

Revisiting the crowding phenomenon in Schwarz-Christoffel mapping

L. Banjai*

May 22, 2007

Abstract

We address the problem of conformally mapping the unit disk to polygons with elongations. The elongations cause the derivative of the conformal map to be exponentially large in some regions. This *crowding* phenomenon creates difficulties in standard numerical methods for the computation of the conformal map. We make use of the Schwarz-Christoffel representation of the mapping and show that a simple change to the existing algorithms introduced by Trefethen [L.N. Trefethen, Numerical computation of the Schwarz-Christoffel transformation, SIAM J. Sci. Statist. Comput., 1(1):82–102, 1980] makes it feasible to accurately compute conformal maps to polygons even in the presence of extreme crowding. For an efficient algorithm it is essential that a good initial guess for the parameters of the Schwarz-Christoffel map is available. A uniformly close initial guess can be obtained from the cross-ratios of certain quadrilaterals as introduced in the CRDT algorithm of Driscoll and Vavasis [T.A. Driscoll, S.A. Vavasis, Numerical conformal mapping using cross-ratios and Delaunay triangulation. SIAM J. Sci. Comput., 19(6):1783–1803, 1998]. We present numerical experiments and compare our algorithms with the CRDT which has been particularly designed to combat crowding.

1 Introduction

Most methods for numerical computation of conformal maps will experience the phenomenon called *crowding*; this term was apparently first used in [20], but it was recognized at least as early as [12]. Crowding arises when the target domain Ω has elongations that are not present in the original domain D . In this case the derivative of the conformal map $f : D \rightarrow \Omega$ is exponentially large in some regions; for small changes of argument z the image $f(z)$ can change by large amounts. If not dealt with properly, such a phenomenon can cause difficulties in a numerical method.

In this paper we let the canonical domain D be the unit disk and Ω a polygon with N vertices w_k , $k = 1, \dots, N$; slits and vertices at infinity are allowed, see [10]. For this special case a semi-explicit formula exists: the Schwarz-Christoffel (SC) formula. The SC formula for the interior map of the unit disk to a polygon is given by

$$f(z) = A + C \int^z \prod_{k=1}^N (\zeta - z_k)^{\alpha_k - 1} d\zeta, \quad (1)$$

where $\alpha_k \pi$ is the interior angle at w_k , z_k the k th *prevertex*, i.e. $z_k = f^{-1}(w_k)$, and A and C are constants. The lower integration limit is left unspecified since it affects only the constant A . The formula is semi-explicit since the prevertices are not given and cannot in general be found analytically. A conformal map between two domains is not unique unless we fix three real parameters. Two common choices are to fix three prevertices, e.g. $z_1 = -1, z_2 = -i, z_3 = 1$, or to fix the *conformal centre* $f(0) = w$ and $f'(0) > 0$. For $N > 3$ this leaves $N - 3$ unknowns in the formula (1).

*lehelb@math.unizh.ch, Institut für Mathematik, Universität Zürich, Winterthurerstrasse 190, CH-8057 Zürich, Switzerland

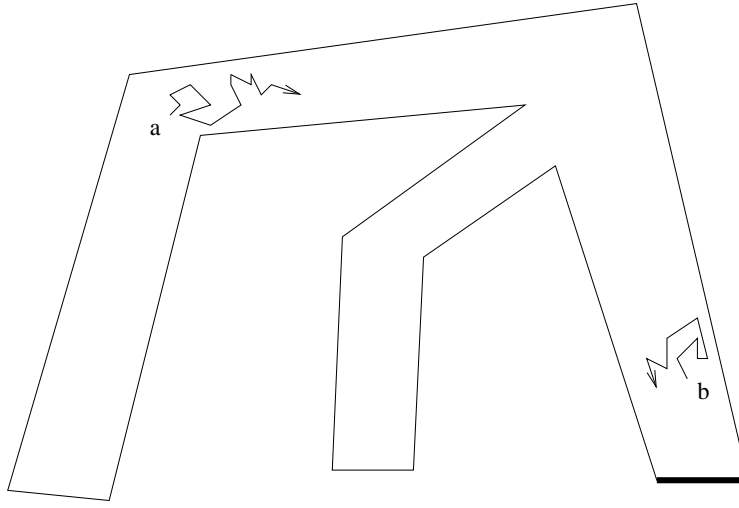


Figure 1: The probability of a Brownian motion starting at a reaching the highlighted edge is extremely (exponentially) small. However, if the starting point is moved to b , the probability of it exiting at the same edge is substantial.

In this setting, the simplest example of crowding is that of a rectangle with aspect ratio a . The minimum distance between two prevertices decays exponentially with increasing a ; see [1, 10]. Therefore a tiny area around these prevertices is mapped to a large part of the rectangle.

An intuitive feeling for the crowding phenomenon in the case of general domains can be obtained by looking at the properties of harmonic measure and Brownian motion; see [13, 23]. The harmonic measure is conformally invariant, hence the harmonic measure, with respect to $f(0)$, of the edge of the polygon connecting w_j and w_{j+1} is proportional to the length of the arc of the circle connecting prevertices z_j and z_{j+1} . In other words, the probability of a Brownian motion, starting at $f(0)$, exiting the polygon through the edge $w_j w_{j+1}$ is proportional to the length of the arc connecting prevertices z_j and z_{j+1} . Given the start $w = f(0)$ of the Brownian motion, it is intuitively clear which prevertices will be extremely close to each other; see Figure 1. Note that for a different choice of $w = f(0)$ different prevertices will be crowded.

Driscoll and Vavasis have used this observation to construct a family of conformal maps, each of which is well conditioned in a different region of the polygon. The complete algorithm uses cross-ratios and Delaunay triangulation, hence the name the CRDT algorithm, and can be used to accurately map extremely crowded regions; see [11]. For singly elongated polygons, to circumvent crowding one can change the canonical domain to the rectangle or strip [18]. This idea can be extended to more general polygons, however for each number of elongations a new canonical domain needs to be devised [17]. If the polygonal shape causes no crowding the "standard" methods introduced by Trefethen in [25] and implemented in a Fortran package SCPACK [26], can be used. All of these methods have been implemented by Driscoll in SC Toolbox [8, 9], a user friendly GUI oriented Matlab toolbox.

One of the reasons why the standard methods cannot be used in the presence of crowding is that if the distance between two prevertices becomes smaller than around 10^{-16} then their floating point representations in double precision are identical. The main, but simple, observation we make is that the *difference* of the consecutive prevertices can be accurately represented in floating point arithmetic up to much smaller distances; in IEEE standard double precision this is around 10^{-308} . Therefore we propose to use the differences of prevertices as the parameters of the SC formula (1) rather than the prevertices themselves. The main message is that to compute maps accurately in presence of crowding the consideration of numerical stability of all the steps in the algorithm is crucial. In this paper we argue, in part by proof and in part by experiment, the following points:

- (i.) with care the standard algorithm of the SC toolbox can be made numerically stable so that

it performs well in presence of extreme crowding

- (ii.) in a stable implementation, the problem of crowding is reflected in the difficulty of solving the parameter problem
- (iii.) the latter problem can be alleviated by using an initial guess which is *uniformly* close to the solution.

We illustrate the modified standard method by many numerical examples. In these examples, when given a good initial guess, the modified method is almost always faster than the CRDT. Further, polygons with multiple elongations and some vertices at infinity, can only be mapped by the method proposed in this paper; see Figure 7. However, if crowding is so severe that the distance between some prevertices cannot be accurately represented in double precision, then CRDT still remains the only effective alternative.

The material presented in this paper is original; most of it appeared as a chapter in the author's D.Phil. thesis [1]. However as early as 1980 [20] Menikoff and Zemach were able to compute conformal maps of extremely crowded regions using similar concepts to the ones described here. The context is different: they consider a certain special class of smooth periodic regions and construct an approximation to the conformal map by solving a suitable first order integral equation. Nevertheless, the main notion that the numerical stability is a crucial issue when dealing with crowded regions is present.

2 Floating point number system

To be able to give a rigorous rounding error analysis of the proposed method, we begin by describing a floating point number system and introducing a model of arithmetic. Our presentation closely follows that of [16]; see also [22].

Let $\mathcal{R} \subset \mathbb{R}$, the floating point number system, be the set of numbers representable in a particular computer architecture. An element of \mathcal{R} has the form

$$y = \pm m \times \beta^{e-t}, \tag{2}$$

where β is the *base*, t the *precision*, e the *exponent* which is in some range $e_{\min} \leq e \leq e_{\max}$, and m is an integer called the *mantissa* satisfying $0 \leq m < \beta^t$. To insure a unique representation it is assumed that $m \geq \beta^{t-1}$ if $y \neq 0$. In that case we say that y is *normalized*. The integers β , t , e_{\min} , and e_{\max} completely characterize the number system. The IEEE standard for double precision has the following values for these parameters:

$$\beta = 2, \quad t = 53, \quad e_{\min} = -1021 \quad \text{and} \quad e_{\max} = 1024. \tag{3}$$

An important constant is *machine epsilon*, ϵ_m , the distance between $1 \times \beta^0$ and the least number in \mathcal{R} larger than 1. It can be shown that if y is a normalized floating point number then the closest normalized floating point number is at least a distance $\beta^{-1}\epsilon_m|y|$ away; see [16]. This gives us the limit of accuracy of representation of a real number in the finite precision system. For the IEEE standard $\beta^{-1}\epsilon_m \approx 1 \times 10^{-16}$ which is approximately the distance at which two prevertices would be indistinguishable in double precision. However, the smallest normalized positive number representable in floating point arithmetic is $\beta^{e_{\min}-1}$ which is in the IEEE standard approximately 2×10^{-308} .

The most useful quantity associated with a floating point system is the *unit roundoff* which is defined as $u = \frac{1}{2}\beta^{1-t}$. Also we define the *rounding* operator fl which maps a real number in the range of \mathcal{R} , i.e. belonging to the interval $[\min \mathcal{R}, \max \mathcal{R}]$, to an element of \mathcal{R} closest to it. The definition of fl will be extended in the next section. The connection between these two definitions is given in the following crucial result whose proof can be found in [16, p. 38]. Note that a real number x is said to be in the *normalized range* of the floating point system if $N_{\min} \leq |x| \leq N_{\max}$, where

N_{\min} is the smallest positive normalized floating point number and N_{\max} is the largest normalized floating point number.

Theorem 1. *If a real number x lies in the normalized range of the floating point system then there exists a $\delta \in \mathbb{R}$ such that*

$$fl(x) = x(1 + \delta), \quad |\delta| < u. \quad (4)$$

2.1 Floating point arithmetic

If u is the unit roundoff, for $x, y \in \mathcal{R}$ we use the following standard model

$$fl(x \text{ op } y) = (x \text{ op } y)(1 + \delta), \quad \delta \in \mathbb{R}, \quad |\delta| \leq u, \quad \text{op} = +, -, *, /, \quad (5)$$

where $fl(\cdot)$ with an argument that is an arithmetic expression denotes the computed value of that expression; if the argument is a real number it denotes the rounded value of that number. The IEEE standard requires this model to hold. We assume that the unary analogue of the above result is true for the elementary functions $\exp, \log, \sin,$ and \cos . Also we shall assume that these results can be extended in a similar way to operations on complex numbers so that for $x, y \in \mathcal{C} := \mathcal{R} + i\mathcal{R}$ we have

$$fl(x \text{ op } y) = (x \text{ op } y)(1 + \delta), \quad \delta \in \mathbb{C}, \quad |\delta| \leq u, \quad \text{op} = +, -, *, /, \quad (6)$$

Again we assume that equivalent results are true for the elementary functions $\exp, \log, \sin,$ and \cos .

Not all the assumptions we make here are true for most computer implementations, but they are true up to some multiples of the roundoff u . These constants are not large enough to be significant so we omit them for simplicity.

3 Stable numerical computation of Schwarz-Christoffel integrals

In this section we are not concerned with the solution of the parameter problem, but only the computation of the SC integral in finite precision arithmetic when all the parameters are known. We wish to accurately compute the integral

$$f(z) = \int_a^z \prod_{j=1}^N (\zeta - z_j)^{\beta_j} d\zeta. \quad (7)$$

Throughout this section, for simplicity we take as the integration path the straight line $[a, z]$ connecting the limit points a and z . The arguments in this section can also be adapted to the case of paths lying on the boundary of the unit disk.

Let θ_j be the argument of the j th prevertex and assume that $z_1 = 1$, i.e. $\theta_1 = 0$. Define

$$\begin{aligned} \phi_j &:= \theta_{j+1} - \theta_j, \\ \tilde{\phi}_j &:= fl(\phi_j) \\ \tilde{z}_j &:= fl(z_j), \quad j = 1, \dots, N, \end{aligned} \quad (8)$$

with $\theta_{N+1} = \theta_1 = 0$.

Note that the numbers $\{\phi_j\}$ characterize the function (7) completely under the assumption that the first prevertex is at 1; otherwise they characterize it up to a multiplicative constant. Both sets $\{\tilde{z}_j\}$ and $\{\tilde{\phi}_j\}$ can be used to compute the SC integral. However as soon as two prevertices have their real or imaginary parts equal in at least the first significant digit the latter set contains more information and hence is the more appropriate one as a representation of the function (7). It remains to show how to compute the integral from the numbers $\{\tilde{\phi}_j\}$. We begin with two lemmas that show how to compute the integrand. In all the results in this section we single out one prevertex z_k which is, up to a constant multiple, the closest prevertex to the path taken in the integral.

Lemma 2. Let z, z_k , and z_j be such that $|z - z_k| \leq C_1|z - z_j|$. Let $\tilde{\eta} = (z - z_k)(1 + \delta_1)$, $\tilde{\omega}_j = fl(z_j - z_k)$, and $\tilde{\beta}_j = fl(\beta_j)$, where for some $C \geq 0$, $|\delta_1| < Cu$. Then

$$fl\left((\tilde{\eta} - \tilde{\omega}_j)^{\tilde{\beta}_j}\right) = (z - z_j)^{\beta_j}(1 + \delta) + O(u^2), \quad |\delta| < (7 + 3C_1(C + 1) + 3\left|\log|z - z_j|\right|)u. \quad (9)$$

Proof.

$$\begin{aligned} fl\left((\tilde{\eta} - \tilde{\omega}_j)^{\tilde{\beta}_j}\right) &= (((z - z_k)(1 + \delta_1) - (z_j - z_k)(1 + \delta_2))(1 + \delta_3))^{\beta_j(1 + \delta_4)}(1 + \delta_5) \\ &= (z - z_j + (z - z_k)\delta_1 + (z_k - z_j)\delta_2)^{\beta_j(1 + \delta_4)}(1 + \delta_3)^{\beta_j(1 + \delta_4)}(1 + \delta_5) \\ &= (z - z_j)^{\beta_j(1 + \delta_4)}\left(1 + \frac{z - z_k}{z - z_j}\delta_1 + \frac{z_k - z_j}{z - z_j}\delta_2\right)^{\beta_j(1 + \delta_4)} \times \\ &\quad (1 + \delta_3)^{\beta_j(1 + \delta_4)}(1 + \delta_5), \end{aligned}$$

where $\delta_i \in \mathbb{C}$ and $|\delta_i| < u$, $i = 2, \dots, 5$. Since

$$\left|\frac{z - z_k}{z - z_j}\right| \leq C_1, \quad \text{and} \quad \left|\frac{z_k - z_j}{z - z_j}\right| = \left|1 + \frac{z_k - z}{z - z_j}\right| \leq 1 + C_1$$

we can write the above expression as

$$\begin{aligned} &(z - z_j)^{\beta_j(1 + \delta_4)}\left(1 + \beta_j\frac{z - z_k}{z - z_j}\delta_1 + \beta_j\frac{z_k - z_j}{z - z_j}\delta_2 + \beta_j\delta_3 + \delta_5\right) + O(u^2) \\ &= (z - z_j)^{\beta_j}\left(1 + \beta_j\log|z - z_j|\delta_4 + \beta_j\frac{z - z_k}{z - z_j}\delta_1 + \beta_j\frac{z_k - z_j}{z - z_j}\delta_2 + \beta_j\delta_3 + \delta_5\right) + O(u^2) \\ &= (z - z_j)^{\beta_j}(1 + \delta) + O(u^2), \end{aligned}$$

where

$$|\delta| \leq (1 + |\beta_j|(2 + C_1(C + 1) + \left|\log|z - z_j|\right|))u \leq (7 + 3C_1(C + 1) + 3\left|\log|z - z_j|\right|)u.$$

Here we have used the fact $|\beta_j| \leq 3$ and the expansion $a^{\delta_4} = 1 + \log(|a|)\delta_4 + O(\delta_4^2)$. \square

Lemma 3. Let z and z_k be such that $|z - z_k| \leq C_1|z - z_j|$, $j = 1, 2, \dots, N$. Let $\tilde{\eta} = (z - z_k)(1 + \delta_1)$, $\tilde{\omega}_j = fl(z_j - z_k)$ and $\tilde{\beta}_j = fl(\beta_j)$, where for some $C \geq 0$, $|\delta_1| < Cu$. Then

$$fl\left(\prod_{j=1}^N(\tilde{\eta} - \tilde{\omega}_j)^{\tilde{\beta}_j}\right) = \prod_{j=1}^N(z - z_j)^{\beta_j}(1 + N\delta) + O(u^2), \quad (10)$$

where

$$|\delta| < (8 + 3C_1(C + 1) + 3\max_j\left|\log|z - z_j|\right|)u. \quad (11)$$

Proof. This is an easy consequence of the previous result. \square

It is clear that from $\{\tilde{\phi}_j\}$ we can compute expressions $z_j - z_k$ accurately and hence in the above results one of the assumptions is that the rounding of these numbers is known. In practice this is true up to a small constant which we omit for simplicity. However, for the above estimate to be useful constants C and C_1 must not be too large. As we will see later in this section, an easy bound can be found for the constant C_1 . For C not to be large, the position of the evaluation point z needs to be accurately known relative to the closest prevertex. This is a necessary assumption, since the user must accurately specify at which point the integral should be evaluated. How the user determines which prevertex is closest to z and their relative positions is problem specific. In this aspect, a particularly interesting situation occurs when computing the inverse function $f^{-1}(w)$; this problem is discussed in the section on applications.

From above discussion we see that we can assume that the relative positions of both integration limit points to the prevertices are accurately known. Note that this statement remains true, if the integral needs to be split:

$$\int_a^z = \int_a^w + \int_w^z. \quad (12)$$

To numerically evaluate the integral (7) we use Gauss-Jacobi quadrature [14], which is a highly accurate method for the computation of integrals of the form

$$\int_{-1}^1 g(x)(1+x)^\alpha(1-x)^\beta dx,$$

for smooth functions $g(\cdot)$. For example, to evaluate f at the point $b = z_N$, assuming a is not a singular point, we can rewrite (7) as

$$\begin{aligned} f(b) &= \frac{b-a}{2} \int_{-1}^1 \prod_{k=1}^{N-1} \left(\frac{b-a}{2}x + \frac{b+a}{2} - z_k \right)^{\beta_k} \left(\frac{b-a}{2}x - \frac{b-a}{2} \right)^{\beta_N} dx \\ &= - \left(\frac{a-b}{2} \right)^{1+\beta_N} \int_{-1}^1 \prod_{k=1}^{N-1} \left(\frac{b-a}{2}x + \frac{b+a}{2} - z_k \right)^{\beta_k} (1-x)^{\beta_N} dx, \end{aligned}$$

which is precisely in the form required by Gauss-Jacobi quadrature. If b is not a singular point, analogously the following form is obtained

$$\frac{b-a}{2} \int_{-1}^1 \prod_{k=1}^N \left(\frac{b-a}{2}x + \frac{b+a}{2} - z_k \right)^{\beta_k} dx. \quad (13)$$

The Gauss-Jacobi quadrature accounts for the singularities at the endpoints. If other prevertices are close to the interval of integration the rate of convergence of the quadrature is severely degraded. To avoid this problem Trefethen [25] proposed using a compound Gauss-Jacobi method, in which the integration interval is subdivided if some prevertices are too close to the interval. Following the ‘‘one-half rule’’ for this subdivision proposed in [25], we make the following assumption:

$$\frac{1}{2} \leq \frac{|\zeta - z_j|}{|b-a|}, \quad \text{for all } \zeta \in [a, b] \text{ and } z_j \notin \{a, b\}. \quad (14)$$

Since there is no need to subdivide the interval unless a singularity is in the vicinity of the interval, it is also reasonable to make the following assumption:

$$\text{There exists } k, \text{ such that } \frac{|\zeta - z_k|}{|b-a|} \leq 1, \quad \text{for all } \zeta \in [a, b]. \quad (15)$$

These assumptions now give us a bound on the constant C_1 occurring in the previous two lemmas:

$$|\zeta - z_k| \leq |b-a| \leq 2|\zeta - z_j|, \quad \text{for all } \zeta \in [a, b] \text{ and } z_j \notin \{a, b\}. \quad (16)$$

Theorem 4 investigates the rounding errors involved in the computation of the non-singular integral (13). The argument is almost identical for the singular integral, since the extra term $(1-x)^{\beta_N}$ is absorbed in the quadrature routine. Notice that in this result, unlike before, we give a bound on the absolute error.

Theorem 4. *Let $\{x_l\}$ and $\{w_l\}$ be the nodes and weights of the Gaussian quadrature rule described above, such that for some $\epsilon > 0$*

$$\left| \frac{b-a}{2} \sum_{l=1}^M F(\zeta_l)w_l - \int_a^b F(\zeta)d\zeta \right| \leq \epsilon, \quad (17)$$

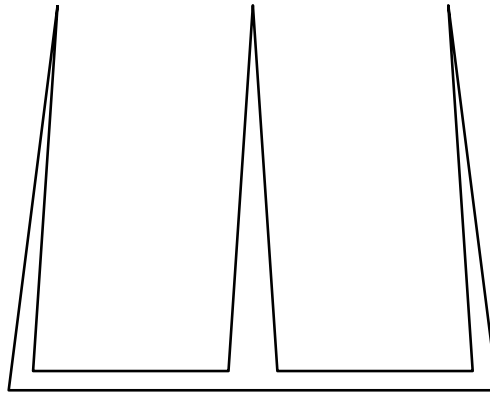


Figure 2: A polygon with small internal angles.

where

$$F(z) = \prod_{j=1}^N (z - z_j)^{\beta_j} \quad \text{and} \quad \zeta_l = \frac{b-a}{2}x_l + \frac{b+a}{2}.$$

Further, let $\tilde{\nu}_1 = fl(a - z_k)$, $\tilde{\nu}_2 = fl(b - z_k)$, $\tilde{\omega}_j = fl(z_j - z_k)$ and $\tilde{\beta}_j = fl(\beta_j)$. Then, under the assumptions (14) and (15),

$$\left| fl \left(\frac{\tilde{\nu}_2 - \tilde{\nu}_1}{2} \sum_{l=1}^M \prod_{j=1}^N \left(\frac{1}{2} ((\tilde{\nu}_2 - \tilde{\nu}_1)\tilde{x}_l + \tilde{\nu}_1 + \tilde{\nu}_2) - \tilde{\omega}_j \right) \tilde{w}_l \right) - \int_a^b F(\zeta) d\zeta \right| \leq \epsilon + |\delta|,$$

where

$$|\delta| \leq \left(71N + 3 \max_j |\log |b - z_j|| N + M + 7 \right) Au \quad (18)$$

and

$$A = \frac{|b-a|}{2} \sum_{l=1}^M \prod_{j=1}^N |\zeta_l - z_j|^{\beta_j} |w_l|. \quad (19)$$

Proof. The proof is given in the appendix. \square

The constant A is more difficult to estimate. In [1] a running error analysis has been performed for a number of polygons. The maximum value of A occurring during the iterative solution for various polygons is listed in Table 1.

polygon	A	polygon	A
Fig. 4: Y (crowded)	24.7	Fig. 2: Fork	1.00×10^4
Fig. 5: Spiral	15.1	Fig. 6: Emma's maze	45.8
Rectangle(200)	585	Fig. 7: ∞ Polygon	9.44

Table 1: Maximum value of A , see (19), reached during the computation of maps to various polygons.

We can see that for all but one polygon we would not expect to lose much accuracy due to rounding errors. For the Fork polygon we may expect to lose a few more digits. However, in the last few iterations A is much smaller, around 30.2 and we were able to find the map to accuracy of around 10^{-14} . Similar behaviour is typical for polygons with small internal angles.

The polygons we consider in this paper have only a few vertices, but in [2] we consider polygons with hundreds of thousands of vertices. For these polygons the above estimate would not be sharp, since our error bound is for the worst case. As a rule of thumb we can expect that replacing N in

(18) with \sqrt{N} would give a more realistic bound. For a thorough discussion of such estimates, see Wilkinson [27], Higham [16], and the references within.

We now turn to the problem of determining the parameters $\{\tilde{\phi}_j\}$ for an arbitrary polygon.

4 Parameter problem

As with the numerical evaluation of the SC integral, existing methods described in [10, 25] for the solution of the parameter problem need to be made stable.

We fix three prevertices $z_1 = 1$, $z_{N-1} = -1$, and $z_N = -i$ which implies $\sum_{k=1}^{N-2} \phi_k = \pi$ and $\phi_{N-1} = \phi_N = \pi/2$. We are left with $N - 3$ real quantities to determine. For a bounded polygon, this is achieved by the following $N - 3$ real conditions:

$$\frac{\left| \int_{z_j}^{z_{j+1}} \prod_{k=1}^N (\zeta - z_k)^{\beta_k} d\zeta \right|}{\left| \int_{z_1}^{z_2} \prod_{k=1}^N (\zeta - z_k)^{\beta_k} d\zeta \right|} = \frac{|w_{j+1} - w_j|}{|w_2 - w_1|}, \quad j = 2, 3, \dots, N - 2. \quad (20)$$

For the case of polygons with infinite vertices, a pair of real conditions in (20) are replaced by a single complex condition; for details see [10].

Note that we have a constrained nonlinear system to solve. Inspired by the choice made in [25], we use the following, numerically stable, transformation to formulate an equivalent unconstrained nonlinear system:

$$\psi_j = \log \left(\frac{\phi_{j+1}}{\phi_1} \right), \quad j = 1, \dots, N - 3. \quad (21)$$

The variables ϕ_j can be recovered by the formulas

$$\phi_1 = \frac{\pi}{\sum_{j=1}^{N-3} e^{\psi_j} + 1}, \quad (22)$$

$$\phi_j = e^{\psi_{j-1}} \phi_1, \quad j = 2, \dots, N - 2. \quad (23)$$

Multiplication, division, and summation of positive real numbers are the only operations used, hence we do not expect significant loss of accuracy due to rounding errors.

Now that we have a stable way of expressing the nonlinear system in the unconstrained variables $\{\psi_j\}$ we can solve the system by an iterative numerical method. We use the same solver used in the SC Toolbox so that we can compare the speed and accuracy of the two methods. In initial tests, see [1], our algorithm has shown to be able to compute maps to extremely crowded regions. However, for some particularly difficult regions the nonlinear solver needed many iterations to converge. This resulted in the method being slower than the CRDT for these regions. Fortunately, the CRDT furnishes us with a uniformly good initial guess which alleviates this problem significantly. The details are given in the next section.

5 A uniformly good initial guess

The CRDT algorithm uses as primitive variables cross-ratios of quadrilaterals formed by prevertices, instead of the prevertices themselves. The cross-ratio of an ordered 4-tuple (a, b, c, d) is defined as, see [21],

$$\rho(a, b, c, d) = \frac{(d - a)(b - c)}{(c - d)(a - b)}. \quad (24)$$

We list some of the properties of cross-ratios:

- If the four points are in counterclockwise order along the boundary of a disk, the cross-ratio is real and negative.
- Cross-ratios are invariant under Möbius transformations.

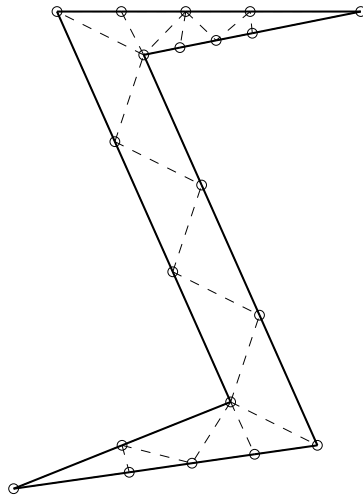


Figure 3: Delaunay triangulation of a polygon once extra vertices have been added to prevent the occurrence of ill-conditioned quadrilaterals.

- Let a, b, c be points on the unit circle and let r be an arbitrary negative real number. Then there exists a unique point d on the unit circle such that $\rho(a, b, c, d) = r$.

Driscoll and Vavasis devised an algorithm that uses Delaunay triangulation to find $N - 3$ quadrilaterals Q_i with vertices $w_{i_1}, w_{i_2}, w_{i_3}, w_{i_4}$, $i = 1, \dots, N - 3$, where w_{i_j} are the vertices of the polygon. The next step in the algorithm is to find an embedding of prevertices for each quadrilateral so that the conformal map is well-conditioned nearby the four prevertices. If one quadrilateral Q_J were itself long and narrow, it would not be possible to find such an embedding. Driscoll and Vavasis suggest that degenerate vertices be added to the polygon to insure that this does not happen. In Figure 3 we show a Delaunay triangulation of a polygon after the extra vertices have been added. There are $N = 20$ vertices and $N - 3 = 17$ diagonals interior to the polygon. Each diagonal defines a single quadrilateral. This quadrilateral is constructed from the two triangles whose one side coincides with the diagonal.

Further, the authors of the CRDT propose to use the cross ratios for the initial guess:

$$\rho(\tilde{z}_{j_1}, \tilde{z}_{j_2}, \tilde{z}_{j_3}, \tilde{z}_{j_4}) = \rho(w_{j_1}, w_{j_2}, w_{j_3}, w_{j_4}).$$

In [4] it is proved that the above initial guess is uniformly close to the solution, in the following sense: There exists a constant $K < \infty$, independent of the polygon, such that

$$d_{\text{QC}}(\{\tilde{z}_j\}, \{z_j\}) \leq \log K, \quad (25)$$

where

$$d_{\text{QC}}(\{\tilde{z}_j\}, \{z_j\}) = \inf\{\log K : \exists K - \text{quasiconformal } h : D \rightarrow D \text{ such that } h(z_j) = \tilde{z}_j\}.$$

We stress that an estimate (25) could not be proved without the addition of extra vertices. However, once the initial guess is computed the extra vertices can again be removed, thereby avoiding having to solve a larger nonlinear system.

It still remains to discuss whether the initial guess can be computed in a stable way. Let us consider a problem of determining d when a, b, c , and the cross ratio $r = \rho(a, b, c, d)$ are known. In fact we assume that the differences $b - c, b - a$, and $c - a$ are known to double precision. We can compute accurately the differences $d - c$ and $d - a$ as follows

$$d - c = \frac{a - c}{h + 1}, \quad d - a = h \frac{c - a}{h + 1}, \quad \text{where } h = r \frac{b - a}{c - b}.$$

The above expressions can be derived from Lemma 2 of [11]. Since the four points a, b, c, d are in the counterclockwise order the difference $d - b$ can also be computed accurately from the above two differences. The vector connecting d to some already computed prevertex \tilde{z}_j can also be computed from these differences, but perhaps not accurately. Nevertheless, we use this method to compute an initial guess. Numerical instability at this stage does not influence the accuracy of the computed solution, but might require the solver to take more iterations to converge.

We conclude with a section that displays the capabilities of the improved method for computation of SC maps. These will also support the choice, and the computation, of the initial guess.

6 Numerical experiments

In this section we show a few examples of conformal maps and compare the behaviour of our proposed method to that of the methods available in the SC Toolbox [8, 9]. For regions with crowding, we compare the cross-ratio Delaunay triangulation (CRDT) [11] and standard SC Toolbox (SCT) [25] methods with our more stable version of SCT (SCTS). SCTS is obtained by modifying the SC Toolbox to include the modifications described in the previous sections. To solve the nonlinear problem of finding the unknowns $\{\psi_j\}$ we use a Gauss-Newton method with a Broyden update of the Jacobian described in [7]. We use an implementation NESOLVE of this method written by Behrens; the same code is used in the SC Toolbox. As a measure of the accuracy of computed maps, we use the maximum of the error in the conditions (20) used to define the non-linear parameter problem.

The examples support the following statements:

1. SCTS is not significantly slower than SCT when both work.
2. Even for mildly crowded regions SCTS typically obtains higher accuracy than SCT.
3. The initial guess for SCTS can reduce the number of iterations as much as ten fold.
4. SCTS with the initial guess is occasionally slower and often much faster than the CRDT.

We have tested our code on a number of arbitrarily chosen polygons and polygons from the literature that could also be mapped by SCT. A comparison of running times to compute these maps to an accuracy of about 1×10^{-8} for the two methods has revealed that at least for these polygons, SCTS is never slower by more than 50% than SCT. It was also noticed that the differently realized unconstrained system of the SCTS occasionally requires more evaluations of the nonlinear function to converge. If we require higher accuracy for the maps we can see that even mildly crowded regions cannot be computed to high accuracy by the SCT. For example the region on the left in Figure 4 can only be computed to accuracy of 4×10^{-10} even though the minimum distance between two prevertices is not smaller than 1×10^{-4} . SCTS can, however, compute this map to accuracy of 4×10^{-16} in 3 seconds. Even the polygon on the right of Figure 4 can be computed without difficulty to an accuracy of 4×10^{-15} even though the closest two prevertices are at distance 8×10^{-19} apart. With this we have given evidence to support the first two claims given above and these two claims in turn imply that there is little reason to give preference to SCT over SCTS in any situation.

We have already mentioned one example of a highly crowded region that can be mapped by the new method. A number of others will follow, the results are shown in Table 2. There we compare the number of nonlinear function evaluations and the CPU time in seconds needed by the two methods to solve the nonlinear system for various polygons to an accuracy of 10^{-8} . The polygons are shown in figures scattered around the section. We use two choices for the initial guess for SCTS: the standard one, equally spaced prevertices, and the uniformly close initial guess provided by the cross-ratios.

As we have alluded to before, there is a limit to what can be mapped by SCTS. The rounding error results in Section 3 assume that the distance between the arguments of prevertices is accurately

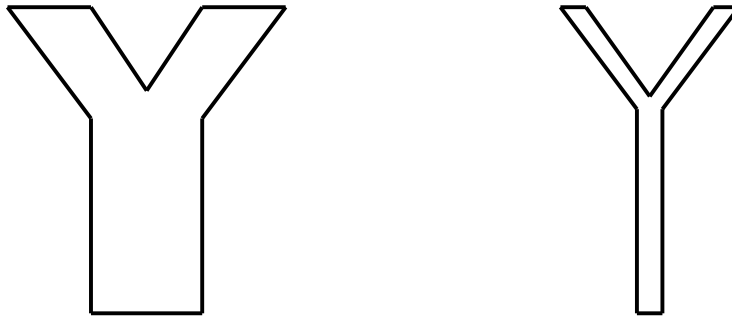


Figure 4: A mildly crowded polygon for which the minimum distance between prevertices is about 1×10^{-4} and a more extreme variation for which this distance is not smaller than 8×10^{-19}

Figure	polygon	CRDT	SCTS	SCTS with initial guess
2	Fork	109/102.3s	181/6.1s	91/5.2s
4 right	Y (crowded)	30/7.4s	164/9.1s	34/1.8s
5	Spiral	71/33.7s	185/14s	41/3.5s
6	Emma's maze	33/59.1s	1326/411.3s	368/114.0s
-/-	Rectangle (200)	31/161.0s	28/2.8s	3/2.4s
-/-	Rectangle (250)	28/154.8s	-/-	-/-
7	∞ Polygon	-/-	268/20.9s	-/-

Table 2: Number of evaluations of the nonlinear function are shown against the time in seconds needed to compute a map to the listed polygons. We compare the results obtained using CRDT and the algorithm described in this paper (the SCTS).

representable in floating point arithmetic. This is not true for a rectangle of aspect ratio 250, yet such a rectangle can be mapped by CRDT. However the rectangle is also an example for which SCTS, when it works, performs strikingly better than CRDT. CRDT added 160 vertices to the rectangle of aspect ratio 200, so that a system of size 161 had to be solved whereas SCTS had to solve a system of size only 1.

For polygons Y, Fork, and Spiral, the SCTS also performs well. Even without the uniformly close initial guess the convergence of the nonlinear solver is satisfactory. The CRDT is slow for these polygons since it needs to add extra vertices to help make the quadrilaterals better conditioned. To produce well conditioned quadrilaterals for the Fork polygon CRDT added 55 vertices. Driscoll and Vavasis state that often many vertices need to be added “near sharp corners and in narrow channels of the region” [11]. This is a well recognized problem with the CRDT.

A recognized advantage of the CRDT is that the nonlinear system formed in terms of cross-ratios is very well behaved. Due to this good behaviour of the nonlinear system CRDT is more than 6 times faster for “Emma’s maze” polygon when SCTS uses no initial guess. However with the initial guess SCTS is only two times slower than the CRDT though it still needs many more evaluations of the nonlinear function. This example of a polygon was taken from the CRDT paper [11].

The final example is an elongated polygon with two vertices at infinity, see Figure 7. Maps to such polygons cannot be computed by CRDT, but can by the SCTS. However, we also cannot obtain an initial guess using the CRDT.

7 Applications and further computational issues

The applications of conformal mapping are numerous, the most famous being the solution of the Laplace’s equation. For many other we refer the reader to [15, 24] and, for the applications of the

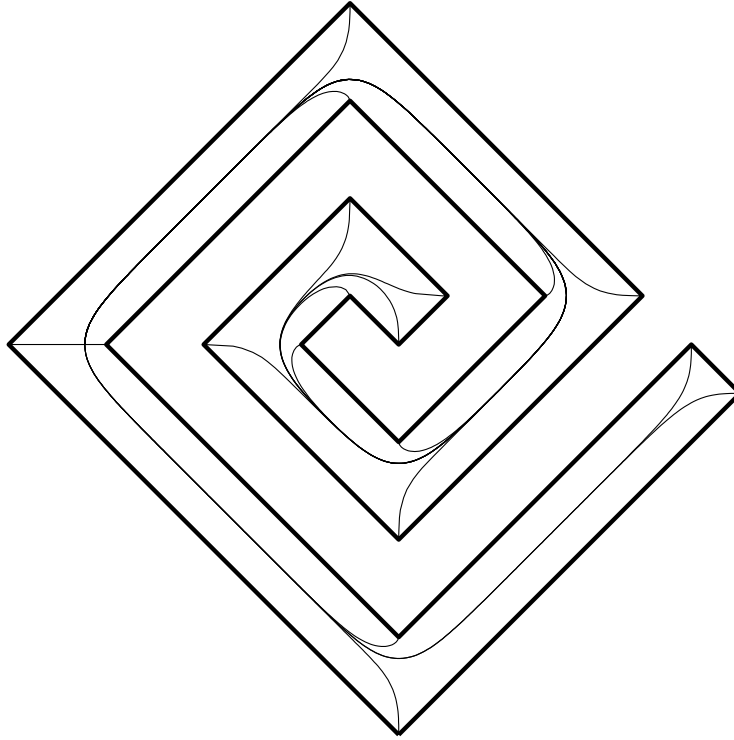


Figure 5: Conformal map from the disk to a spiral shaped polygon. Images of radial lines connecting the origin with the prevertices are shown.

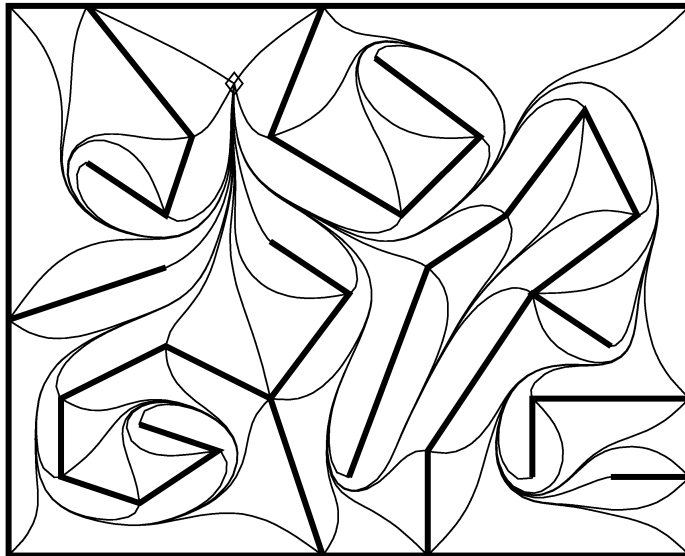


Figure 6: A polygon for which SCTS converges slowly unless a good initial guess is provided.

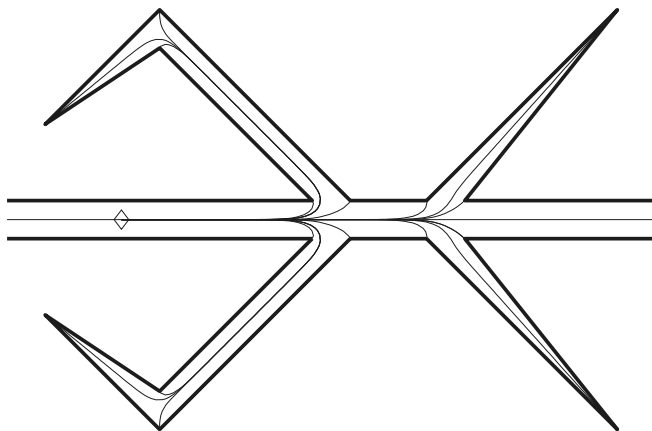


Figure 7: Conformal map from the disk to a polygon with a number of elongations and with vertices at infinity. Images of radial lines connecting the origin with the prevertices are shown.

Schwarz-Christoffel mapping, to the monograph [10]. The algorithms introduced in the previous sections open the possibility of extending these application areas also to the cases where elongated polygons appear. Since the computation of the conformal map is often only one step in a larger computational procedure, we describe next a few standard applications of SC mapping.

Transplantation of a problem posed on a complicated domain to a problem on a canonical domain is a classical application of conformal mapping. In [2] a conformal map was used to transplant an eigenvalue problem from a fractal domain to the unit disk. The map was approximated by a Schwarz-Christoffel map from the unit disk to a polygon with $3 \times 4^9 \approx 768 \times 10^3$ vertices. To make this possible, the fast multipole method was used to accelerate the computations; see [3]. Further, a simple iteration, Davis's iteration [6], was used to solve the nonlinear problem, thereby avoiding the high complexity of a nonlinear solver such as NESOLVE. To obtain an accurate map the considerations of numerical stability at all stages of the computation were essential. In particular, the fast multipole method needed to be implemented with care. For details of the implementation, see [1].

Another standard application of conformal mapping is the computation of the Green's function: Given a region Ω and a source point $w_0 \in \Omega$, the Green's function $g(w)$ is defined by the conditions

$$\begin{aligned}
 \Delta g(w) &= 0 & \text{for } w \in \Omega \setminus \{w_0\}, \\
 g(w) &\rightarrow 0 & \text{for } w \rightarrow \partial\Omega, \\
 g(w) &\sim -\log |w - w_0| & \text{for } w \rightarrow w_0.
 \end{aligned}
 \tag{26}$$

If f is a conformal map of the unit disk to Ω , then $g(w) = -\log |f^{-1}(w)|$ is the solution of the above problem. In this paper we have chosen to represent the maps from the unit disk by plotting the images of radial lines. In view of the above problem, these are the flow lines of the point charge centred at the conformal centre w_0 (image of the origin under the disk map f). So far the conformal centre of the maps we have computed was determined by the choice of the three fixed prevertices. To solve the above problem we could, in the parameter problem, fix a single prevertex and require the conformal centre to be at w_0 . However, assuming that a conformal map from the disk to Ω has already been computed, the map with the required conformal centre is obtained by mapping the prevertices under a Möbius automorphism of the disk h , which sends $z_0 = f^{-1}(w_0)$ to 0.



Figure 8: To compute $z_0 = f^{-1}(w_0)$ we first need to find the prevertex z_k closest to z_0 . The prevertex z_k can be located by considering the images of radial lines; see the left picture. Once z_0 has been computed an SC map with the conformal centre at w_0 can be computed via a stable computation of a Möbius transformation of the prevertices.

So far we have not discussed the computation of the inverse map. If the forward map can be computed accurately, the zero of the function $f(z) - w_0$ can be found iteratively; see [10]. But, to compute the forward map accurately we need to know the position of z relative to the closest prevertex z_k accurately, see Lemma 2. A solution to this seemingly critical problem can be found by investigating the plots showing the images of radial lines connecting the origin to the prevertices. If w_0 is inside a sector defined by images of two consecutive radial lines we know that we can take as z_k one of the two corresponding prevertices. Computing the image of a further radial line determines which of the two prevertices should be used; see Figure 8. The precomputations involved can also give a good initial guess for the iteration. A crucial thing to note is that the iterative method should be used to find $z_0 - z_k$ and *not* z_0 .

For crowded maps the computation of the images of prevertices under the Möbius map h has also to be done with care. We find that

$$h(z_{j+1}) - h(z_j) = \left(\frac{1 - \bar{z}_0}{1 - z_0} \right) \frac{(z_{j+1} - z_j)(1 - |z_0|^2)}{(1 - z_j \bar{z}_0)(1 - z_{j+1} \bar{z}_0)},$$

for the Möbius automorphism of the unit disk h that maps z_0 to 0 and 1 to 1. In the example shown in Figure 8 $|z_0 - z_k| \approx 10^{-18}$, therefore $|z_0|$ is in double precision equal to 1 and the term $1 - |z_0|^2$ must be computed as follows:

$$1 - |z_0|^2 = -z_k(\overline{z_0 - z_k}) - \bar{z}_0(z_0 - z_k).$$

Similar expressions are needed for the remaining terms. The map with the conformal centre at w_0 is shown in the right part of Figure 8.

As the final example, we consider the computation of the conformal modulus of a generalized polygonal quadrilateral. This requires only little extra computation once the conformal map from the unit disk has been computed. If $w_{k_1}, w_{k_2}, w_{k_3}, w_{k_4}$ are the vertices of the quadrilateral it is sufficient to compute the polygonal image of the unit disk under the SC map defined by the four prevertices $z_{k_1}, z_{k_2}, z_{k_3}, z_{k_4}$ and four equal angle parameters $\alpha = 1/2$. The aspect ratio of the resulting rectangular polygon gives the conformal modulus of the generalized quadrilateral. We have computed the conformal modulus of the polygonal quadrilateral in Figure 5 with the two end edges of the Spiral polygon being the short edges of the quadrilateral. The 15 digits 31.9059755866729 computed in this way all agreed with the number obtained by using the map from a rectangle as implemented in the SC Toolbox. The conformal modulus of the quadrilateral indicated in the left part of Figure 8 has been computed using the two maps with two different

conformal centres computed before and these have agreed to 12 digits. This suggests that the Möbius map $h(\cdot)$ in the previous example, has been accurately computed. The approximate value of this modulus is 18.20544347160.

8 Conclusion

We have been able to adapt the standard methods for the computation of Schwarz-Christoffel maps from the unit disk to obtain high accuracy even in the presence of extreme crowding. With the use of a uniformly good guess, the computation is also reasonably fast. The fact that polygons like Figure 6 can be computed accurately without the need to consider a family of maps, change the canonical domain, or use domain decomposition has come as a surprise.

In a particular application, once the conformal map has been computed, further computations are often needed. The algorithms described in this paper also suggest and give strategies how to overcome the difficulties due to crowding in such further computations. As an illustration we have described how to deal with problems arising in some standard applications. In particular we have discussed the computation of the inverse map.

The initial guess provided by the CRDT is not the only choice we could have made. Bishop in [4, 5] describes another fast method for obtaining a uniformly good initial guess. The Zipper algorithm might also be used to provide such a guess; see [19]. Finally, stabilizing the SC maps from different canonical domains (half plane, strip, exterior of the unit disk) should be straightforward. The starting point would again be the implementation given in the SC Toolbox.

A Proof of Theorem 4

We start with a simple lemma on the evaluation of inner products.

Lemma 5. *Let $x, y \in \mathbb{C}^n$ and suppose that*

$$\tilde{x}_j = x_j(1 + \delta_j) \text{ and } \tilde{y}_j = y_j(1 + \gamma_j), \quad j = 1, \dots, n,$$

where

$$|\delta_j| \leq Cu, \quad |\gamma_j| \leq Du, \text{ and } \tilde{x}_j, \tilde{y}_j \in \mathcal{C}.$$

Then,

$$|fl(\tilde{x}^T \tilde{y}) - x^T y| \leq (C + D + n)|x|^T |y| u + O(u^2),$$

where $|x|$ denotes the vector with elements $|x_i|$.

Proof. Let us first define the partial sums

$$s_m = \sum_{j=1}^m x_j y_j, \quad \text{and} \quad \tilde{s}_m = fl \left(\sum_{j=1}^m \tilde{x}_j \tilde{y}_j \right).$$

We proceed by induction. Note that terms of size $O(u^2)$ are ignored. The base case $m = 1$ of induction is clear. Now assume that

$$|\tilde{s}_m - s_m| \leq (C + D + m) \sum_{j=1}^m |x_j y_j| u.$$

Then for some $|\delta| < u$

$$\begin{aligned}
|\tilde{s}_{m+1} - s_{m+1}| &= |(\tilde{s}_m + fl(\tilde{x}_{m+1}\tilde{y}_{m+1}))(1 + \delta) - s_{m+1} - x_{m+1}y_{m+1}| \\
&\leq (C + D + m) \sum_{j=1}^m |x_j y_j| u + \sum_{j=1}^m |x_j y_j| u + (C + D + 2)|x_{m+1}y_{m+1}| u \\
&\leq (C + D + m + 1) \sum_{j=1}^m |x_j y_j| u + (C + D + 2)|x_{m+1}y_{m+1}| u \\
&\leq (C + D + m + 1) \sum_{j=1}^{m+1} |x_j y_j| u,
\end{aligned}$$

and hence the result follows by induction. \square

Next we give a proof of Theorem 4.

Proof. We shall give the proof in stages. All the numbers $\delta_i \in \mathbb{C}$ are such that $|\delta_i| < u$; these will be reused, i.e. their value will change from stage to stage. The numbers γ_i will be bounds computed at each stage that are to be used in later computations. We begin by considering the computations of $\tilde{\nu}_2 - \tilde{\nu}_1$ and $\tilde{\nu}_1 + \tilde{\nu}_2$. Note that in the proof we omit terms of size $O(u^2)$. The proof of Lemma 2 should be sufficient to understand where these omissions took place.

$$\begin{aligned}
fl(\tilde{\nu}_2 - \tilde{\nu}_1) &= ((b - z_k)(1 + \delta_2) - (a - z_k)(1 + \delta_1))(1 + \delta_3) \\
&= (b - a) \left(1 + \frac{(b - z_k)(\delta_2 + \delta_3) - (a - z_k)(\delta_1 + \delta_3)}{b - a} \right) \\
&= (b - a)(1 + \gamma_1),
\end{aligned}$$

where, using (14) and (15)

$$|\gamma_1| \leq \frac{|b - z_k||\delta_2 + \delta_3| + |a - z_k||\delta_1 + \delta_3|}{|b - a|} \leq |\delta_1| + |\delta_2| + 2|\delta_3| \leq 4u.$$

Similarly

$$\begin{aligned}
fl(\tilde{\nu}_1 + \tilde{\nu}_2) &= ((a - z_k)(1 + \delta_1) + (b - z_k)(1 + \delta_2))(1 + \delta_3) \\
&= (a + b - 2z_k) \left(1 + \frac{(b - z_k)(\delta_2 + \delta_3) + (a - z_k)(\delta_1 + \delta_3)}{a + b - 2z_k} \right) \\
&= (a + b - 2z_k)(1 + \gamma_2),
\end{aligned}$$

where, using (14) and (15)

$$|\gamma_2| \leq \frac{|b - z_k||\delta_2 + \delta_3| + |a - z_k||\delta_1 + \delta_3|}{|a + b - 2z_k|} \leq \frac{|b - z_k||\delta_2 + \delta_3| + |a - z_k||\delta_1 + \delta_3|}{|b - a|} \leq 4u.$$

Next we find an error bound for the computation of $\frac{1}{2}((\tilde{\nu}_2 - \tilde{\nu}_1)\tilde{x}_l + \tilde{\nu}_1 + \tilde{\nu}_2)$.

$$\begin{aligned}
\tilde{\eta}_l &:= fl\left(\frac{1}{2}((\tilde{\nu}_2 - \tilde{\nu}_1)\tilde{x}_l + \tilde{\nu}_1 + \tilde{\nu}_2)\right) \\
&= \left(\frac{b - a}{2}x_l(1 + \gamma_1 + 3\delta_1) + \left(\frac{a + b}{2} - z_k\right)(1 + \gamma_2 + \delta_2)\right)(1 + \delta_3) \\
&= \frac{1}{2}((b - a)x_l + a + b - 2z_k) \left(1 + \frac{(b - a)x_l(\gamma_1 + 4\delta_4) + (a + b - 2z_k)(\gamma_2 + 2\delta_5)}{(b - a)x_l + a + b - 2z_k}\right) \\
&= \frac{1}{2}((b - a)x_l + a + b - 2z_k)(1 + \gamma_3)
\end{aligned}$$

where

$$\begin{aligned} |\gamma_3| &\leq \frac{|b-a|(|\gamma_1| + 4u) + |a+b-2z_k|(|\gamma_2| + 2u)}{|(b-a)x_l + a + b - 2z_k|} \\ &\leq \frac{8|b-a||u+12|\frac{a+b}{2} - z_k|u}{|b-a|} \leq 8u + 12u = 20u. \end{aligned}$$

To obtain to above estimate we have used the assumption (14) and the fact that $(a+b)/2$ and $((b-a)x_l + a + b)/2$ are points on the line connecting a and b .

Finally using Lemma 3 and Lemma 5,

$$\begin{aligned} &\left| fl \left(\frac{\tilde{\nu}_2 - \tilde{\nu}_1}{2} \sum_{l=1}^M \prod_{j=1}^N (\tilde{\eta}_l - \tilde{\omega}_j)^{\tilde{\beta}_j} \tilde{w}_l \right) - \frac{b-a}{2} \sum_{l=1}^M \prod_{j=1}^N (\zeta_l - z_j)^{\beta_j} \right| \\ &\leq \frac{1}{2}(C+M)|b-a| \sum_{l=1}^M \left| \prod_{j=1}^N (\zeta_l - z_j)^{\beta_j} \right| |w_l| u \\ &\leq A(C+M)u, \end{aligned}$$

where a simple but tedious calculation gives

$$\begin{aligned} Cu &\leq |\gamma_1| + 3N|\gamma_3| + (11N + 3 \max_j |\log |b - z_j|| N + 3)u \\ &\leq (71N + 3 \max_j |\log |b - z_j|| N + 7)u. \end{aligned}$$

The final result now follows from a single application of the triangle inequality. \square

References

- [1] L. Banjai. *Computation of Conformal Maps by Fast Multipole Method Accelerated Schwarz-Christoffel Transformation*. PhD thesis, Oxford University, England, 2003.
- [2] L. Banjai. Eigenfrequencies of fractal drums. *J. Comput. Appl. Math.*, 198(1):1–18, 2007.
- [3] L. Banjai and L. N. Trefethen. A multipole method for Schwarz-Christoffel mapping of polygons with thousands of sides. *SIAM J. Sci. Comput.*, 25(3):1042–1065, 2003.
- [4] C. J. Bishop. A fast approximation to the Riemann map. *Preprint*, 2003.
- [5] C. J. Bishop. Conformal mapping in linear time. *Preprint*, 2007.
- [6] R. T. Davis. Numerical methods for coordinate generation based on Schwarz-Christoffel transformations. In *4th AIAA Comput. Fluid Dynamics Conf., Williamsburg, VA, 1979*, pages 1–15, 1979.
- [7] J. E. Dennis, Jr. and R. B. Schnabel. *Numerical methods for unconstrained optimization and nonlinear equations*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1996.
- [8] T. A. Driscoll. A MATLAB toolbox for Schwarz-Christoffel mapping. *ACM Trans. Math. Soft.*, 22:168–186, 1996.
- [9] T. A. Driscoll. Algorithm 843: Improvements to the Schwarz-Christoffel toolbox for MATLAB. *ACM Trans. Math. Soft.*, 31:239–251, 2005.
- [10] T. A. Driscoll and L. N. Trefethen. *Schwarz-Christoffel Mapping*. Cambridge University Press, Cambridge, 2002.

- [11] T. A. Driscoll and S. A. Vavasis. Numerical conformal mapping using cross-ratios and Delaunay triangulation. *SIAM J. Sci. Comput.*, 19(6):1783–1803, 1998.
- [12] D. Gaier. Ermittlung des konformen Moduls von Vierecken mit Differenzenmethoden. *Numer. Math.*, 19:179–194, 1972.
- [13] J. B. Garnett and D. E. Marshall. *Harmonic measure*, volume 2 of *New Mathematical Monographs*. Cambridge University Press, Cambridge, 2005.
- [14] G. H. Golub and J. H. Welsch. Calculation of Gauss quadrature rules. *Math. Comp.*, 23:221–230, 1969.
- [15] P. Henrici. *Applied and computational complex analysis. Vol. 3*. John Wiley & Sons Inc., New York, 1986.
- [16] N. J. Higham. *Accuracy and stability of numerical algorithms*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, second edition, 2002.
- [17] L. H. Howell. Schwarz-Christoffel methods for multiply-elongated regions. In *Proc. of the 14th IMACS World Congress on Computation and Applied Mathematics, Atlanta*, 1994.
- [18] L. H. Howell and L. N. Trefethen. A modified Schwarz-Christoffel transformation for elongated regions. *SIAM J. Sci. Statist. Comput.*, 11(5):928–949, 1990.
- [19] D. E. Marshall and S. Rohde. Convergence of the Zipper algorithm for conformal mapping. *Preprint 2006*.
- [20] R. Menikoff and C. Zemach. Methods for numerical conformal mapping. *J. Comput. Phys.*, 36(3):366–410, 1980.
- [21] Z. Nehari. *Conformal mapping*. McGraw-Hill Book Co., Inc., New York, Toronto, London, 1952.
- [22] M. L. Overton. *Numerical computing with IEEE floating point arithmetic*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2001.
- [23] C. Pommerenke. *Boundary behaviour of conformal maps*, volume 299. Springer-Verlag, Berlin, 1992.
- [24] R. Schinzinger and P. A. A. Laura. *Conformal mapping: methods and applications*. Elsevier Science Publishers B.V., Amsterdam, 1991.
- [25] L. N. Trefethen. Numerical computation of the Schwarz-Christoffel transformation. *SIAM J. Sci. Statist. Comput.*, 1(1):82–102, 1980.
- [26] L. N. Trefethen. SCPACK user’s guide. *MIT Numerical Analysis Report 89-2*, 1989.
- [27] J. H. Wilkinson. *Rounding errors in algebraic processes*. Prentice-Hall Inc., Englewood Cliffs, N.J., 1963.