885919666338E-02
4	0.7828690501786438E-04

Table 12: Coefficients for the Corrected Trapezoidal Rule (35): p=4

j	α_j
1	-.5763224261186158E-01
2	0.9905467894350714E-02
3	-.1735836457536894E-02
4	0.2213870245446548E-03
5	-.1431001195267904E-04

Table 13: Coefficients for the Corrected Trapezoidal Rule (35): p=5

j	α_j
1	-.6471094753809971E-01
2	0.1471321624569456E-01
3	-.4244313571022683E-02
4	0.1219746091263614E-02
5	-.3156154921336350E-03
6	0.6890800654569580E-04
7	-.1195656336479857E-04
8	0.1529459820049702E-05
9	-.1274231726028842E-06
10	0.5166969831297037E-08

Table 14: Coefficients for the Corrected Trapezoidal Rule (35): p=10

j	α_j
1	-.6745551530557688E-01
2	0.1689299430945688E-01
3	-.5726331418330602E-02
4	0.2079973982393914E-02
5	-.7402722529894703E-03
6	0.2463303649153491E-03
7	-.7432197238379932E-04
8	0.1984260034353227E-04
9	-.4580836910292848E-05
10	0.8916453665775555E-06
11	-.1418448246845963E-06
12	0.1767037741742959E-07
13	-.1614096203394717E-08
14	0.9602551109604562E-10
15	-.2789306492547372E-11

Table 15: Coefficients for the Corrected Trapezoidal Rule (35): p=15