DOMAIN DECOMPOSITION METHODS FOR CONFORMAL MAPPING AND EIGENVALUE PROBLEMS

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Tobin Allen Driscoll

May 1996

© Tobin Allen Driscoll 1996

ALL RIGHTS RESERVED

DOMAIN DECOMPOSITION METHODS FOR CONFORMAL MAPPING AND EIGENVALUE PROBLEMS

Tobin Allen Driscoll, Ph.D.

Cornell University 1996

Domain decomposition is widely used in the numerical solution of elliptic boundary value problems. It is appealing in part because of improved efficiency and straightforward parallelization. Conceptually, domain decomposition often exploits a natural feature of elliptic problems: data at one point may have an exceedingly weak influence on the solution at a far-removed point.

The application of domain decomposition to other elliptic problems is less fully developed. One such area is numerical conformal mapping. We introduce the SC Toolbox for MATLAB, an interactive graphical software package for the Schwarz–Christoffel mapping of polygons. The Toolbox can be used for interior and exterior mapping from several fundamental domains. Elongations in the polygon lead to crowding, in which the preimages of affected vertices are exponentially close together.

Such regions are candidates for decomposition. We describe CRDT, an overlapping subdomain method developed with Vavasis for numerical Schwarz–Christoffel

mapping. The method uses Delaunay triangulation to decompose the polygon into overlapping quadrilaterals, which in turn define cross-ratios that form the basis of a nonlinear system. Each quadrilateral induces an embedding of the prevertices so that locally, the map can be computed accurately. Apparently CRDT can deal with any degree of crowding, as is demonstrated by examples.

Another application in conformal mapping is in Symm's integral equation. An important feature of existing software for Symm's equation is the efficient treatment of corner singularities. Careful generalization to multiple domains allows this treatment to be preserved and extended. A nonoverlapping formulation leads to a linear system that is ideal for Schur complementation. The resulting method asymptotically requires a fraction of the single-domain work and is easily parallelized.

We also consider a domain decomposition algorithm for the Laplace eigenvalue problem on polygons. The method, an improvement on one described by Descloux and Tolley, searches for the matching of Fourier–Bessel expansions at each corner to locate eigenvalues. We apply the algorithm to the "isospectral drums" discovered by Gordon, Webb, and Wolpert to find 25 eigenvalues to 12 digits. The method is far more accurate and efficient than standard methods for this problem.

BIOGRAPHICAL SKETCH

Tobin A. Driscoll grew up in Chadds Ford, Pennsylvania, graduating in 1987 from Unionville High School. He then attended Pennsylvania State University in State College, PA, on a Braddock Scholarship, finishing in 1991 with B.S. degrees in physics and in mathematics with honors. Driscoll became a graduate student at the Center for Applied Mathematics at Cornell University in Ithaca, New York, as a National Science Foundation and A. D. White Fellow. In 1993 he received a Master's in applied mathematics from Cornell. After receiving his Ph.D. in applied mathematics in 1996 under the supervision of Lloyd N. Trefethen, Driscoll will become an NSF Mathematical Sciences Postdoctoral Research Fellow at the University of Colorado at Boulder.

In loving memory of Robert Allen Driscoll.

ACKNOWLEDGEMENTS

This thesis would not have been possible without the guidance and support of Nick Trefethen. In addition to sharing funding and his vast scientific knowledge and ability, Nick fastidiously kept my interests as top priority in my own work and spent a great deal of effort promoting those interests on his own time as well. I have no doubt that my graduate experiences and future professional opportunities have been tremendously enhanced through my association with him. I also consider him a good friend.

Much of the work in this thesis originated from discussions and collaborations with Nick. The material in Chapter 3 was based on close collaboration with Steve Vavasis, who generously shared his original idea and early development. An observation made by Steve also led to the algorithmic improvement described in Chapter 5. I would like to thank Martin Gutknecht, David Hough, and Olof Widlund for answering my queries regarding `CONFPACK` and domain decomposition. I am also grateful to David Webb, Peter Doyle, Jean Descloux, Srinivas Sridhar and Arshad Kudrolli for their cooperation and helpful pointers during the isospectral drum project. I thank Nick, Steve, and Lars Wahlbin for their service on my committee.

Financially this work was supported in part by a National Science Foundation Graduate Fellowship, NSF Grant DMS-9116110, and DOE grant DE-FG02-94ER25199.

I gratefully acknowledge the indispensable friendship and technical assistance I have received from the CAMsters, including Dave Bond, Aaron Deever, Rob Ghrist, Patty Hough, Rob Manning, Jen Mohlenhoff, Kaila Patel, Dolores Pendell, Katy Simonsen, and Laurel Stell. I am especially grateful to Jeff Baggett and Kim-Chuan Toh, who as my academic siblings provided countless hours of fellowship, advice, and support.

Most of all, I thank my family for twenty-seven years of love, support, and belief in me, and my wife Jennie, who has faithfully shared all the frustrations, triumphs, and long winters, and whose steadfast love makes all things better.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Domain decomposition

An essential feature of an elliptic partial differential equation (PDE) is that its solution at any point depends on data at all the other points of a connected domain. This can make the numerical solution of an elliptic problem time-consuming and serial. However, while the influence of a given point is nonzero globally, it can be extremely weak over large portions of the domain. This fact is illustrated in Figure 1.1. At the top left of the figure is a region consisting of two disjoint squares and the locations of unknowns in a sample finite difference discretization. Below this is the sparsity pattern for the discrete Laplace operator. This matrix has a natural block diagonal structure: one block for each disjoint square. At the bottom left is the sparsity pattern for its inverse. Each diagonal block fills in completely, but the overall block diagonal structure is preserved. This structure is simply a reflection of the fact that the Laplace problems on the two squares can be solved independently. On the other hand, at the top right of Figure 1.1 we consider a region in which the squares are connected by a narrow channel. The resulting discrete Laplacian

still has a block structure corresponding to the squares and the channel, but there are just a few elements off the block diagonal, as can be seen in the close-up at the middle right. These few elements are enough to completely fill in the inverse of the matrix—hence the global dependence of the solution at each unknown on all the other unknowns. However, as the graph at the bottom right demonstrates, the block off-diagonals have mostly exponentially small elements. This effect, typical when a region has long, thin components, is a feature of the continuous problem, as can be seen from the Green's function.

When two subregions influence one another only weakly, one can hope to solve the problems on the subregions nearly independently, with minimal interaction. This observation is one of the fundamental ideas behind *domain decomposition*, a major topic in numerical methods for PDEs [14, 53, 73]. While many variations exist, domain decomposition methods generally fall into two categories: overlapping, in which communication between subdomains is achieved through shared regions, and nonoverlapping, in which the communication is through a shared interface of lower spatial dimension. Roughly speaking, overlapping methods are usually associated with iterative solutions, while nonoverlapping methods are more likely to be embedded in direct solvers.

Domain decomposition methods are often more efficient than global methods, because the cost of solving elliptic problems usually grows superlinearly with the problem size, and the subproblems can often be solved in parallel, if desired. The greatest gains are for those regions that have long, narrow parts, or parts on substantially different scales. Yet domain decomposition is more than a computational device. There are situations where the decomposition reflects a natural means of expressing the problem. For example, a composite structure is most conveniently expressed in terms of its component elements and a description of how those elements join together. It may be much more efficient or convenient to solve homogeneous

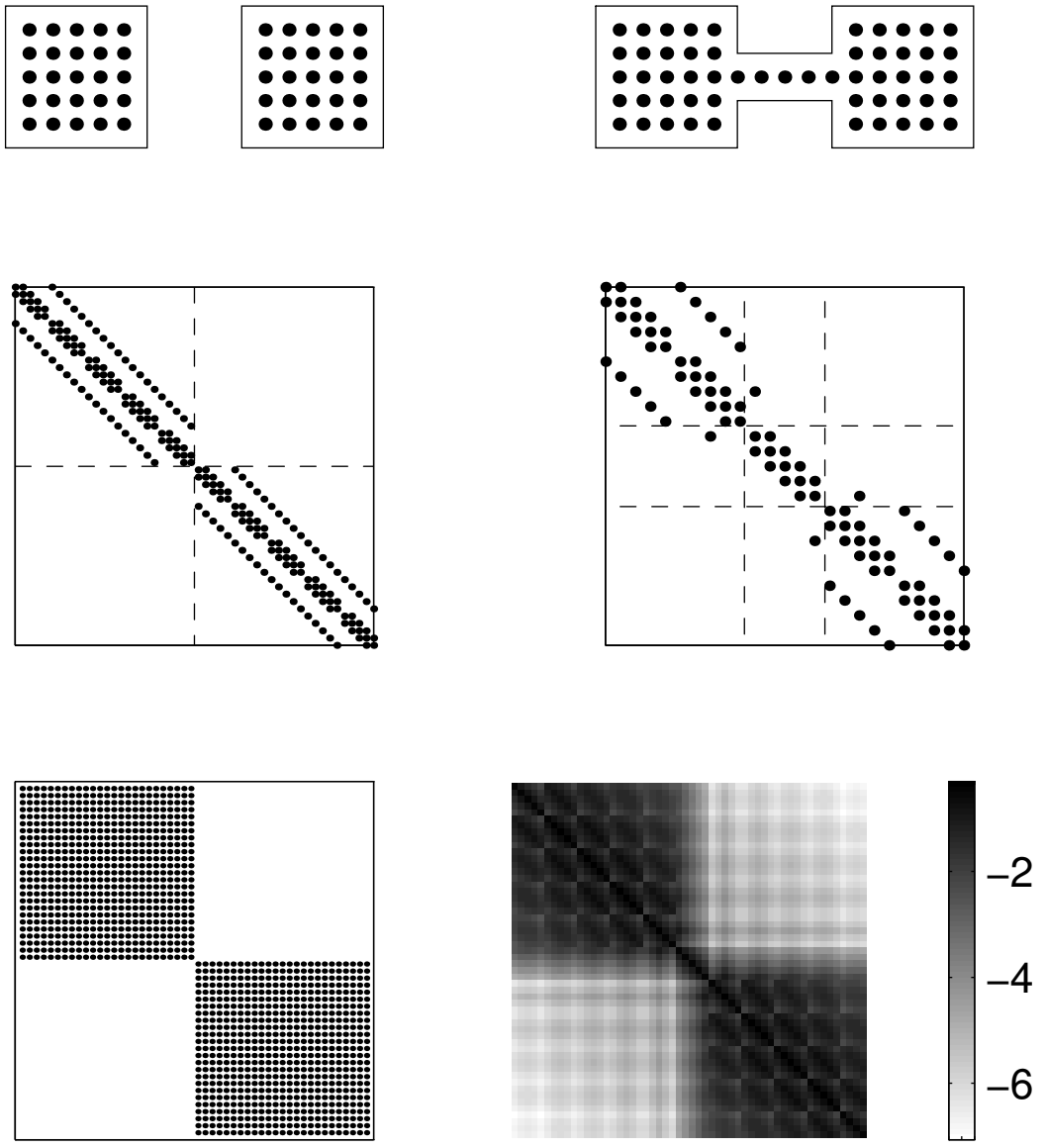


Figure 1.1: Nonzero but weak dependence in elliptic problems. At the top are two regions and points in typical discretizations. In the middle are sparsity patterns for the discrete Laplacians; the graph on the right is restricted to unknowns in and near the channel. At the bottom are views of the inverses of these operators; on the left is a sparsity pattern, and on the right is a graph of the $\log_{10}$ of the magnitude of the elements.

problems within each component and patch the results together than to attempt the global solution immediately. Even when there is no such direct analogy, one may find a decomposition for which the subproblems are qualitatively simpler.

While a great deal has been done with domain decomposition for linear boundary value problems, much less is known about many other applications. Surprisingly few methods exist even for the common linear eigenvalue problem, which arises in acoustics, electromagnetism, heat conduction, queueing theory, and quantum physics. Even less appears to have been done for conformal mapping, which has applications in potential theory and grid generation. In this thesis we shall explore some new or revised domain decomposition methods for these problems. In each case, the domain decomposition either reflects or reveals some interesting aspect of the special structure of the problem.

1.2 Summary of this thesis

This thesis deals with various numerical methods, mostly related to domain decomposition methods for conformal mapping and eigenvalue problems. The emphasis is on algorithms and numerical experimentation rather than theoretical analysis.

We now briefly describe our four main contributions.

A Matlab toolbox for numerical Schwarz–Christoffel mapping
Chapter 2 describes the implementation and use of the Schwarz–Christoffel Toolbox (SC Toolbox) for MATLAB. The SC Toolbox is based largely on algorithms that have previously been developed by Trefethen and others for several variations of Schwarz–Christoffel mapping, i.e., conformal mapping to polygons. The Toolbox features an interactive and graphical interface to drawing polygons, solving for the unknown parameters, and visualizing the resulting conformal maps. The

phenomenon of exponentially weak dependence within the solution is manifested as *crowding*, in which the preimages the vertices of elongated polygons are exponentially close together. Crowding is the major limitation of the methods of the SC Toolbox.

An overlapping algorithm using cross-ratios for numerical Schwarz–Christoffel mapping

Chapter 3 describes a new algorithm for Schwarz–Christoffel mapping, developed jointly with S. A. Vavasis, based on an overlapping domain decomposition of the polygon. The decomposition is based on quadrilaterals found in a Delaunay triangulation of the polygon, and the quadrilaterals are used to define *cross-ratios* as the fundamental unknowns. Even in the presence of severe crowding, the cross-ratios allow an accurate representation of the prevertices and the decomposition allows locally accurate evaluation of the map. The resulting algorithm can accurately solve problems from potential theory for arbitrarily elongated regions.

A nonoverlapping method for Symm's equation for conformal mapping

In Chapter 4 we study Symm's method for computing conformal maps to regions bounded by piecewise analytic curves. Symm's method is an indirect boundary element method for Laplace's equation with Dirichlet data. Although domain decomposition has been applied to many boundary integral problems, apparently it has not previously been studied within the context of conformal mapping. A straightforward nonoverlapping version of Symm's equation changes the meaning of the unknown quantity in the integral equation, introducing inelegant complications in the representation and use of the solution. Instead, a nonoverlapping algorithm for a related function leads to a system of integral equations of the second kind for which the unknown on the original boundary retains its meaning. The decomposition allows efforts to be focused on interesting subregions. We present the algorithm

and numerical experiments demonstrating its efficiency.

A high-accuracy nonoverlapping method for the Laplace eigenvalue problem on polygons

Chapter 5 examines the simplest linear eigenvalue problem, that of finding the eigenvalues of the 2D Laplace operator with homogeneous Dirichlet boundary conditions on a polygon. Our work is an improvement on a method previously introduced by Descloux and Tolley. An eigenfunction can be expanded about each corner in terms of a Fourier-Bessel series that exactly satisfies the interior equation and the original boundary conditions. Thus the original polygon can be decomposed into nonoverlapping subdomains, each with one of the original corners, and the eigenvalue problem reduces to finding expansions that match at the interfaces. This algorithm has been applied to the "isospectral drums" discovered by Gordon, Webb, and Wolpert, resulting in the first high-accuracy determination of these regions' spectrum.

Chapter 2

A Matlab toolbox for numerical Schwarz-Christoffel mapping*

Conformal mapping has long been an important topic in complex analysis, with applications in many fields of physics and mathematics. Historically, one of the major limitations of these applications was the difficulty of finding analytical formulas for conformal maps of general regions.

Even a region as simple as a polygon poses a serious problem. In the 1860s, H. A. Schwarz and E. B. Christoffel independently discovered a formula for mapping a half-plane to the interior of a polygon. The Schwarz–Christoffel formula can be derived merely by accounting for the singularity necessarily encountered at each pre-image of a polygon vertex (each *prevertex*). However, the locations of these prevertices are not known in closed form. Furthermore, the formula consists of a generally intractable contour integral, so it remained of mostly theoretical interest for a century.

*Portions reprinted with permission from T. A. Driscoll, "A MATLAB Toolbox for Schwarz–Christoffel mapping," *ACM Trans. Math. Soft.*, to appear, 1996. Copyright 1996 by the Association for Computing Machinery. All rights reserved.

The computer has brought practical conformal mapping within reach. Numerical conformal mapping has long been a topic in numerical analysis [30, 39, 59, 83], in which time variants of Schwarz–Christoffel mapping have received special attention [19, 20, 27, 28, 40, 47, 71, 74, 81, 86]. Part of the reason for the popularity of Schwarz–Christoffel mapping is that polygons are an important class of regions in applications; additionally, they can be used in principle to approximate any simply connected planar region. But another aspect of the appeal is that Schwarz–Christoffel maps can be computed very quickly to high accuracy, especially if the numerical method handles the corners intelligently.

In this chapter we introduce the latest addition to Schwarz–Christoffel mapping software: the Schwarz–Christoffel Toolbox (SC Toolbox) for MATLAB. The Toolbox provides an interactive, graphical interface to numerical routines for Schwarz–Christoffel mapping based on generalizations of methods first developed by Trefethen [81, 84]. The chapter begins with a description of the basic Schwarz–Christoffel formula and its variations, and a summary of the numerical methods that are used in the Toolbox. Then we summarize the structure and syntax of the Toolbox and provide examples and simple applications. Finally, in anticipation of the new method we shall introduce in Chapter 3, this chapter ends with a discussion of the limitations of the Toolbox, especially the "crowding phenomenon."

2.1 The Schwarz–Christoffel formula

Suppose a polygon P has complex vertices $w_1, \ldots, w_n$, given in counterclockwise order. Some of the vertices may be infinite. To each vertex w_k corresponds an exterior turning angle $-\beta_k\pi$, or equivalently, an interior angle $(1-\beta_k)\pi$. If $w_k \neq \infty$, then $-1 < \beta_k \leq 1$; otherwise, $-3 \leq \beta_k \leq -1$. See Figure 2.1 for an illustration.

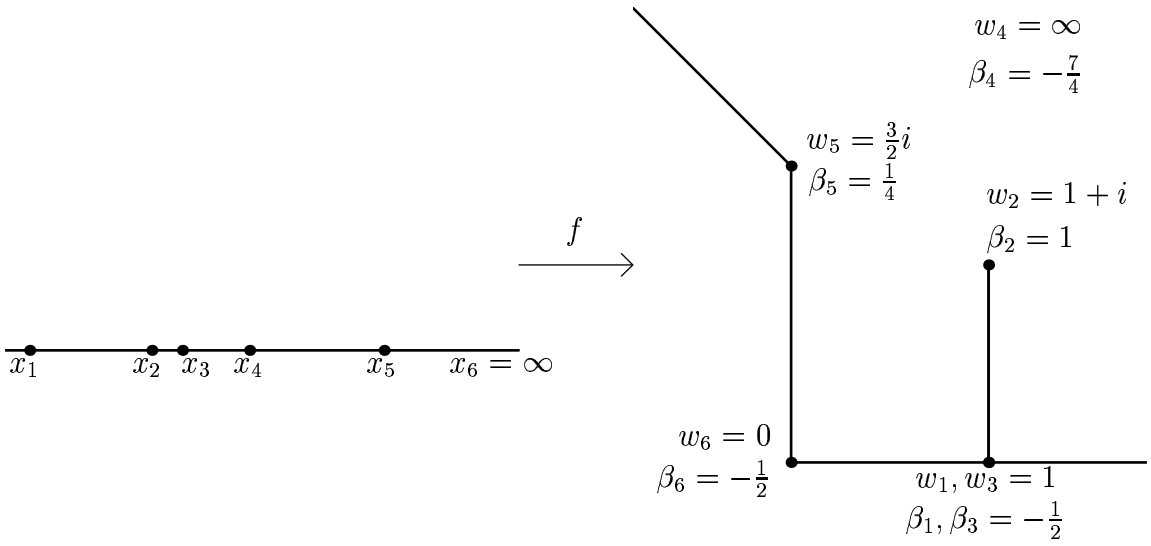


Figure 2.1: An unbounded polygon with $n = 6$.

Note that

$$\sum_{k=1}^{n} \beta_k = -2. \tag{2.1}$$

Let $\mathbf{C}^+$ denote the open complex upper half-plane, and define f on $\mathbf{C}^+$ by

$$f(z) = A + B \int_0^z \prod_{k=1}^{n-1} (s - x_k)^{\beta_k} \, ds \tag{2.2}$$

for some real $x_1, \ldots, x_{n-1}$ satisfying

$$x_1 < x_2 < \cdots < x_{n-1} < x_n = \infty \tag{2.3}$$

and complex constants A and B. Equation (2.2) is the *Schwarz–Christoffel formula*. It is not hard to see that f is analytic in $\overline{\mathbf{C}^+} \setminus \{x_1, \ldots, x_{n-1}\}$ and can be extended continuously to $\overline{\mathbf{C}^+}$ at each x_k for which $\beta_k > -1$ [38].

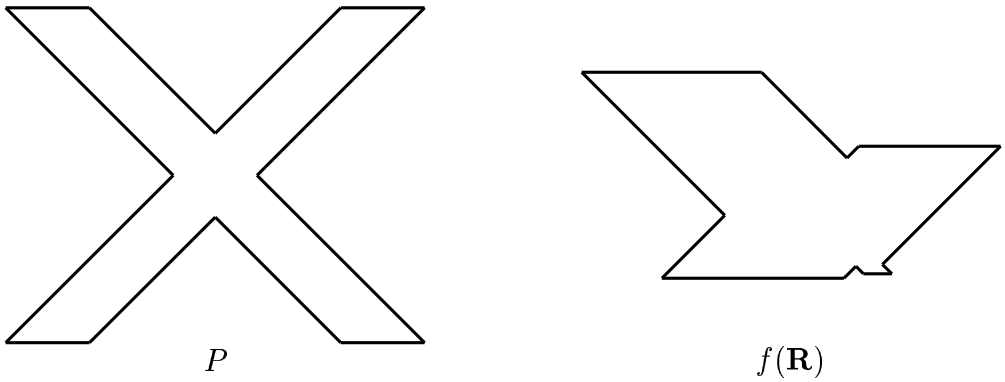


Figure 2.2: The effect of arbitrary prevertices. The polygon on the left is given; the one on the right is the image of the real line under (2.2), for arbitrarily chosen $x_1, \ldots, x_6$ satisfying (2.3). The angles match, but the side lengths differ.

The form of (2.2) can easily be understood by considering the derivative f'. As we let z traverse the real line rightward from $-\infty$, term k of the product in f' induces a jump at $z = x_k$ of $-\beta_k\pi$ in the complex argument of f', which is the tangent to the boundary of the image $f(\mathbf{R})$. Clearly, we want x_k to map to the kth vertex of the polygon, and is therefore we call x_k a *prevertex*. The Schwarz–Christoffel formula is nothing more than accounting for the effect of each corner of P on the image of $\mathbf{R}$.

The locations of the prevertices on the real line are crucial. The formula ensures that the angles of the image polygon match those of P, regardless of the choice of the x_k. But in general the side lengths will be wrong, as in Figure 2.1. Clearly we cannot choose the prevertices and constants A and B (collectively, the *accessory parameters*) arbitrarily. However, the Riemann mapping theorem guarantees that there is some conformal map from $\mathbf{C}^+$ to Ω, the interior of the region bounded by P, and the fundamental theorem of Schwarz–Christoffel mapping guarantees that an appropriate choice of parameters will yield such a map:

Theorem 2.1. *Let Ω be a simply connected region bounded by a polygon having*

(possibly infinite) vertices $w_1, \ldots, w_n$ *and exterior turning angles* $-\beta_1\pi, \ldots, -\beta_n\pi$. *Then every function which maps* $\mathbf{C}^+$ *conformally onto* Ω *can be expressed in the form* (2.2).

Proof. See, for example, [38], Thm 5.12e. □

In addition, any such map can be extended continuously to $\overline{\mathbf{C}^+}$ (except at preimages of infinity) as a map onto $\overline{\Omega}$. By considering Möbius transformations, we see that any three prevertices can be chosen arbitrarily. Since $x_n = \infty$, two other prevertices, say x_1 and x_2, may be fixed. Then the remaining prevertices, and the constants A and B, are determined uniquely by P.

The Schwarz–Christoffel formula is mathematically appealing, but not immediately practical. To compute the map, one must find the prevertices by solving a system of nonlinear equations—a problem which, except in a few special cases, is analytically intractable. This is the Schwarz–Christoffel *parameter problem.* Furthermore, the integral in (2.2) rarely has a simple closed form. Finally, it is usually impossible to invert f explicitly. For these reasons, calculations involving Schwarz–Christoffel maps must generally be done on a computer.

2.2 Variations of the formula

Composing (2.2) with standard conformal maps leads to variations of the Schwarz–Christoffel formula for mapping from other fundamental domains, many of which are outlined in [86]. The simplest such modification has the unit disk as domain. The prevertices then lie in counterclockwise order on the unit circle, and the resulting formula is nearly identical:

$$f(z) = A + B \int_0^z \prod_{k=1}^{n} \left(1 - \frac{s}{z_k}\right)^{\beta_k} ds. \tag{2.4}$$

Observe that there are now n rather than $n-1$ terms in the product of the integrand, and the prevertices $\{z_k\}$ lie on the unit circle. The constant A, which is the image of the origin, is known as the *conformal center*.

By considering the logarithm of the upper half-plane, we arrive at the transformation from a biinfinite strip $\{z : 0 \leq \operatorname{Im} z \leq 1\}$, which is especially useful when P is an infinite polygonal channel, such as might arise in a fluids problem, for example. In this case the formula becomes [47, 74, 89]

$$f(z) = A + B \int_0^z e^{\theta s} \prod_{k=1}^{n} \left[\sinh \frac{\pi}{2}(s - z_k)\right]^{\beta_k} ds, \tag{2.5}$$

where θ depends on the divergence angles at the ends of the channel, and the prevertices $\{z_k\}$ lie on both sides of the strip.

By further composing (2.5) with the Jacobi elliptic function sn, a rectangle may be used as the fundamental domain, which is appropriate when P is highly elongated in one direction. In this case four distinguished vertices of the polygon map to the corners of the rectangle, and the polygon is said to be a *generalized quadrilateral*. The aspect ratio of the rectangle, which is the *conformal modulus* of the generalized quadrilateral, cannot be specified in advance, but is determined by solving an appropriately constrained parameter problem on the strip (see [47] for details).

By modifying the usual derivation of (2.2) slightly, we can also find a function that maps the unit disk onto the exterior of P [38]:

$$f(z) = A + B \int_0^z s^{-2} \prod_{k=1}^{n} (s - z_k)^{\beta_k} \, ds. \tag{2.6}$$

The term s^{-2} reflects a choice of mapping the origin to the point at infinity. Thus, only a single prevertex may be chosen arbitrarily. Here the vertices are specified in counterclockwise order about the exterior of P, or clockwise about the interior, so that the sum in (2.1) is $+2$.

A different sort of generalization is the map to a circular-arc polygon (each side is an arc of a circle). This variant requires more fundamental changes to the Schwarz–Christoffel formula, which becomes an ordinary differential equation rather than an integral. There are also n new accessory parameters that have no obvious geometric interpretation. Although numerical solution is possible [10, 45], so far the methods used have not been fast or reliable.

2.3 Summary of the numerical algorithms

Any numerical method for implementing one of the Schwarz–Christoffel maps has to contend with the three issues outlined above: solving the parameter problem, computing integrals of the form in (2.2), and, if desired, inverting the map to find points in the fundamental domain given their images in the polygon. All these issues were dealt with by Trefethen [81, 85] for the disk and half-plane maps, and by Howell and Trefethen [47] for the strip and rectangle maps. The exterior map is very similar to the disk map.

We first consider the parameter problem. Recall that for the half-plane formula (2.2), x_1, x_2, and $x_n = \infty$ are fixed, and the remaining $n-3$ real prevertices must be determined. Because the angles of the resulting polygon are guaranteed to be correct for any set of prevertices, we must use the side lengths of P to derive $n-3$ real conditions. If all the vertices $\{w_k\}$ are finite, a natural set of conditions is [85]

$$\frac{\left|\int_{x_k}^{x_{k+1}} f'(s)\,ds\right|}{\left|\int_{x_1}^{x_2} f'(s)\,ds\right|} = \frac{|w_{k+1}-w_k|}{|w_2-w_1|}, \qquad 2 \le k \le n-2. \tag{2.7}$$

The affine constants A and B do not appear, and the equations are scaled by a length occurring in the problem. If all $n-3$ equations are satisfied, then $w_1, \ldots, w_{n-1}$ are

correctly located relative to one another, and w_n is then located by the intersection of the two sides adjacent to it. As a consequence, w_n may not be a finite vertex whose adjacent sides are collinear; i.e., β_n must not be 0 or 1.

If a vertex w_K is infinite, then equations $K-1$ and K of (2.7) are useless. In this case we replace them by the complex equation

$$\frac{\int_{x_{K-1}}^{x_{K+1}} f'(s)\,ds}{\int_{x_1}^{x_2} f'(s)\,ds} = \frac{w_{K+1} - w_{K-1}}{w_2 - w_1}. \tag{2.8}$$

Thus we require that there be no adjacent infinite vertices and that two consecutive finite vertices serve as w_1 and w_2.

Equations (2.7) give us a system of $n-3$ nonlinear equations in $n-3$ unknowns, but the unknowns are constrained by (2.3). A change of variables enforces the constraints implicitly:

$$y_j = \log(x_{j+2} - x_{j+1}), \qquad 1 \le j \le n-3. \tag{2.9}$$

Now we have an unconstrained $(n-3)\times(n-3)$ system of nonlinear equations that has a unique solution.

The nonlinear system is easily modified to suit variations of the fundamental domain. For the disk map, nothing need be changed. For the strip map, the ends of the strip are already constrained to map to the ends of the target channel, so only one z_k in (2.5) can be fixed. Now $n-1$ side-length conditions are needed, but there is no redundancy in this information because two real conditions are needed to locate one side of the channel with respect to the other. In the rectangle problem, the prevertices are still considered to be on the strip, but pairs of the pre-images of the rectangle corners are constrained to have the same real part on the strip, leading again to $n-3$ unknowns. Finally, for the exterior map, the image of the center of the disk is complex infinity, so there are $n-1$ unknowns. The $n-3$ equations of (2.7) are supplemented by requiring that the complex residue of f' be zero, which

implies

$$\sum_{k=1}^{n} \frac{\beta_k}{z_k} = 0. \tag{2.10}$$

We now turn to the problem of computing Schwarz–Christoffel integrals of the form (2.2). Numerically these are not trivial, because the integrand is singular at the prevertices, and these singularities are often at or near endpoints of the integration. (In fact, they are often exponentially nearby, as explained in the next section.) Trefethen [81] suggests a compound Gauss–Jacobi quadrature rule along straight-line paths between the endpoints. The selection of the Gauss–Jacobi method allows an exact treatment of endpoint singularities, and an adaptive compound rule deals with the influence of singularities close to the endpoints by requiring integration subintervals to be comparable in length to the proximity of the singularities. Howell [44] compares this method to various combinations of adaptive quadrature and singularity removal and found it to be satisfactory, if not superior, except possibly when precise error control is desired.

Krikeles and Rubin [54] recommend a more efficient approach to integration for the parameter problem when the fundamental domain is a disk. They note that if z_p and z_q are on the unit circle and $g(z)$ is a Schwarz–Christoffel integrand,

$$\left| \int_{z_p}^{z_q} g(s)\, ds \right| = \int_{\theta_p}^{\theta_q} |g(\theta)|\, |d\theta|.$$

Hence the side lengths in the nonlinear equations can be computed using real logarithms to evaluate the integrand, rather than the complex logarithms required by straight-line paths. Integrations along the real axis or a strip edge also require only real logarithms. When dealing with infinite vertices by inserting equations like (2.8), however, the integration paths must be modified to avoid passing through a prevertex of infinity, and complex logarithms are required in every fundamental domain.

If we wish to invert f, there are two natural strategies. One is to do a Newton iteration with the exact f'; alternatively, one can pose and solve a differential equation for $1/f'$. Trefethen [81] suggests a hybrid wherein the ODE is solved to get an approximate answer which is then improved by a few Newton steps.

2.4 Overview of the Schwarz–Christoffel Toolbox

An important trend in scientific computing has been the growing use of high-level, interactive environments. In numerical analysis and scientific computing, the dominant environment (as of this writing) is MATLAB.[1] MATLAB provides an interactive command-line interface, built-in visualization, and the elements of graphical user interfaces. At first MATLAB was used primarily for education and algorithm prototyping and development, but it is increasingly a major all-purpose computing environment.

The Schwarz–Christoffel Toolbox (SC Toolbox) for MATLAB is a suite of M-files designed for computing and visualizing Schwarz–Christoffel maps. The SC Toolbox is intended to be the successor to Trefethen's Fortran package SCPACK [84], the first and most influential public-domain software for numerical conformal mapping. The Toolbox incorporates the algorithm used in SCPACK, extended to fundamental regions other than the disk, as described in Section 2.3. It also adds graphical interfaces to input and manipulate polygons, and quick visualization of conformal maps.

The SC Toolbox currently works with the following types of Schwarz–Christoffel maps:

- Half-plane

[1] MATLAB is a registered trademark of The MathWorks, Inc.

- Disk
- Strip
- Rectangle
- Disk to polygon exterior

All of the functions are original and are written as M-files compatible with version 4.1 of MATLAB (version 4.0 under MS-Windows). Emphasis was placed on making the Toolbox easy and convenient to use.

The Toolbox, along with detailed documentation, is available via anonymous ftp, at the URL

```
ftp://ftp.cs.cornell.edu/pub/driscoll/SC-Toolbox/
```

Inquiries may be sent to the author at `driscoll@na-net.ornl.gov`.

2.5 Implementation issues

Each of the different types of Schwarz–Christoffel maps has an associated block of M-files, distinguished by a two-letter prefix. Each block has functions for solution of the parameter problem, computation of forward and inverse maps, and automated visualization. In addition, there are several functions for working with polygons, including one for drawing them with the mouse. Finally, most of the Toolbox's functions can be accessed through a graphical user interface (GUI). A demonstration of most of the Toolbox's basic capabilities appears in the next section.

The numerical methods used by the SC Toolbox are summarized in Section 2.3. For solution of the nonlinear system arising from the parameter problem, we chose the public domain package NESOLVE, which is Behrens' implementation of some

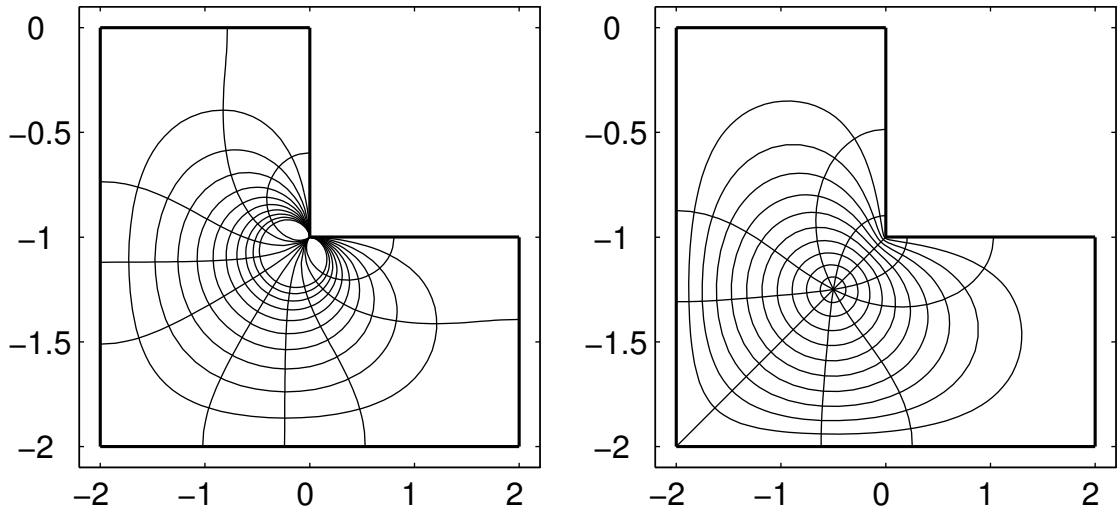
of the modules in [21]. The algorithm used is a Gauss–Newton trust region method with Broyden update of the Jacobian. The Broyden update is important because of the relatively high cost of function evaluations. Iterations are continued until the 2-norm of the nonlinear function vector is smaller than a user-adjustable tolerance. Experience indicates that the values of prevertices and function maps are generally at least as accurate as this tolerance, and often the maps are more accurate away from the singularities.

The only drawback of the SC Toolbox compared to SCPACK is execution speed. The experience of most MATLAB users is that the performance of M-files usually cannot match that of an equivalent function written in a compiled language. If speed were of utmost importance, the numerically most intensive routines could be rewritten in C or Fortran (possibly with the help of MATLAB's own C translator). However, we believe that a factor of 2 or 3 lost to coding in MATLAB is well worth the portability and clarity gained, in most applications.

2.6 Examples and applications

We now illustrate the capabilities of the SC Toolbox. The Toolbox is also distributed with several automated demonstrations accessible through the function `scdemo`. Complete details on using the package are given in a user's guide distributed with the software and in online help accessible via MATLAB's `help` and `lookfor` commands.

All of the polygons for the following figures were drawn by hand with the Toolbox's drawing function, `drawpoly`. This function allows the user to constrain vertices to a grid, quantize polygon angles and lengths, and place vertices at infinity.




```
>> [w,b] = drawpoly;
>> [z,c] = hpparam(w,b);
>> hpplot(w,b,z,c,10,0.8:.8:8)
```

```
>> [w,b] = drawpoly;
>> [z,c] = dparam(w,b);
>> ptsource(w,b,z,c,-.5-.5i)
```

Figure 2.3: Half-plane (left) and disk (right) maps for an L-shaped region. The half-plane plot is the image of ten evenly spaced vertical and ten evenly spaced horizontal lines with abscissae from -2.7 and 15.6 (chosen automatically) and ordinates from 0.8 to 8. The disk plot is the image of ten evenly spaced circles and radii in the unit disk. Below each plot is the MATLAB code needed to generate it.

2.6.1 Basic maps

Figure 2.3 shows the half-plane and interior disk maps for an L-shaped region. The plot on the left is the image of a regular rectangular mesh in the upper half-plane. The images of the vertical lines converge at one end to $w_n = f(\infty)$, and the horizontal ones converge there at both ends. The graph on the right is the image of a polar grid in the unit disk. Of course, all the intersections in both graphs are at right angles. Underneath each plot is a listing of lines that are entered at the MATLAB prompt or in a function file to create each picture. The listings have one line each for creating the polygon via the drawing tool, solving the parameter problem, and visualizing the result. The slightly different plotting command for the

disk case stems from requiring the conformal center to be at $-0.5 - 0.5i$. Each of the plots in this section is produced similarly, with possibly an extra command for manually selecting the axes limits. For the half-plane case, the parameter problem solution took 1.1 seconds and the plot about 15 seconds on a SPARC-10 workstation. The corresponding times for the disk were about 2.8 and 9.4 seconds. The parameter problem times correspond to the number of nonlinear function evaluations needed: 12 for the half-plane and 30 for the disk. The plotting time for the half-plane is longer because the program must figure out how far towards infinity it must go in three directions to produce accurate and smooth curves near w_n.

Figure 2.4 demonstrates the half-plane and disk maps for several polygons featuring infinite vertices and slits. Half-plane maps are suggestive of potential flows of fluids, while disk maps invite interpretation as electrostatic fields due to point charges.

The map from a strip is natural when the target region is a polygonal channel. Another interesting use is when a source or sink is desired at some finite vertex of a polygon. Figure 2.5 demonstrates these situations.

For reasons discussed in Section 2.7, rectangle maps are appropriate for regions which are elongated in one direction. Figure 2.6 displays the rectangle maps for two polygons for which neither the half-plane nor disk maps can be computed by the Toolbox. The conformal modulus of each polygon can be interpreted as the electrical resistance of a polygonal resistor [82].

Figure 2.7 shows the exterior maps for two polygons. From the Toolbox user's standpoint, there is little difference from the other maps. A natural interpretation of the plots is as equipotential and field flow curves for polygonal conductors. The region on the left is an early stage in the production of the fractal Koch snowflake and also appears in the `CONFPACK` documentation [42]. This polygon was drawn quite easily by hand using angle and length quantization. Because of the manifold

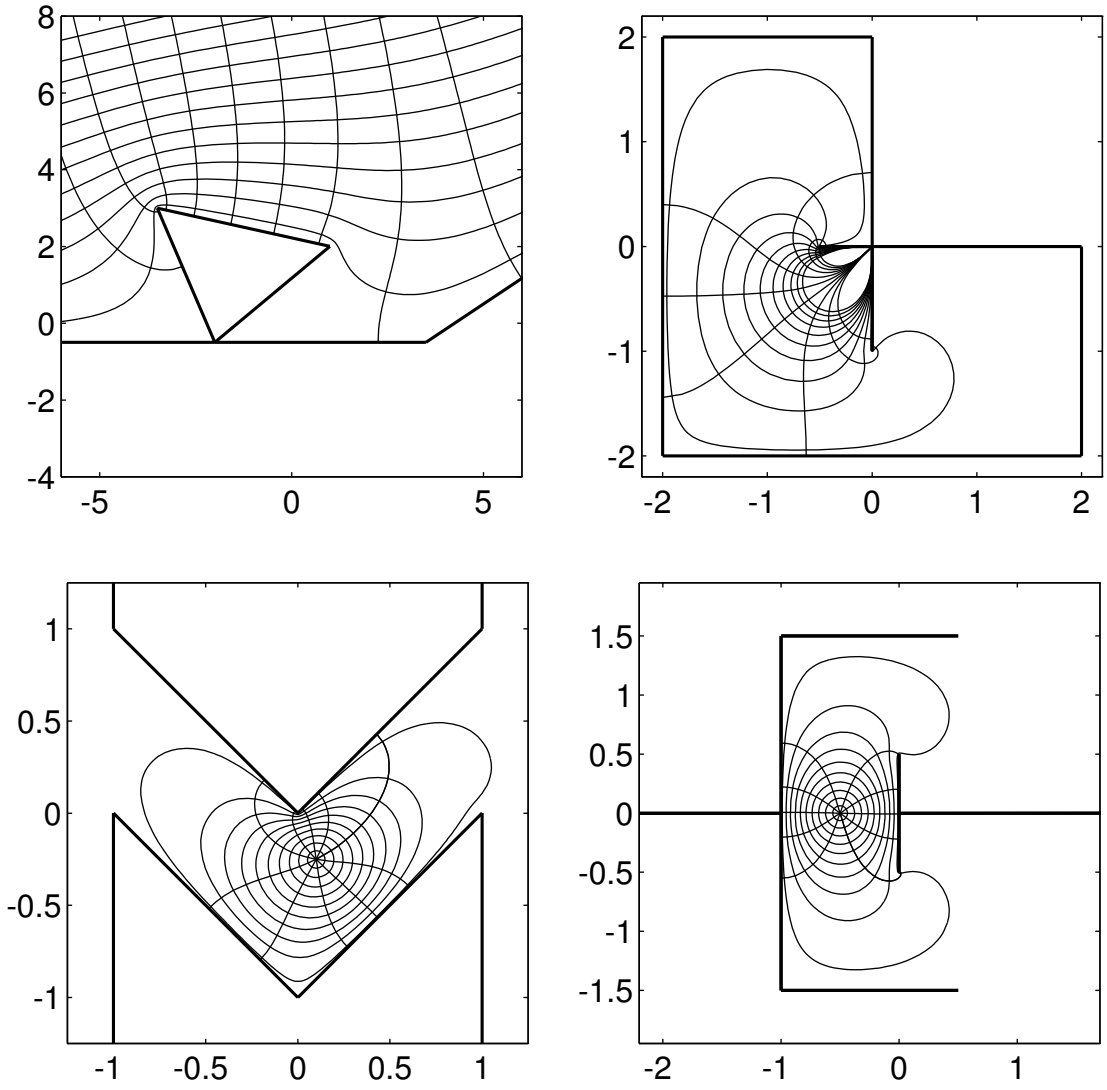


Figure 2.4: The half-plane (top) and disk maps (bottom) for several polygons. Except at top right, the regions are unbounded.

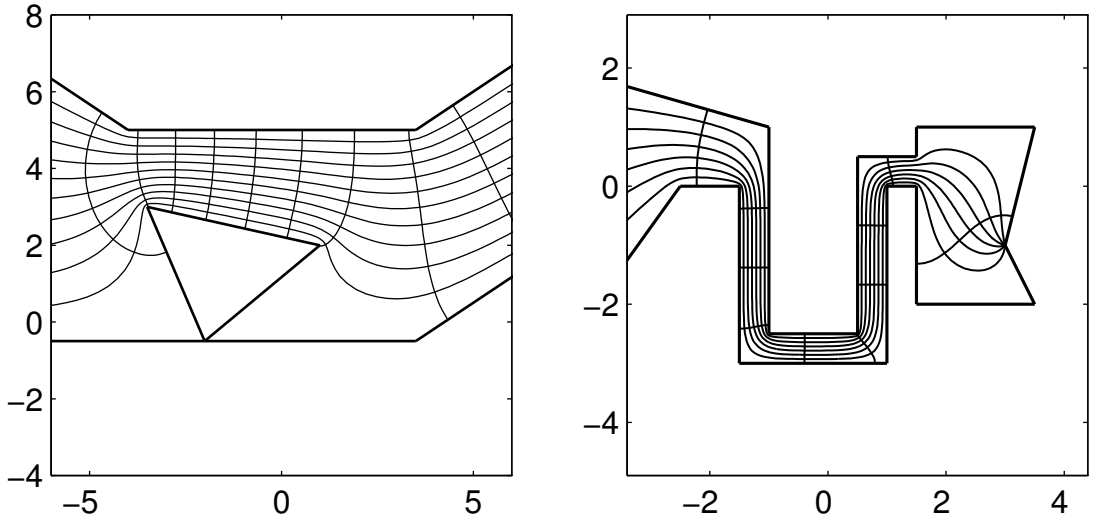


Figure 2.5: Maps from the infinite strip $0 \leq \operatorname{Im} z \leq 1$. On the left the ends of the strip map to the ends of the channel (compare to Figure 2.4), and on the right, one end of the strip maps to a finite point.

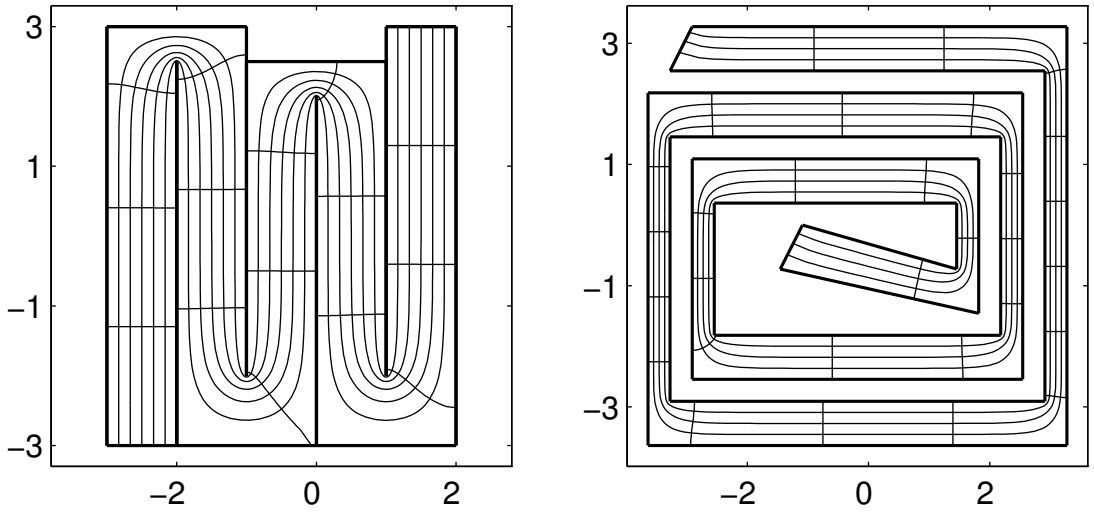


Figure 2.6: The rectangle map for two highly elongated regions. The curves are images of equally spaced lines in the interior of the rectangles. The conformal moduli of the regions are about 27.2 (left) and 91.5 (right).

symmetry of the region, the exterior map could be computed by solving an interior mapping problem for a sector of angle $\pi/6$ of the snowflake exterior and applying Schwarz reflection. This would reduce the effective number of vertices from 48 to 6, and in principle would speed up the solution to the parameter problem by a factor of thousands. However, using the built-in functions of the Toolbox for the full polygon, the parameter problem solution and the plot in Figure 2.7 each required about four minutes on a SPARC-10, so the potential gains are not large in an absolute sense. Moreover, the Toolbox also computes the exterior map for regions with no symmetry, as the graph on the right in Figure 2.7 demonstrates.

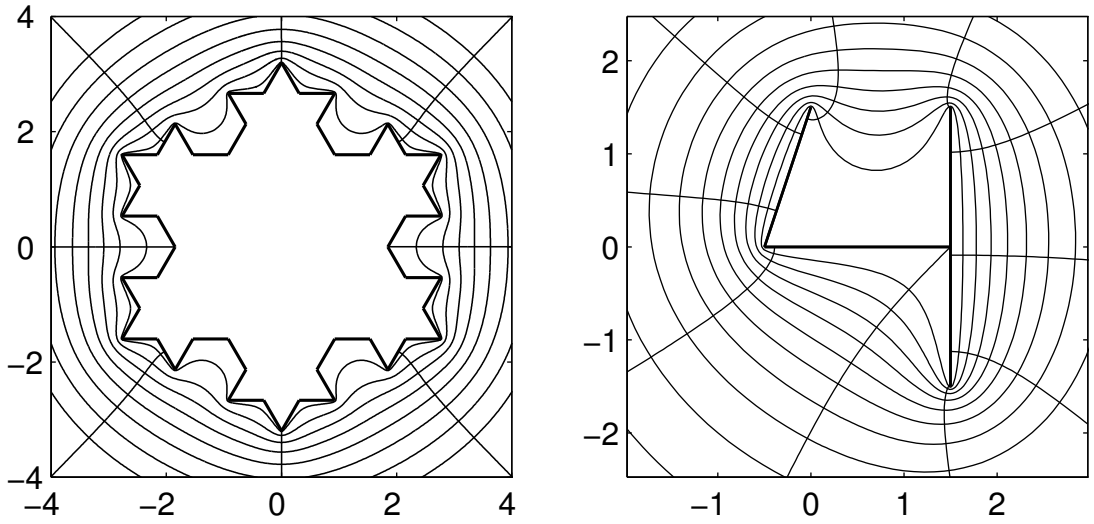


Figure 2.7: Maps from the unit disk to two polygon exteriors. The region on the right is the complement of three connected line segments.

2.6.2 Extensions to other domains

The Schwarz reflection principle was mentioned above as a means of speeding up computations for regions with reflective symmetry. The reflection technique can also be used to compute conformal maps for regions to which the Schwarz–Christoffel

formula does not directly apply. Two examples of this are exhibited in Figure 2.8. For periodic regions such as the one on the left, the overall map can be constructed

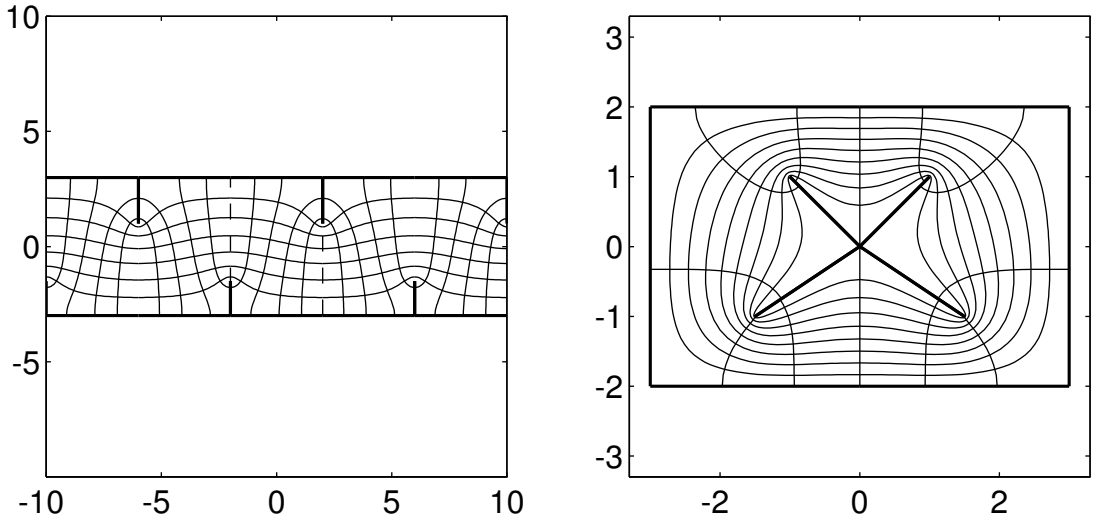


Figure 2.8: Maps computed by reflections. The region on the left is periodic with reflective symmetry at the dashed lines, and is mapped from a strip. The region on the right is doubly connected with an axis of symmetry, and is mapped from an annulus.

by computing the rectangle map for the fundamental region (shown in dashed lines) and reflecting and translating the results [27]. Since the Toolbox plotting routines return the coordinates of the plotted points upon request, this graph can be constructed with one `plot` statement per fundamental unit. Another interesting way to use reflection is for doubly connected regions with an axis of symmetry, as on the right in Figure 2.8. Because the entire region maps to an annulus, half of the region maps to half an annulus, which after a logarithm becomes a rectangle. Thus the displayed plot is the result of a Toolbox rectangle map and a reflection.

Many of the variations considered so far have centered on transformations of the fundamental domain. Transformation of the image domain also leads to interesting results, such as maps to gearlike domains [67]. An example of a gearlike domain is

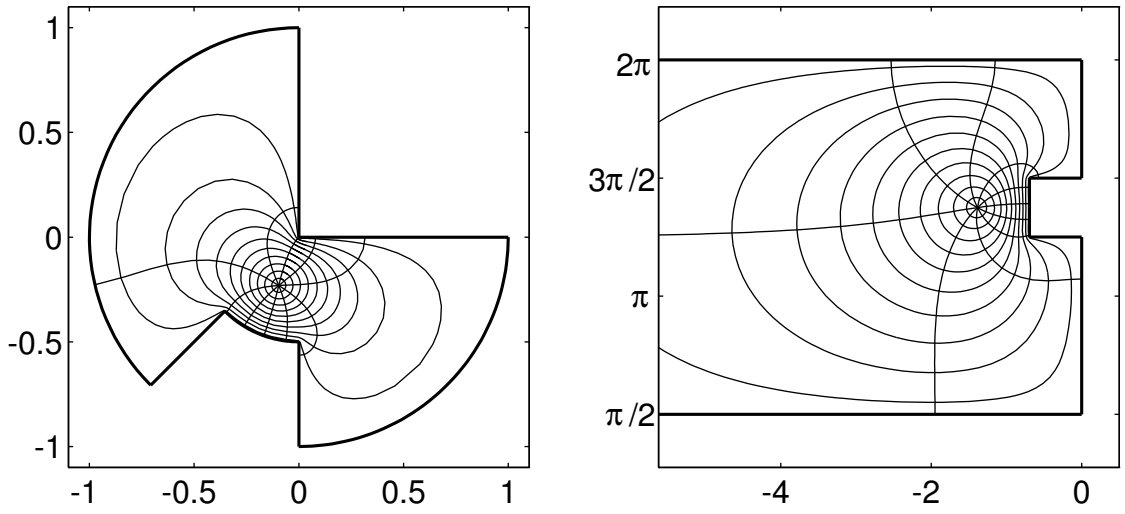


Figure 2.9: Map from the unit disk to a gearlike domain (left). The logarithms of these curves are shown on the right.

shown on the left in Figure 2.9. The simplest case is when one of the notches goes to the origin. Upon taking a logarithm, the region becomes a notched semi-infinite strip, shown on the right, for which a standard interior map can be computed.

2.6.3 Applications of exterior maps

Many applications of conformal mapping stem from the well-known fact that solutions to Laplace's equation are preserved by such maps. A classical example for exterior mapping is shown in Figure 2.10: streamlines for two flows past a crude "airfoil." The method is outlined in [38], pages 358–367. The situation on the left is flow with zero circulation, for which one stagnation point is on top of the airfoil. This plot is made by considering the model problem of flow exterior to the real interval $[-1, 1]$. The streamlines of the model problem are horizontal lines, and can be mapped to the unit disk by an inverse Joukowski map and thence to the airfoil exterior via the Toolbox. On the right is a flow with negative circulation meant to

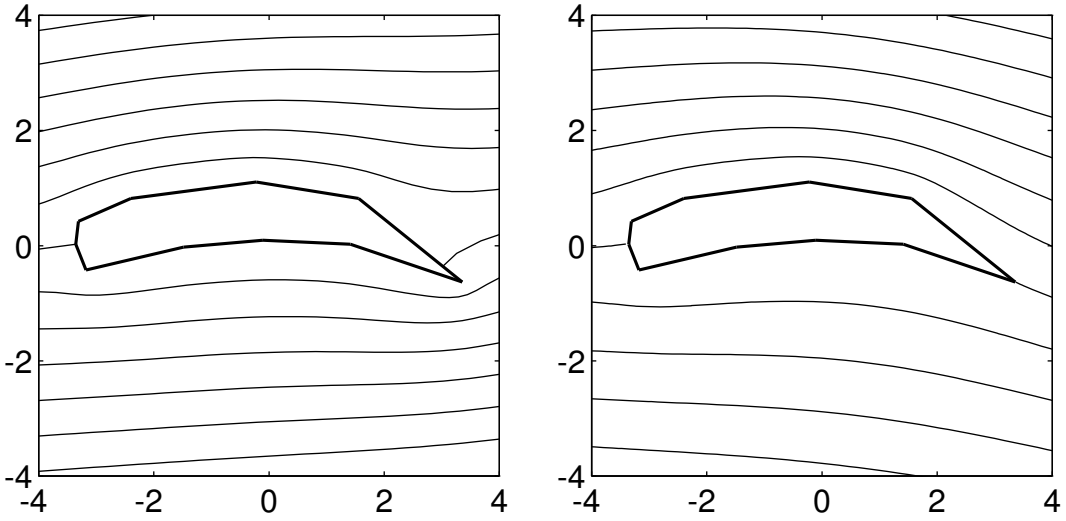


Figure 2.10: On the left, non-circulating potential flow past an "airfoil." On the right, flow past the same airfoil with negative circulation.

illustrate the Kutta condition. In this case the model flow is the exterior of the unit circle, which was the intermediate plane in the previous case. The stagnation points in the model plane are prevertices of the airfoil, and from this the streamlines can be computed in the model plane (via MATLAB's `contour` command) and mapped to the airfoil exterior. One potential pitfall of applying this technique to real airfoils would be the introduction of singularities in the flow at artificial corners. However, we expect that with careful use, polygonal approximation may often produce acceptable results.

Another interesting use of the exterior map arises in the area of approximation theory. Let Ω be a bounded, connected region whose complement Ω^c is simply connected in the extended plane. There is a unique conformal map ϕ which takes Ω^c onto the exterior of the unit disk, fixes the point at infinity, and has positive real derivative there. When Ω is the interior of a bounded polygon, ϕ is just the inverse of a Schwarz–Christoffel exterior map. If ϕ is renormalized to be 1 at the

origin, then $[\phi(z)]^m$ is minimax on Ω among origin-normalized functions with a pole of order m at infinity [34].

A natural way to approximate ϕ^m is by *Faber polynomials*. The mth degree Faber polynomial F_m for Ω is given by the polynomial part of the Laurent expansion of ϕ^m at infinity. For polygonal domains, the Faber polynomials can be computed easily via a recurrence relation [76], as is implemented in the Toolbox function `faber`. By construction, Faber polynomials are "nearly minimax" on the region Ω. This property has been used as a justification for choosing Faber polynomials based on approximate spectra as iteration polynomials in Krylov subspace iterations for the solution of linear systems [76].

In Figure 2.11 we plot the lemniscates $\{z : |F_m(z)| = 1\}$ for a particular polygon and various m. Because F_m is an analytic projection of ϕ^m, which is identically one in modulus on the polygon, the lemniscates approximate the polygon. The re-entrant corners of the exterior region are resolved first, with the "deadwater" regions being most difficult. Note the tendency of the lemniscates to oscillate about the polygon edges. This is related to equioscillation, familiar from Chebyshev approximation, and to the near-minimality of the Faber polynomials.

2.7 Limitations of the Toolbox

The most serious limitation of the SC Toolbox, and perhaps of numerical conformal mapping in general, is the *crowding phenomenon* [59]. Consider the rectangle R with corners at $-K+iK'$, $-K$, K, and $K+iK'$, where K and K' are the complete elliptic integrals of the first kind with parameters m and $1-m$, respectively. The Jacobi elliptic function $\mathrm{sn}(z|m)$ maps R conformally onto the upper half-plane, sending the corners to $-m^{-1/2}$, -1, 1, and $m^{-1/2}$, respectively. As $m \to 0$, it is easy

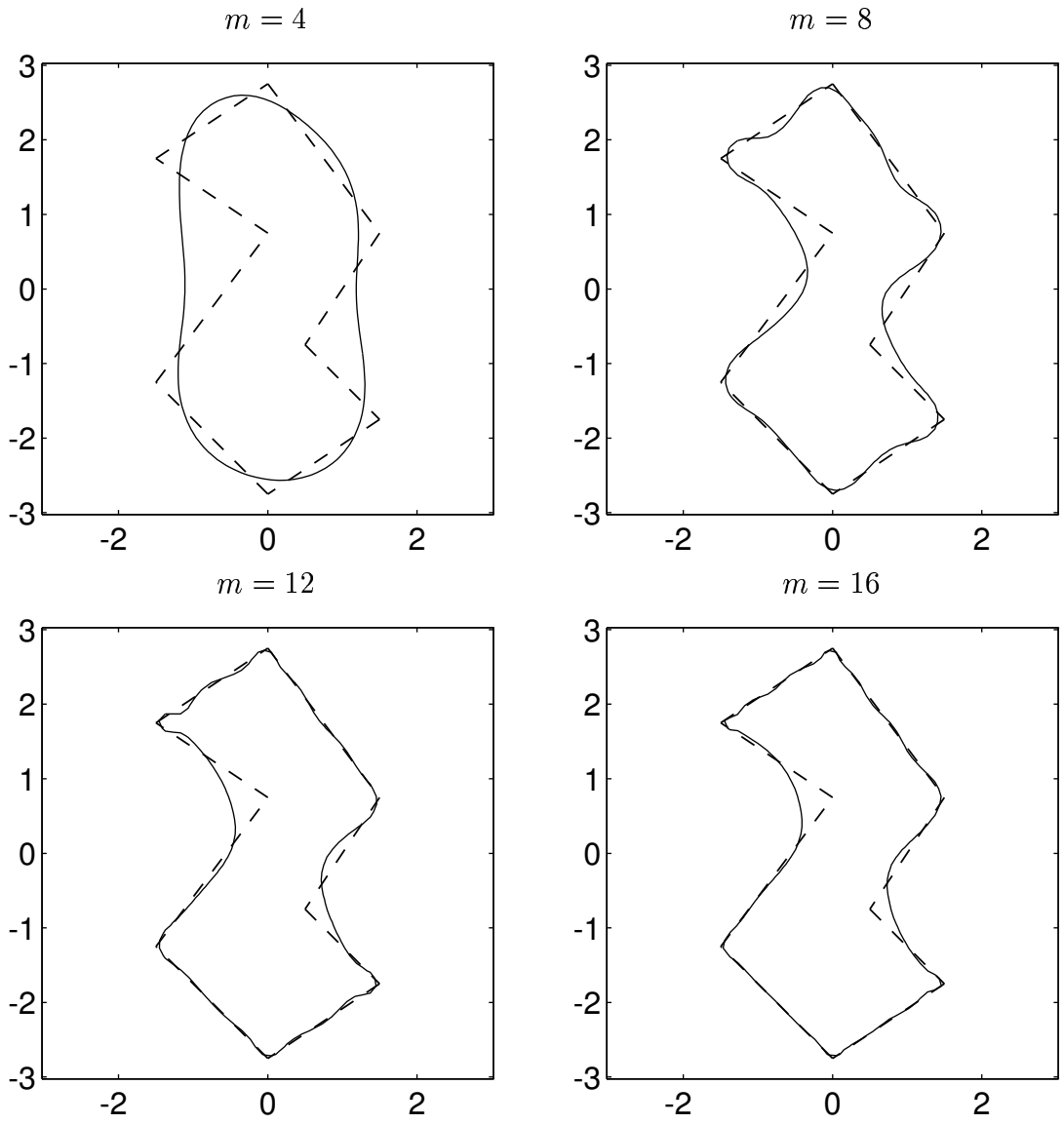


Figure 2.11: Lemniscates of Faber polynomials. The solid curves are the level sets $\{z : |F_m(z)| = 1\}$ for Faber polynomials of degrees $m = 4$, 8, 12, 16 for the dashed polygon. The level sets approximate the polygon better as m increases.

to see [33] that

$$m \approx 16e^{-\pi\frac{K'}{K}},$$

so that $m^{-1/2}$ is exponentially large in the aspect ratio of the rectangle. To compute the Schwarz–Christoffel integral in (2.2) accurately, we must be able to resolve the length scale between -1 and 1 as well as between 1 and $O(\exp(\pi K'/2K))$. If we try to do this simultaneously, we will need better than double precision arithmetic when the aspect ratio exceeds about 23.

A similar exponential crowding effect is found whenever the target region has long, narrow parts. Although the logarithm in the change of variables (2.9) allows an accurate representation of the prevertices even in pathologically crowded situations, the relationship has to be inverted to find the prevertices in order to compute the Schwarz–Christoffel integral. Thus, solving the parameter problem for the half-plane or disk becomes impossible in principle when using fixed precision for regions with arbitrarily long, thin channels. In practice, the numerical solution usually becomes difficult in double precision when some local aspect ratio is 20 or less. Figure 2.12 shows a polygon that exhibits crowding. In Table 2.1 we see that the prevertices of this polygon for both the half-plane and disk formulations are highly clustered.

From one point of view, crowding results from an ill-conditioned choice of fundamental domain. Thus, for example, one way to circumvent crowding for regions that are elongated primarily in one direction only is to map from a strip or rectangle [47]. Because the elongation of the fundamental domain matches that of the target region, the prevertices need not become crowded. For multiply elongated regions, one can work with a slit strip [46], but such methods quickly become rather delicate and specialized, and the fundamental domains are not convenient.

The crowding problem can also been seen as resulting from insisting on a global,

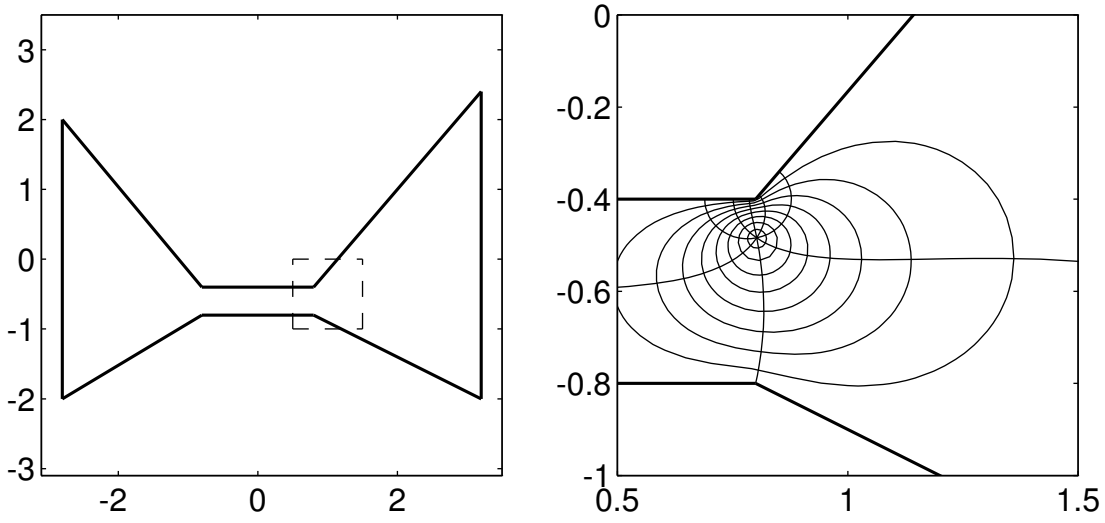


Figure 2.12: On the left, a polygon that exhibits crowding of the prevertices (see Table 2.1). On the right, the disk map for the region inside the dashed lines.

Table 2.1: Prevertices for half-plane and disk maps of the polygon in Figure 2.12.

k	w_k	Half-plane: x_k	Disk: $(\arg z_k)/\pi$
1	$3.2 + 2.4i$	-1	0.00800451739
2	$0.8 - 0.4i$	0	0.606337224
3	$-0.8 - 0.4i$	0.0217140432	1.49999746
4	$-2.8 + 2i$	0.0217140885	1.49999860
5	$-2.8 - 2i$	0.0217140902	1.49999865
6	$-0.8 - 0.8i$	0.0217141442	1.5
7	$0.8 - 0.8i$	0.0396571622	1.75
8	$3.2 - 2i$	∞	2

fixed representation of the prevertices. An alternative numerical method that uses locally adaptive computation to circumvent crowding is discussed in the next chapter.

Another difficulty in using the Toolbox is more particular to the algorithms of the Toolbox itself. The nonlinear systems of Section 2.3 have unique zeros, but Howell [44] points out that they may have many local minima that trap solution methods. The polygon on the left in Figure 2.13 exhibits this phenomenon. If `dparam` is used to solve for this polygon, the nonlinear solver eventually ends up at the polygon on the right in Figure 2.13 and quits, unable to make an improving step. We find that SCPACK also fails on this polygon for many choices of conformal center. The problem is that the two horizontal slits cannot slide past each other without temporarily increasing the solution residual. (The fact that part of the plane is covered more than once is irrelevant.) As Howell points out, this effect of local minima is often masked for the disk map by crowding. This is because the "bumps" that give rise to the local minima frequently cause some prevertices to be so crowded that the solution is unreachable anyway. However, the correct prevertices of the polygon of Figure 2.13 have a minimum spacing larger than 10^{-10}, which is not pathologically crowded in double precision.

Of course, if there were a reliable way to find a good starting guess for the nonlinear solver, local minima would not be an issue. This cannot be done automatically in general, but one can simply solve the nonlinear system first for a modified polygon with the troublesome bumps shrunken until they are not a problem. The resulting solution can be used as a starting guess for the desired polygon. This was done in the case of Figure 2.13 to check that the final solution was not too crowded. This continuation approach is supported by the Toolbox, but it requires user intervention and is inelegant and imperfect. The method of the next chapter produces a nonlinear system that appears to be easier to solve numerically.

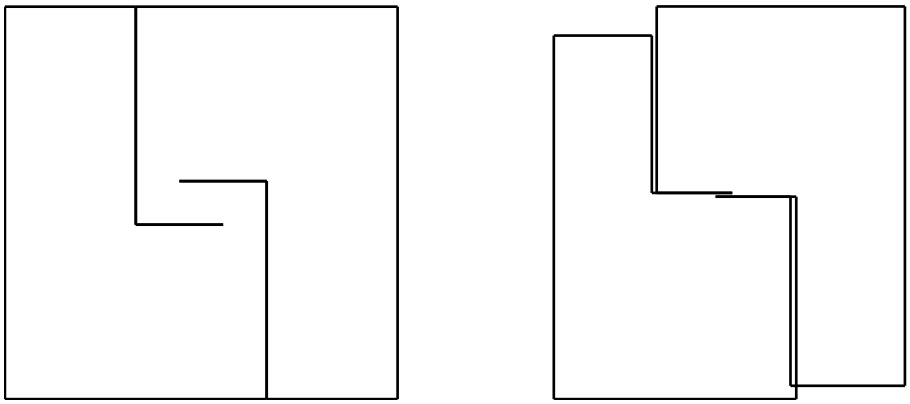


Figure 2.13: Local minima of the nonlinear system. When `dparam` tries to find the prevertices of the polygon on the left, it quits after arriving at the trial polygon on the right. There is no way to continuously deform the trial polygon to the target without temporarily increasing the nonlinear system residual.

Chapter 3

An overlapping algorithm using cross-ratios for numerical Schwarz–Christoffel mapping*

In this chapter we introduce a new algorithm for the Schwarz–Christoffel parameter problem whose accuracy is unaffected by crowding. The algorithm is called CRDT, which stands for "cross-ratios of a Delaunay triangulation."

A key idea underlying the method is that if the degrees of freedom in the Riemann mapping from the disk are left unspecified, there are many possible configurations of the prevertices (related by conformal equivalence) that all map to the same polygon under the Schwarz–Christoffel formula. For many regions, every such configuration, or *embedding*, of the the prevertices will have some very crowded members. However, a small group of prevertices may be uncrowded in *some* embedding, as is illustrated in Figure 3.1. If we could use such an embedding for a group of nearby prevertices whenever we wanted to map in the vicinity of the associ-

*This work was performed with S. A. Vavasis and has been submitted for publication in *SIAM J. Sci. Comput.* [24].

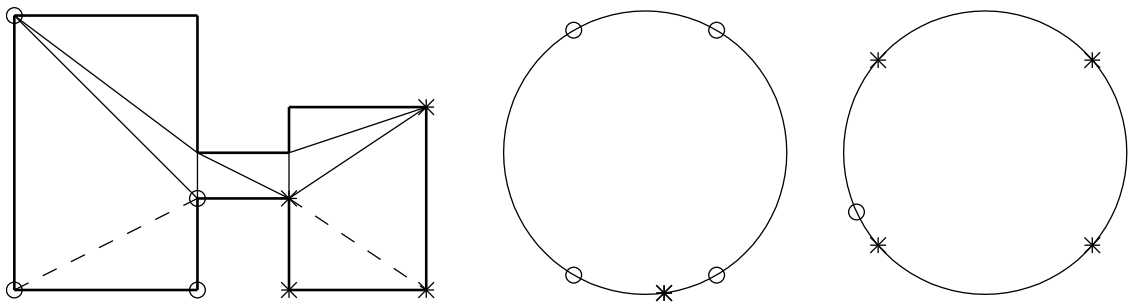


Figure 3.1: Local circumvention of crowding. No embedding of the Schwarz–Christoffel prevertices for the polygon on the left will be globally uncrowded. In the middle picture we see an embedding that keeps the prevertices of the first quadrilateral (marked with circles) uncrowded, while the prevertices of the other (marked with stars) are crowded, too closely to be distinguished. However, in another embedding (far right), the crowding situation is reversed.

ated vertices, we would avoid numerical difficulties with crowding in computing the map. To put matters more crudely, crowding is a global problem, and to circumvent it we act locally.

The grouping of nearby vertices of the polygon is accomplished via domain decomposition. CRDT uses each quadrilateral associated with one of the $n-3$ interior edges of a Delaunay triangulation of the polygon as a fundamental grouping of vertices. Neighboring quadrilaterals overlap at three vertices, a fact which allows their respective embeddings to be uniquely related. Note that since a long, narrow rectangle is a quadrilateral whose prevertices will be crowded in every embedding, a necessary first step will be to decompose such regions by adding new vertices to the long sides.

Given that we want to consider all possible embeddings for a polygon simultaneously, we need a representation of the prevertices that represents the entire conformal equivalence class compactly. A good choice is the set of *cross-ratios* (defined below) of the groupings of four prevertices described above. There will be $n-3$

such cross-ratios, which is the number required to determine the Schwarz–Christoffel mapping, and they are invariant under Möbius transformations and thus represent every possible embedding for a particular polygon. Furthermore, if we choose the constraints in the nonlinear system using cross-ratios of the target polygon rather than side lengths, the resulting system of equations appears to have a monotonicity property that makes its solution generally easier to find than those of the systems described in Chapter 2. Unfortunately, we do not yet have a proof that CRDT always converges to the correct polygon, although we believe this to be true.

We describe the CRDT algorithm in detail in the following sections. We then discuss how CRDT circumvents crowding and support the explanation with numerical evidence. We compare the performance of CRDT to that of the Schwarz–Christoffel Toolbox in numerical experiments. Finally, we indicate how to use the solutions obtained by CRDT in two applications that we believe no other algorithm can handle.

3.1 Delaunay triangulation

We now consider the polygon P to be bounded in the plane. A *triangulation* of P is a division of P into nondegenerate triangles such that (a) the intersection of any two triangles is either a common edge of the two, a common vertex, or empty; (b) the union of the triangles is P; (c) all three vertices of each triangle are also vertices of P; and conversely, (d) every vertex of P is also a vertex of a triangle. It is well known that for any n-vertex simple polygon P, there exists a triangulation of P, and that any triangulation has exactly $n-2$ triangles. A triangulation edge that is not a polygon edge is called a *diagonal*. It is easy to see that any triangulation of P has exactly $n-3$ distinct diagonals.

The *dual* of a triangulation is an $(n-2)$-node graph. The graph has one node for

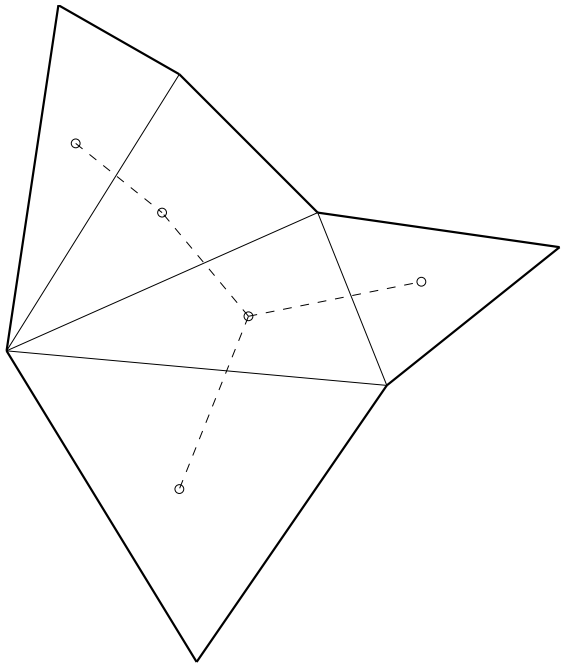


Figure 3.2: The Delaunay triangulation of a 7-sided polygon. The heavy solid segments are boundary edges and the light solid segments are diagonals of the triangulation. The dual graph has five nodes (circles) and four edges (dashed lines). The dual is abstract; the geometry shown here is for convenience.

each triangle and an edge between two nodes if their corresponding triangles have a common diagonal. Thus, the dual graph has exactly $n-2$ vertices and exactly $n-3$ edges, and is in fact always a tree; see Figure 3.2 for an example.

There is a special triangulation of P known as the *constrained Delaunay triangulation* or simply the Delaunay triangulation [6]. This triangulation possesses the following defining property: If d is a diagonal of the triangulation and $Q(d)$ is its associated quadrilateral (the union of the two triangles on either side of d), then the sum of the two opposite interior angles of $Q(d)$ that are split by d is at least π.

Such a triangulation always exists and, if no four points of P are cocircular, it is unique. Moreover, there is a simple algorithm that converges to the Delaunay

triangulation in $O(n^2)$ steps, beginning with an arbitrary triangulation of P: If there exists a diagonal d such that the condition in the last paragraph is violated, then "flip" d—replace d with the other diagonal of $Q(d)$ and update the triangulation accordingly. Repeat until no flips are needed.

3.2 Splitting edges

The first step of CRDT is to split some edges of the polygon. Splitting an edge means replacing it by several smaller edges joined by vertices whose angles are π. Notice that this operation does not affect the Schwarz–Christoffel formula (2.4); a vertex whose interior angle is π has an exponent of 0 in the product.

The purpose of splitting is to make sure each individual quadrilateral in the Delaunay triangulation is well-conditioned. By "well-conditioned" we mean that the prevertices of the quadrilateral are not too crowded in some embedding of the prevertices. In particular, we want to avoid quadrilaterals that are long and narrow with the long edges equal to polygon edges, because the prevertices of such a quadrilateral will be crowded on the unit circle. (A long and narrow quadrilateral is acceptable provided that the polygon is "fat" around it. See Figure 3.3.)

The splitting procedure has two phases. First, for every vertex v with an interior angle of $\pi/4$ or less, we chop off the corner at v as follows. Find the largest isoceles triangle T that can be formed by v and its two adjacent edges such that T is contained in P, and introduce new vertices along the two edges that are adjacent to v at the midpoints of the two sides of T. After this split, the two edges adjacent to v are said to be *protected*; that is, we do not allow them to be split during the second phase. Let P' denote the polygon obtained after this first part of the splitting procedure is complete.

The second phase of the edge-splitting procedure is iterative and generates a

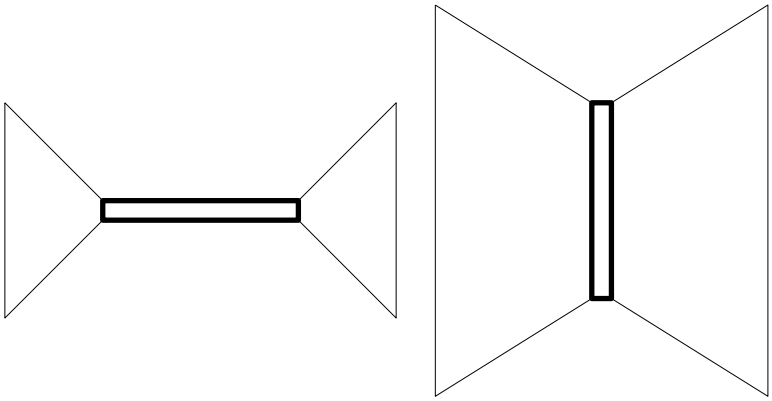


Figure 3.3: On the left is an octagon with a long, narrow quadrilateral in its triangulation (heavy outline). This quadrilateral would have to have its sides split because of crowding: in a mapping with conformal center at the center of the quadrilateral, the short edges of the quadrilateral are exponentially contracted in the preimage on the disk. By contrast, for the polygon on the right, no splitting is necessary, because the polygon is "fat" around the quadrilateral; that is, the quadrilateral can be enclosed in a disk that is mostly interior to the polygon.

sequence of polygons, each of which is a subdivision of its predecessor, starting with P'. Let e be an unprotected edge of some polygon occurring in the iteration. Let $l(e)$ be its length. Let $d(e)$ be the smallest distance from e to any foreign vertex, where "foreign" means a vertex of P other than the endpoints of e and distance is measured geodesically, i.e., along the shortest piecewise linear path that remains inside the polygon. (It turns out that $d(e)$ can be determined efficiently given a triangulation of the polygon.) We say that e is *ill-separated* if

$$d(e) < l(e)/(3\sqrt{2}). \tag{3.1}$$

At each iteration we identify all ill-separated edges and split them into three equal pieces. We repeat this until all edges are well-separated. See Figure 3.4 for an example of both phases of the splitting process. The splitting of edges and protecting of sharp angles is reminiscent of techniques previously introduced

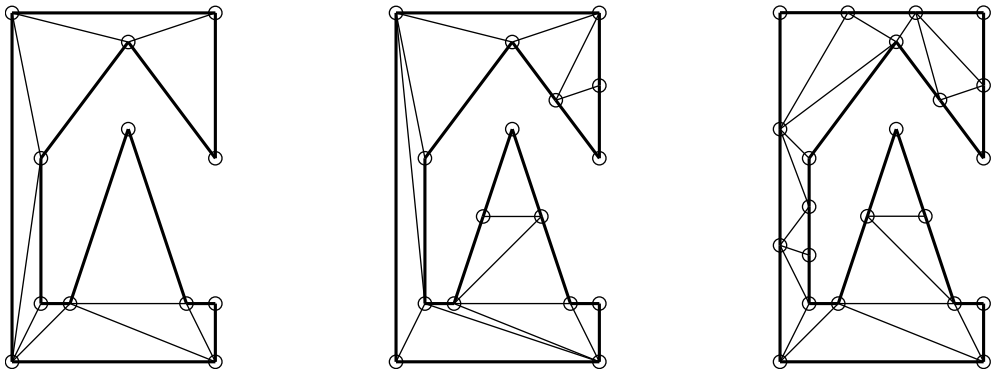


Figure 3.4: On the left is the Delaunay triangulation of a polygon. In the middle we show the Delaunay triangulation after the sharp corners have been chopped in the first splitting phase. The right frame shows the subsequent results of the second phase, in which narrow regions are subdivided.

in the finite-element mesh generation literature; see for example [7, 16, 60]. In the mesh generation literature, the purpose of these techniques is similar to our own, namely, to prevent the occurrence of poorly shaped triangles that would arise in a triangulation of the original (unsplit) polygon. However, finite-element mesh generation subdivides the interior of the domain as well as its boundary, and thus would avoid both kinds of long, skinny quadrilaterals illustrated in Figure 3.3.

We do not try to prove that the splits computed by this procedure are "effective" for our algorithm, because we do not yet have an *a priori* characterization of well-conditioned quadrilaterals. We do prove, however, that the second phase of the splitting procedure always terminates after a finite number of steps. Let $r(e)$ for an edge e be the geodesic distance from e to the closest foreign edge. (A foreign edge is one that is not adjacent to e.) Let r_0 be the minimum of $r(e)$ over all unprotected edges of P'. Suppose there were an edge $e_0 = (v_1, v_2)$ of length shorter than r_0. Let e_1 and e_2 be the other edges whose endpoints are v_1 and v_2, respectively. Then $\text{dist}(e_1, e_2) \leq l(e_0) < r_0$, contradicting the choice of r_0. Thus no edge is shorter

than r_0 in P'.

We claim that the splitting procedure above never produces an edge shorter than r_0. To see this, let $e = (v_1, v_2)$ be an unprotected edge of a polygon at some intermediate stage of the above algorithm whose length is less than $3r_0$. We must argue that we could never split e. By induction, let us assume that no edges up to now are shorter than r_0. Assume e_0 is the original edge of P' that contains e. Let v be the foreign vertex closest to e, i.e., $\text{dist}(v, e) = d(e)$. There are three cases: either v lies on an edge that was foreign to e_0 in P', or it lies on an edge that was adjacent to e_0, or it lies on e_0 itself.

In the first case, we know that $\text{dist}(e_0, v) \geq r_0$ and hence $\text{dist}(e, v) \geq r_0$, whereas $l(e) < 3r_0$. So (3.1) is not satisfied and e is not split.

In the second case, let e_0' be the original edge that contains v, so that e_0 and e_0' are adjacent; say their common point is v'. Notice that e_0 cannot be protected, by assumption. Therefore, the interior angle at v' is greater than $\pi/4$. By assumption, the distance from v' to v is at least r_0. Therefore, some simple trigonometry shows that the distance from v to e_0 is greater than $r_0/\sqrt{2}$. Thus, $d(e) > r_0/\sqrt{2}$, whereas $l(e) < 3r_0$, so (3.1) is not satisfied.

In the third case v is collinear with e, and its distance from e again must be at least r_0, so the same reasoning shows that (3.1) is not satisfied. This completes the proof of termination.

3.3 Cross-ratios

Let a, b, c, d be four distinct points in the complex plane such that the order $abcd$ forms a quadrilateral with counterclockwise vertex order and such that ac is an interior diagonal of the quadrilateral. We define the *cross-ratio* of these points to

be [62]

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

Note the identity $\rho(a,b,c,d) = \rho(c,d,a,b)$. Thus, the cross-ratio depends on the quadrilateral and the diagonal, but not on which endpoint of the diagonal we start at.

In general, the cross-ratio is a complex number, but there is an important special case when it is real.

Lemma 3.1. *Let a, b, c, d be four distinct points on the unit circle in counterclockwise order. Then $\rho(a,b,c,d)$ is a negative real number.*

Proof. The angle of the quadrilateral *abcd* at a and the angle at c are inscribed in complementary arcs of the unit circle, so the sum of these angles must be π. A quick diagram shows that $(d-a)/(b-a)$ has its argument equal to the angle at a, and $(b-c)/(d-c)$ has its argument equal to the angle at c. Therefore, the argument of the cross-ratio, which is the sum of these arguments, is π. □

The $n-3$ primitive real variables of the CRDT nonlinear system arise from $n-3$ cross-ratios of prevertices. The preceding lemma confirms that these variables are indeed real. However, raw cross-ratios are not quite suitable as variables, because we would have to impose side constraints that they be negative. Instead, our unconstrained primitive variables are logarithms of the negatives of the cross-ratios (see (3.2) below).

We now explain *which* $n-3$ cross-ratios we use. Assume the vertices $w_1, \ldots, w_n$ of the polygon P are given in counterclockwise order. Let $d_1, \ldots, d_{n-3}$ and $Q(d_1), \ldots, Q(d_{n-3})$ be the $n-3$ diagonals and associated quadrilaterals of the Delaunay triangulation of P as defined above. Let the vertices of $Q(d_i)$ be denoted by $(w_{\kappa(i,1)}, w_{\kappa(i,2)}, w_{\kappa(i,3)}, w_{\kappa(i,4)})$, so that for each i, $(\kappa(i,1), \kappa(i,2), \kappa(i,3), \kappa(i,4))$ is a

four-tuple of distinct indices in $\{1, \ldots, n\}$. Then the primitive variable σ_i is defined to be

$$\sigma_i = \log(-\rho(z_{\kappa(i,1)}, z_{\kappa(i,2)}, z_{\kappa(i,3)}, z_{\kappa(i,4)})), \qquad i = 1, \ldots, n-3. \tag{3.2}$$

It is apparent that given a list of prevertices $z_1, \ldots, z_n$, the primitive variables $\sigma_1, \ldots, \sigma_{n-3}$ are easily computed from (3.2). For evaluating the nonlinear CRDT mapping, the process must be reversed.

Observe that there are three degrees of freedom, because (3.2) imposes only $n-3$ real constraints on n real variables. We will later use the flexibility afforded by these degrees of freedom to our advantage; indeed, this is the reason that the CRDT algorithm avoids problems with crowding. For now, let us fix the degrees of freedom by assuming that the three prevertices corresponding to a Delaunay triangle T_0 are arbitrarily placed on the unit circle in a manner preserving their ordering. (It turns out that the choice of T_0 and the three prevertex positions will not affect the Schwarz–Christoffel image. See Theorem 3.5 at the end of this section.)

We now show how to embed the remaining $n-3$ vertices using the cross-ratio information, starting with a lemma that tells us how to place a single prevertex.

Lemma 3.2. *Given distinct points a, b, c on the unit circle in counterclockwise order, and given a negative real number ρ_0, there exists a unique point d on the unit circle such that $\rho(a, b, c, d) = \rho_0$. Furthermore, this point is counterclockwise from c and clockwise from a.*

Proof. If we write out the formula and substitute, we get an explicit formula for d:

$$d = \frac{hc + a}{h + 1}, \tag{3.3}$$

where

$$h = \frac{\rho_0(b - a)}{(c - b)}.$$

We must first show that $h \neq -1$, so that the denominator in the formula for d is nonzero. But this is obvious, because $(b-a)/(c-b)$ must have a nonzero imaginary part (since a, b, c cannot be collinear), so h also has a nonzero imaginary part. Thus d is uniquely determined.

Next, we must show that d lies on the unit circle between c and a. Consider sliding a test point along the unit circle starting from very near a clockwise to c. It is easy to check that the cross-ratio, which is a negative real number by the earlier lemma, varies continuously from 0 to $-\infty$. Therefore, its value must be ρ_0 at some intermediate point. But the last paragraph shows that this intermediate point is unique. □

We can now prove that all $n-3$ undetermined prevertices can be placed uniquely.

Theorem 3.3. *Let T_0 be any triangle in the Delaunay triangulation of P, and let its vertices in counterclockwise order be indexed as w_ϕ, w_ψ, w_χ. Suppose the prevertices z_ϕ, z_ψ, z_χ, are specified as distinct points on the unit circle in counterclockwise order. Then given any real-number values for the primitive variables $\sigma_1, \ldots, \sigma_{n-3}$, there exists a unique solution to* (3.2)*, that is, a unique way to define the remaining $n-3$ z_i's on the unit circle satisfying* (3.2)*. The algorithm to find the z_i's takes $O(n)$ steps. Furthermore, $z_1, \ldots, z_n$ will lie in counterclockwise order.*

We call such a placement of the prevertices an *embedding.*

Proof. This proof relies heavily on the fact that the dual of the Delaunay triangulation is a tree. In the ensuing discussion, "nodes" and "edges" refer to nodes and edges of the tree, whereas "vertex" and "diagonal" refer to vertices and diagonals of the Delaunay triangulation.

In a tree, there is always a unique path between any pair of nodes. Let us root the tree at T_0. Every node in a rooted tree except the root has a parent, and every

node except the leaves has children. Therefore, for each triangle in the triangulation except the root, we can identify the diagonal that separates it from its parent; we call this the *entry diagonal* of the triangle. The vertex opposite the entry diagonal is called the *new vertex*. Notice that choice of entry diagonal and new vertex depends not only on the triangle but on the choice of T_0 as well.

We next claim that the $n-3$ new vertices of the $n-3$ nonroot triangles are precisely the $n-3$ vertices of the polygon whose prevertices are to be determined. It is clear that no new vertex can be a vertex of T_0. Furthermore, two distinct nonroot triangles cannot have a common new vertex. The justification for this claim is provided by Figure 3.5. In particular, the figure shows that if two triangles had the same new vertex, then there would be a cycle in the dual, contradicting the fact that the dual is a tree. Let the nonroot triangles and their new vertices be denoted by $T_1, \ldots, T_{n-3}$ and $w_{i_1}, \ldots, w_{i_{n-3}}$, respectively. Thus, the disjoint union of index sets $\{i_1, \ldots, i_{n-3}\}$ and $\{\phi, \psi, \chi\}$ is $\{1, \ldots, n\}$. Let the entry diagonal of nonroot triangle T_k be denoted by d_{j_k}, which is associated with quadrilateral $Q(d_{j_k})$. Each entry diagonal corresponds to exactly one of the primitive variables, because each diagonal (and its quadrilateral) comes up exactly once in the list for $k = 1, \ldots, n-3$.

We now describe our algorithm for embedding the $n-3$ remaining prevertices. Visit each node of the tree once. When visiting the node for triangle T_k, compute the position of prevertex z_{i_k}. Any search order that guarantees that parents are visited before their children (for instance, depth-first or breadth-first) is acceptable.

In more detail, we use (3.3) for computing z_{i_k} because we know that z_{i_k} corresponds to the quadrilateral $Q(d_{j_k})$, which is the union of T_k and its parent in the tree. The other three vertices of the quadrilateral are already embedded by previous steps in the search, because the parent is embedded before the child. Given the position of the three other vertices of this quadrilateral and the cross-ratio $-\exp(\sigma_{j_k})$, prevertex z_{i_k} is uniquely determined because of Lemma 3.2. But then an induc-

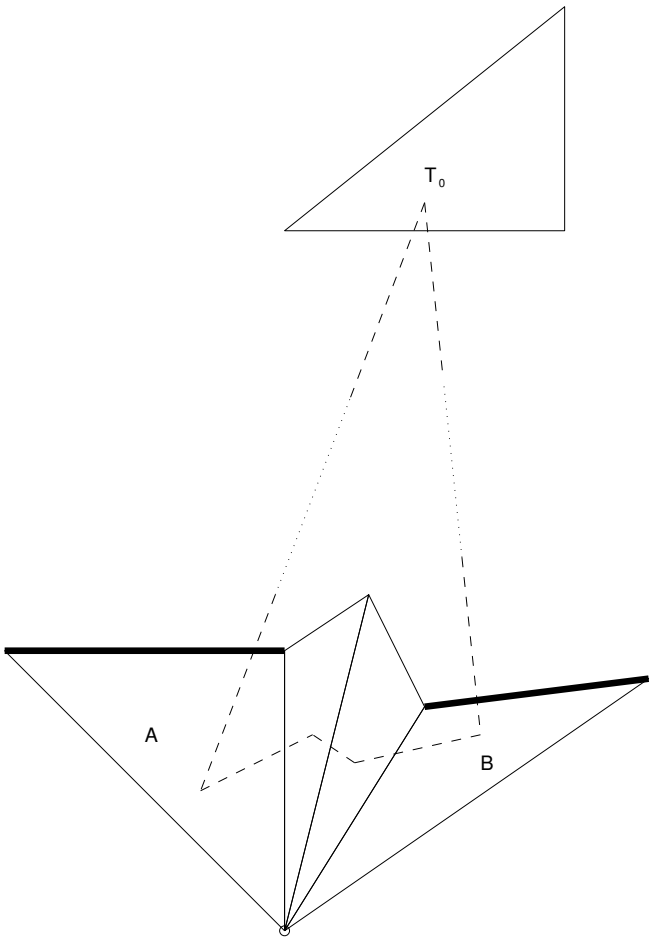


Figure 3.5: An illustration of the contradiction that would arise if two different triangles in the triangulation had the same new vertex. If triangles A and B, whose entry diagonals are shown as heavy lines, both shared this new vertex (marked with the circle), there would be a cycle in the dual graph (marked with dashed lines).

tion argument shows that the positions of all the z_{i_k}'s are uniquely determined, i.e., every step of the construction is forced. This shows uniqueness of the solution. Furthermore, (3.2) is satisfied for this construction for $i = 1, \ldots, n-3$ because we used each σ_i exactly once in the preceding construction.

The only remaining claim is that $z_1, \ldots, z_n$ will end up in counterclockwise order. Again, this follows from a combination of Lemma 3.2 and the fact that the dual is a tree. Let T_k be the current triangle and w_{i_k} its new vertex. Observe that Lemma 3.2 ensures that when we place z_{i_k} on the unit circle, it will be on the arc between the endpoints (call them z_a and z_b) of the entry diagonal of T_k that is complementary to the arc that contains the parent triangle's prevertices. Therefore, z_{i_k} is placed correctly with respect to z_a and z_b. But notice that z_{i_k} must be the first prevertex placed between z_a and z_b because any other prevertices on this arc are the new vertices of children of T_k. Thus, the placement of z_{i_k} is correct with respect to all vertices placed before it. □

This theorem shows that given values for the primitive variables and the positions of the three prevertices corresponding to a Delaunay triangle, we can compute the positions of all of the prevertices. How should we choose the initial triangle and embed its prevertices? It turns out that *any* initialization is acceptable; all embeddings give the same image polygon, up to similarity transform.

To show this, we begin by recalling some standard facts from complex analysis [62]. First, any conformal map from the unit disk to itself is a Möbius transformation of the form

$$g(z) = e^{i\theta} \frac{z - r}{1 - \bar{r}z}, \tag{3.4}$$

where r is a complex number such that $|r| < 1$ and $\theta \in [0, 2\pi)$. As a consequence, there is a unique such mapping that takes any three distinct points on the unit circle

to any other three points on the unit circle. We formalize another consequence in a lemma:

Lemma 3.4. *Let a, b, c, d be four points on the unit circle in counterclockwise order. Let g be a conformal mapping of the unit disk to itself. Then*

$$\rho(a, b, c, d) = \rho(g(a), g(b), g(c), g(d)).$$

Proof. Simple algebra verifies that cross-ratios are invariant under Möbius transformations. See [62] for details. □

We now show that no matter which initial triangle T_0 we select in Theorem 3.3, and no matter how we choose the positions of its three prevertices, we end up with the same image polygon, up to similarity transform.

Theorem 3.5. *Let $\sigma_1, \ldots, \sigma_{n-3}$ be given. Let T_0, T_0' be two triangles in the Delaunay triangulation whose vertices are (w_ϕ, w_ψ, w_χ) and $(w_{\phi'}, w_{\psi'}, w_{\chi'})$, respectively. Let (z_ϕ, z_ψ, z_χ) and $(z'_{\phi'}, z'_{\psi'}, z'_{\chi'})$ be order-preserving embeddings of their prevertices in the unit disk as in Theorem 3.3. Use the construction of Theorem 3.3 to produce the two embeddings $(z_1, \ldots, z_n)$ and $(z'_1, \ldots, z'_n)$, respectively. Let $\tilde{P}$, $\tilde{P}'$ be the images of the unit disk under* (2.4) *using these two prevertex sets. Then $\tilde{P}$ and $\tilde{P}'$ are* similar*, i.e., they agree up to a translation, rotation, and scaling.*

Remark. We have not explained how to choose the affine constants A and B in (2.4). The theorem holds for any choice of these constants. Alternatively, the theorem asserts that, given the constants for one embedding, these constants can be chosen for the other embedding in such a way that the image polygons coincide.

Proof. Let $(z'_\phi, z'_\psi, z'_\chi)$ be the positions of the prevertices of T_0 in the second embedding. Let g be the conformal mapping of the unit disk to itself carrying the points

(z_ϕ, z_ψ, z_χ) to $(z'_\phi, z'_\psi, z'_\chi)$. That is, g maps the prevertices of T_0 in one embedding to the prevertices of T_0 in the other.

We claim that $g(z_i) = z'_i$ for all i, not just $\{\phi, \psi, \chi\}$. Because g preserves cross-ratios, the embedding $(g(z_1), \ldots, g(z_n))$ has the same $n-3$ cross-ratios as $(z_1, \ldots, z_n)$, which by assumption has the same $n-3$ cross-ratios as $(z'_1, \ldots, z'_n)$. But since three entries in $(g(z_1), \ldots, g(z_n))$ are equal to the three corresponding entries in $(z'_1, \ldots, z'_n)$, the uniqueness part of Theorem 3.3 guarantees that $g(z_i) = z'_i$ for all i.

Now consider composing (2.2) with the conformal mapping g^{-1} of the disk to itself. The composition is a conformal mapping from the unit disk to $\tilde{P}$, so the Schwarz–Christoffel formula (2.4) must also hold when the prevertices are given by $(z'_1, \ldots, z'_n)$, for a suitable choice of the affine constants. So $\tilde{P}$ and $\tilde{P}'$ are similar. □

3.4 The CRDT algorithm

We are now prepared to specify the CRDT algorithm in detail.

Step 1. Split the edges of the polygon as described in Section 3.2. We now use P to denote the polygon obtained after splitting, and n to denote the number of its vertices.

Step 2. Compute the Delaunay triangulation of P. Now that the Delaunay triangulation is computed, we can fix a particular numbering of the diagonals and quadrilaterals in the triangulation. Recall the notation $\kappa(i, j)$ used in (3.2) to number the vertices of the quadrilaterals.

We define

$$c_i = \log(|\rho(w_{\kappa(i,1)}, w_{\kappa(i,2)}, w_{\kappa(i,3)}, w_{\kappa(i,4)}|) \tag{3.5}$$

for $i = 1, \ldots, n-3$. Note that the cross-ratio in this formula in general will be a

complex number.

Step 3. Solve the nonlinear system $F(\sigma) = 0$. The map $F : \mathbf{R}^{n-3} \to \mathbf{R}^{n-3}$ is defined as follows. The input variables are the primitive variables $\sigma_1, \ldots, \sigma_{n-3}$ defined by (3.2). It was shown in Section 3.3 that there is a unique (up to similarity) Schwarz–Christoffel mapping that can be computed from these primitive variables. Let $\omega_1, \ldots, \omega_n$ be the vertices of the image of (2.4). For $i = 1, \ldots, n-3$, let

$$F_i(\sigma_1, \ldots, \sigma_{n-3}) = \log(|\rho(\omega_{\kappa(i,1)}, \omega_{\kappa(i,2)}, \omega_{\kappa(i,3)}, \omega_{\kappa(i,4)})|) - c_i.$$

Observe that although the ω's themselves are determined only up to similarity transform, the cross-ratio of four of them is invariant under similarity transform, so this definition makes sense. We discuss nonlinear solvers in Section 3.6.

At a solution to $F(\sigma) = 0$, we have for $i = 1, \ldots, n-3$ that

$$|\rho(\omega_{\kappa(i,1)}, \omega_{\kappa(i,2)}, \omega_{\kappa(i,3)}, \omega_{\kappa(i,4)})| = |\rho(w_{\kappa(i,1)}, w_{\kappa(i,2)}, w_{\kappa(i,3)}, w_{\kappa(i,4)})|.$$

Furthermore, we know that all the interior angles of the polygon determined by $\omega_1, \ldots, \omega_n$ are correct because the angles are inherent in the Schwarz–Christoffel mapping. Is this polygon the correct one? We are not able to prove this, so we state it as a conjecture.

Conjecture 3.6. *Let P be a bounded n-vertex triangulated polygon. P is uniquely determined up to similarity transform by the following data:*

- *the sequence of all interior angles of P at its vertices, and*
- *the list of $n-3$ absolute values of cross-ratios of the quadrilaterals determined by the triangulation of P.*

We have verified this conjecture analytically in the cases $n = 4$ and $n = 5$. In practice, it is easy to check whether the right polygon has been computed. Our

computational experiments (see Section 3.6) support the conjecture, as CRDT has never failed to converge to the correct polygon.

If the conjecture does turn out to be false, we can modify CRDT so that the $n-3$ equations enforce side length conditions as in Section 2.3. The advantage of equations that enforce cross-ratio conditions is that the system of nonlinear equations apparently has a desirable monotonicity property, as we will see in Section 3.6.

We conclude this section with a description of our algorithm for computing F given a value of σ. For each component F_i for some $i = 1, \ldots, n-3$, we need to know four image vertices $\omega_{\kappa(i,1)}, \ldots, \omega_{\kappa(i,4)}$. To find these, we construct a certain embedding E_i of the prevertices. Recall from Section 3.3 that given the vector σ, we can arbitrarily embed three of the prevertices, such as $z_{\kappa(i,1)}, z_{\kappa(i,2)}, z_{\kappa(i,3)}$. We embed these three in such a way so that when $z_{\kappa(i,4)}$ gets placed by the algorithm in Section 3.3, these four prevertices on the unit disk will be arranged in a rectangle centered at the origin with the correct cross-ratio (i.e., cross-ratio $-\exp(\sigma_i)$). Then we position the rest of the prevertices to complete E_i. This embedding defines a Schwarz–Christoffel map f_i, given by (2.4) with $A = 0$ and $B = 1$, which we use to compute the relative positions of the four image vertices $\omega_{\kappa(i,1)}, \ldots, \omega_{\kappa(i,4)}$. The path of integration is a straight-line segment from the origin, and we use the compound Gauss–Jacobi quadrature rules described in Section 2.3.

3.5 On the circumvention of crowding

As was suggested by Figure 3.1, the use of different embeddings for each of the nonlinear equations is the key to why CRDT is unaffected by crowding. Indeed, the embedding E_i described in the last section for computing F_i guarantees that $z_{\kappa(i,1)}, \ldots, z_{\kappa(i,4)}$ will *not* be crowded, either against each other or against any other prevertex. Therefore, even if other groups of prevertices are crowded, the crowd-

ing has no impact on the accuracy of the quadrature rule applied to these four prevertices—the path of integration never passes close to the crowded prevertices.

In order to substantiate the claim that none of the four prevertices are crowded against each other or their neighbors, we need a result stating that none of the cross-ratios $\rho_1, \ldots \rho_{n-3}$ of the prevertices is very large (close to $-\infty$) or very small (close to 0). Unfortunately, there cannot exist fixed (constant) upper or lower bounds on these cross-ratios that apply to all polygons, as the following example shows. Consider CRDT applied to the regular n-gon. Any triangulation of the regular n-gon is a Delaunay triangulation. Thus, CRDT might compute a triangulation that has a quadrilateral whose aspect ratio is $O(n)$. Since the Schwarz–Christoffel mapping of an n-gon is close to the identity mapping, there will also be four prevertices whose cross-ratio is $O(n^2)$, or $O(n^{-2})$. Thus, there is no *a priori* upper or lower bound possible on the ρ_i's, and therefore none on the σ_i's either. This growth of the σ_i's is very slow (logarithmic in n) and is thus not expected to have a significant impact on the accuracy of CRDT. But the absence of a constant upper bound or lower bound means, for example, that there is no *a priori* upper bound on how much adaptation is necessary in the compound Gauss–Jacobi integration used to evaluate (2.4).

The example in the last paragraph has bad triangles in the original Delaunay triangulation, so one could argue that the growth of the cross-ratios of the prevertices as $n \to \infty$ is unavoidable. Thus, a more plausible conjecture might be that the difference $\sigma_i - c_i$, (where c_i was defined by (3.5)) has constant upper and lower bounds. This result would also show that CRDT is not affected by crowding, in the sense that it never works with distances that are substantially shorter than edge-lengths in the original polygon. Note that both quantities σ_i and c_i are logarithms, so subtracting them is the right way to check how they differ. We do not know whether this conjecture is true, but we have not seen substantial divergence

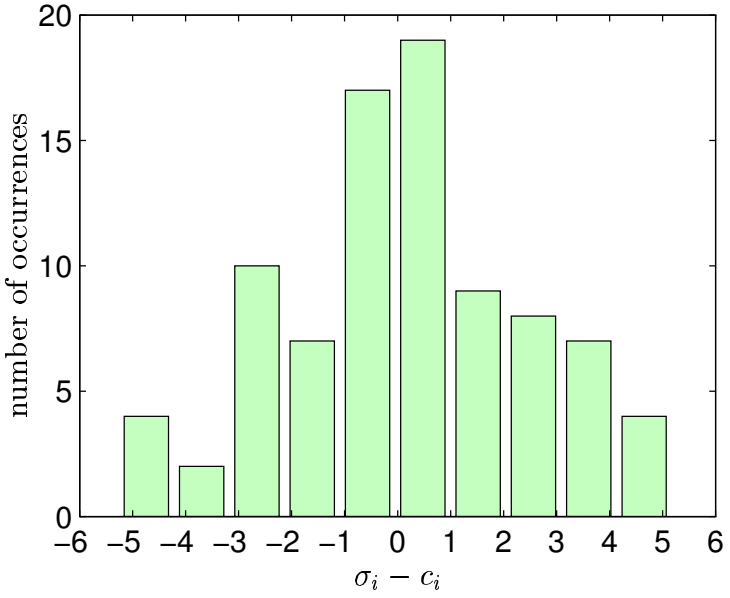


Figure 3.6: Evidence of controlled cross-ratio sizes. The histogram is of the discrepancies $\sigma_i - c_i$ between the primitive cross-ratio variables and polygon cross-ratios for the experiments in Section 3.6. The variables are logarithmic. The clustering near zero and absence of large outliers supports the claim that CRDT circumvents crowding.

between σ_i and c_i in our computational experiments. In Figure 3.6 we present a histogram of $\sigma_i - c_i$ for all polygons in our experiments, for all indices i, at the final solutions obtained by the CRDT algorithm. Observe that points in this histogram lie in a fairly narrow range. (The cross-ratios stay similarly bounded in intermediate steps of the iterations because of the monotonicity observed in our nonlinear equations, described in the next section.)

The representation of prevertices used by the SC Toolbox is also well-conditioned in the presence of crowding, because logarithms of distances between prevertices are used as primitive variables rather than the prevertex positions themselves. But the logarithmic distances cannot be used directly to compute Schwarz–Christoffel maps, so one must pass from these primitive variables to particular fixed prevertex

positions. Hence forward evaluation of F is spoiled by cancellation in these packages because of this intermediate step. In CRDT, our representation is also unaffected by crowding, and, furthermore, when evaluating F we can work directly with the well-conditioned cross-ratio representation (by juggling the prevertices so that the ones of interest are never crowded) to avoid cancellation.

Does this mean that CRDT "solves" all problems with crowding? If CRDT is applied to a problem to potential theory, and the application requires (as an intermediate step) a fixed embedding of the prevertices, then there is no improvement with CRDT compared to the Toolbox. Therefore, if one wants to use CRDT as a subroutine, then one must devise an algorithm in which the cross-ratios produced by CRDT are used to shuffle between different embeddings of the prevertices. In Section 3.7 we give some examples of problems from potential theory that can be solved in this manner by CRDT. In fact, for every reasonable problem in potential theory that we have considered, we have always been able to find a way to use the cross-ratios directly and avoid crowding. But in each case the technique we have devised is slightly different, so we are not able to state with certainty that CRDT can solve all problems in potential theory posed on elongated polygons.

3.6 Numerical experiments

In this section we experiment with an implementation of CRDT in MATLAB. We present some evidence of the monotonicity of the nonlinear system and consider the matter of solving it numerically. We also compare the performance of the CRDT algorithm to that of the SC Toolbox and find that CRDT is competitive for most regions. The principal exceptions are regions for which the edge-splitting algorithm adds a great many extra vertices of angle π. While such vertices do not affect the amount of work in computing the Schwarz–Christoffel integral (2.4), they do affect

the size of the nonlinear system to be solved.

The computational core of CRDT is the solution of the nonlinear system $F(\sigma) = 0$. We consider nonlinear solvers that require only function evaluations. According to our experiments, a particular very simple linear iteration always converges at a linear rate. This iteration starts with $\sigma^{(0)} = c$, where c_i was defined above by (3.5). Then we iterate:

$$\sigma^{(k+1)} = \sigma^{(k)} - F(\sigma^{(k)}). \tag{3.6}$$

Our conclusion from experiments is that this iteration always satisfies $||F(\sigma^{(k+1)})||_2 \leq \alpha||F(\sigma^{(k)})||_2$, for an α that is problem dependent but always satisfies $\alpha < 1$. We have not been able to devise a convincing explanation of this behavior. The essential reason is apparently that the Jacobian F' approximates the identity, but none of the likely conditions on F' that would support this claim have been found to hold in experiments.

By way of contrast, in Figure 2.13 we demonstrated problems with the global convergence of `dparam` in the SC Toolbox. We do not know whether there exist polygons for which CRDT exhibits similar difficulties.

In practice, we do not use the simple iteration (3.6) for CRDT, because its convergence is too slow. Instead we use two variations of the nonlinear equation solver NESOLVE, the package used by the SC Toolbox. In one variant, "Full CRDT," we use the standard finite-difference Jacobian to seed the Broyden update. In the other, "Shortcut CRDT," we attempt to exploit the system's monotonicity by setting the initial $F' = I$. In Figure 3.7 we show the convergence curves of the two NESOLVE variants and the simple iteration for a typical case, the goblet shape in Figure 3.8. The linear convergence of (3.6) is strikingly smooth. The convergence of the NESOLVE variations is more complex, but overall is approximately linear at much better rates than the simple iteration. Note that Shortcut CRDT is faster

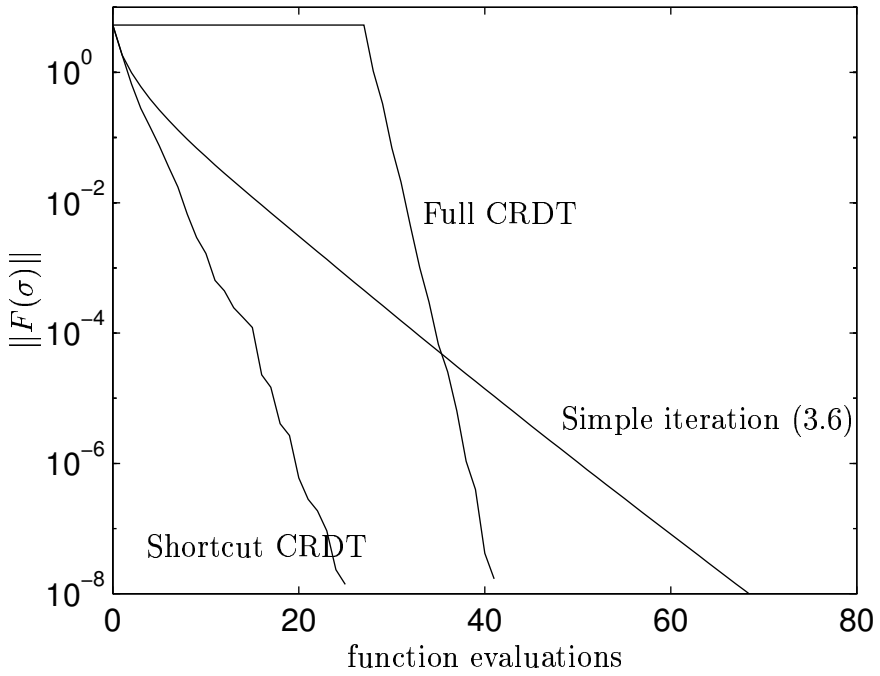


Figure 3.7: Convergence curves for numerical solutions of $F(\sigma) = 0$ for the "goblet" polygon of Figure 3.8. The iteration (3.6) converges linearly. The NESOLVE variations exhibit approximately linear convergence as well, but at much better rates.

Table 3.1: Performance of CRDT variants versus the SC Toolbox for the solution of the parameter problem for the regions in Figure 3.8. The first number in each entry is the number of nonlinear function evaluations made by the nonlinear solver; the other number is the CPU time in seconds.

	SC Toolbox	Full CRDT	Shortcut CRDT
Cross	31/10.03	18/14.52	12/11.78
Goblet	78/14.35	42/142.3	26/112.3
Spiral	186/275.4	51/237.1	34/197.0
Y	−/−	35/76.37	25/70.38

than Full CRDT because of the savings gained by not initializing F'.

We now compare the performance of CRDT to the SC Toolbox functions `dparam` or `rparam` for maps from the disk or rectangle, respectively, depending on whether the target region is elongated. From each method we demand a nonlinear residual with maximum norm no larger than 10^{-8}. Since the CRDT solution is not known to produce the correct polygon in every case, the final CRDT solution is checked by applying the Schwarz–Christoffel formula and checking complex cross-ratios (not just absolute values). In every case the resulting error is within a factor of ten of the nonlinear residual. All the experiments reported here were performed on a SPARCstation-10.

Figure 3.8 shows four test polygons with their Delaunay triangulations, after the edge splitting has been done. In Table 3.6 we present the number of function evaluations and total CPU time required by the nonlinear system solver for these regions.

Note that in every case the Shortcut CRDT variant indeed finishes more quickly than Full CRDT. Also observe that the individual SC Toolbox iterations are much faster than those for CRDT. This is because of the additional unknowns introduced

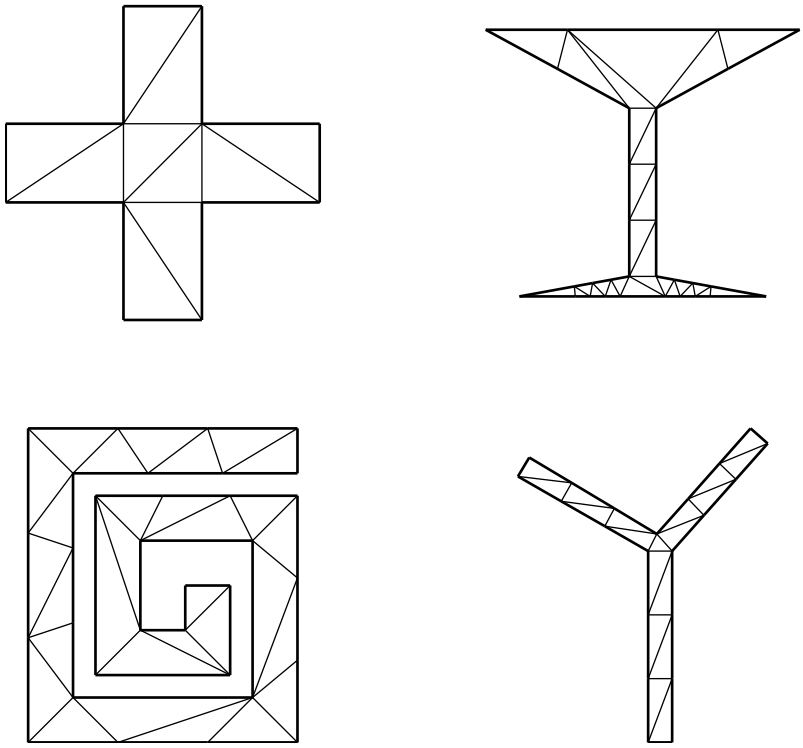


Figure 3.8: Four regions on which CRDT experiments were performed, after splitting and Delaunay triangulation. See text.

by splitting and the adaptive embedding used by CRDT to avoid crowding. However, the nonlinear systems posed by the SC Toolbox for these polygons are not as easily solved, and thus many more iterations may be required even though the systems are smaller.

The cross-shaped region (top left) is not elongated and both `dparam` and the CRDT variants converge rapidly. The SC Toolbox requires less setup effort, and the adaptive re-embedding of the CRDT algorithm is unnecessary, so `dparam` is slightly faster. The goblet region (top right) has 22 vertices added by the splitting algorithm to its original eight. These extra vertices greatly slow down the CRDT solvers, making `dparam` much faster even though it has some difficulty finding the solution. For the spiral (bottom left), relatively fewer vertices are added, and the solution is sufficiently difficult for `rparam` that CRDT is a little faster. Finally, for the Y-shaped region (bottom right), the SC Toolbox is unable to find a solution because of the doubly-elongated nature of the region. CRDT, however, finds the solution easily.

These examples demonstrate what we observe about CRDT in general. CRDT is least efficient when many extra vertices are added during splitting. This most often occurs near sharp corners and in narrow channels of the region, both of which are prominent in the goblet region. While it seems that there is no way to circumvent subdividing a channel because of crowding effects, we do not know if there is a more efficient way to produce well-conditioned quadrilaterals near sharp corners. On the other hand, CRDT handles multiply elongated regions with no difficulty, something which previously no single method for Schwarz–Christoffel mapping has been able to do.

3.7 Applications

In this section we describe two applications of CRDT that demonstrate its ability to handle crowding caused by elongation. While we cannot specify a recipe that will solve any conceivable problem of interest, we believe that the techniques of this section can be adapted to suit a variety of situations. A common thread in all the methods that we have explored is the careful use of local information. Any need for global information is typically obtained by taking a path through the polygon and compounding local effects.

3.7.1 Rectangle map to a multiply elongated polygon

Our first example is the computation of the map from a rectangle to a generalized quadrilateral. For a multiply elongated region, branches other than the main channel will collapse into crowded clusters on the sides of the rectangle. Computing the map accurately in the vicinity of these clusters (into the collapsed branches) is therefore challenging.

In Figure 3.9 we plot the images of straight lines in a rectangle mapped to a certain quadrilateral. The rectangle has a width of 1 and a height of about 18.2, which is the conformal modulus. The solid curves are images of lines which are well separated from the long edges of the rectangle, so they stay out of the wrong turns, or "deadwaters." The dotted curves have preimages that are exponentially close to the long rectangle edges, and they closely follow the maze boundary into the deadwaters. The crowding of the spiral branch is comparable to machine precision, and it would pose no difficulty if it were much more crowded. As far as we know, no other existing conformal mapping algorithm can accurately compute these curves.

We proceed to describe how we produced Figure 3.9. Suppose the prevertices of the Schwarz–Christoffel map to the polygon P are known. Define a new vector of

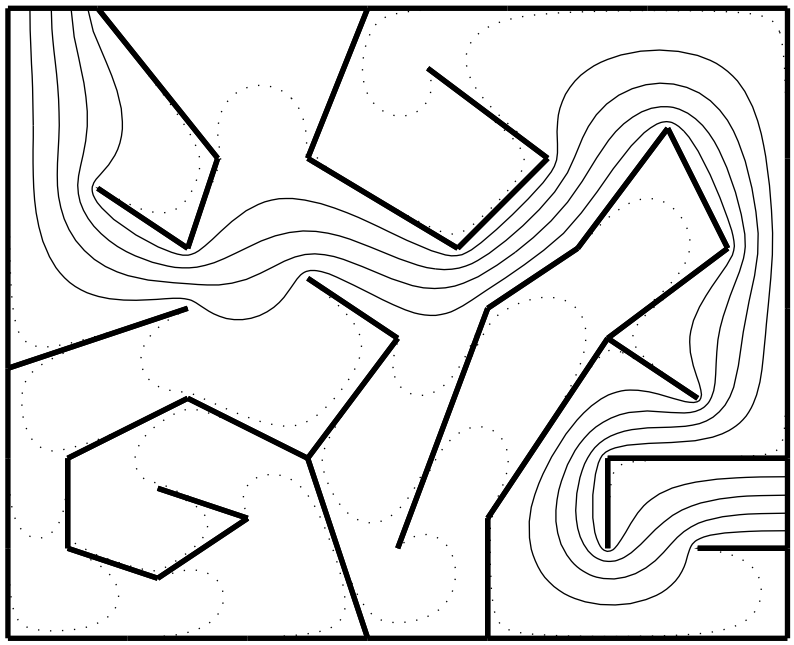


Figure 3.9: Rectangle map to a "maze." The exits of the maze map to the short rectangle sides, which are normalized to unit length. The maze has several branches that are very crowded on the long rectangle sides. The solid curves in the maze are images of lines with abscissae 0.2, 0.4, 0.6, and 0.8. The preimages of the dotted curves are separated from the long rectangle edges by 10^{-2}, 10^{-4}, ..., 10^{-16}. All computations were performed in double precision.

turning angles $\tilde{\beta}$, where $\tilde{\beta}_j = -1/2$ if vertex j is distinguished and $\tilde{\beta}_j = 0$ otherwise. We call the Schwarz–Christoffel map defined by using $\tilde{\beta}$ in place of β in (2.4) and the same prevertices the *rectified map* for the polygon, because the image of the disk under this map is clearly a rectangle. Thus the composition of the inverse of the rectified map with the original map is the desired rectangle map.

We must now consider the mechanics of forward and inverse mapping using CRDT's cross-ratio representation. Recall that each quadrilateral $Q(d_i)$ (Q_i for short) has an associated embedding, $E_i = (z_1^{(i)}, \ldots, z_n^{(i)})$, which depends on the primitive variable vector σ. Each embedding in turn induces a Schwarz–Christoffel map f_i (having $A = 0$ and $B = 1$ in (2.4)) such that $f_i(D)$ is an affine transformation of the target polygon P, assuming $F(\sigma) = 0$. An important feature of CRDT in the evaluation of F is that the prevertices of Q_i are well-separated in E_i.

In order to compute the map to P from embedding E_i, we must find the appropriate affine constants A_i and B_i that transform $f_i(D)$ to P. An obvious way to compute these constants would be to compute the Schwarz–Christoffel integral for the well-separated prevertices of Q_i, and solve for A_i and B_i by matching with vertices of P. However, there are two complications. First, in the case of the rectified map, the only vertices known initially are the ends of one side. Hence the other vertices, and the associated affine constants, must be found in a certain order. Second, even when the computation is ordered correctly, some of the vertices may be arbitrarily crowded, and the linear system that defines A_i and B_i might be ill-conditioned or even numerically singular.

The key to avoiding both of these difficulties is to follow a path through P. Recall that the dual graph of P, which is a tree, has $n - 3$ edges, one for each diagonal (hence quadrilateral) of the Delaunay triangulation. We define another graph, the *quadrilateral graph*, with a node for each quadrilateral. Quadrilaterals are adjacent if their edges in the dual graph share an endpoint, or, in terms of

P, if they share three vertices. The quadrilateral graph is connected but is not necessarily a tree.

Given a reference edge e of the target polygon P, we choose a quadrilateral Q_1 that has e as an edge. As in the CRDT iteration, we compute f_1 to find a scaled and translated version of Q_1. The affine constants for this embedding, A_1 and B_1, can be found using e as reference. Now we visit the nodes of the quadrilateral graph so that every node is visited after at least one of its neighbors has been visited. For each quadrilateral Q_i, we apply f_i to the prevertices of Q_i. Since Q_i shares three vertices with its previously visited neighbor Q_j, we have sufficient reference for computing an affine transformation from the image plane of f_i to the image plane of f_j. But then, by composition with the known affine constants A_j and B_j, we can find A_i and B_i. Even though the resulting scaling constant B_i may be exponentially small, it is found to high relative accuracy by multiplication. The important point is that for each embedding, reference is made to the raw, unscaled image of a neighboring embedding, because the transformation between the two is well-conditioned. If the scaled image were used instead, there could be a total loss of accuracy due to ill-conditioning in the presence of crowding.

Observe that once all of the A_i and B_i have been found, they can be used to compute the images of the prevertices on the rectangle. Even though some may be crowded, they will all be found with high accuracy relative to the overall size of the rectangle. Thus the conformal modulus can be found accurately as well.

To produce Figure 3.9, we ran Shortcut CRDT on the polygon, which after splitting had 87 vertices. This produced the solution vector σ to a residual tolerance of 10^{-8} after just 27 function evaluations. Then we used the procedure above to find the affine constants for the standard and rectified maps, and the rectangle prevertices. Note that the triangulation defined on P is still a triangulation of the rectangle, albeit with many degenerate triangles that lie on the long rectangle

edges. Suppose the rectangle line we wish to map is separated from the rectangle wall by a distance h. In the image plane of f_i, that separation scales to $h/|B_i|$. If $h/|B_i| \ll 1$, the image of the line will be very close to the boundary of P in the vicinity of Q_i. If Q_i is degenerate, its raw image is a line segment, and it is also possible that $h/|B_i| \gg 1$. In fact, this means that Q_i is in a deadwater region and the image of the line will be far from Q_i. In sum, only for those embeddings in which h is comparable to $|B_i|$ do we need to track the image of the line. This is accomplished by choosing points on the portion of the line local to Q_i, inverting the rectified map as in Section 2.3, and computing the standard forward map from the same embedding.

We believe that this technique can be adapted to perform grid generation for any polygon. The main obstacle is in defining the rectified map in general, when a rectangle is not a natural choice.

3.7.2 Solution of a boundary value problem

Our second example of applying CRDT involves the solution to a certain boundary value problem related to harmonic measure. Our BVP is Laplace's equation $\Delta u = 0$ with Dirichlet boundary conditions on a polygonal domain P. There is one edge s of the polygon, which we call the "forced edge," with boundary condition 1, and the solution on the rest of the boundary is zero. We want to find u only at a particular point x in the interior of P. Let the endpoints of s be denoted w_p and w_q, in counterclockwise order.

Mathematically the problem is easily solved. Let $\varphi : D \to P$ be a Schwarz–Christoffel map that has $\varphi(0) = x$. The solution u of the original problem maps to a Laplace solution $\hat{u}$ on the unit disk via the formula $\hat{u}(z) = u(\varphi(w))$. This Laplace problem has boundary condition one on the segment $\varphi^{-1}(w_p)$ to $\varphi^{-1}(w_q)$

and zero elsewhere. Because the solution to Laplace's equation at the center of a disk is identically the average of the boundary Dirichlet values, we have that $u(x)$ is precisely the arc-length distance θ measured in radians between $\varphi^{-1}(w_p)$ and $\varphi^{-1}(w_q)$, scaled by $1/2\pi$.

An obvious algorithm is to compute the prevertices of the disk map and read off the angular distance between prevertices p and q. In fact, it suffices to compute $h = |\varphi^{-1}(w_p) - \varphi^{-1}(w_q)|$. For, given the Euclidean distance h between two points on the unit circle, we can get the angular distance via

$$\theta = 2\sin^{-1}(h/2). \tag{3.7}$$

However, this direct method will fail when the forced edge s is separated from x by a long, thin region. In this case, the value $u(x)$ will be extremely close to zero, and cancellation error will dominate h. Yet we insist on computing $u(x)$ accurately in a *relative* sense. Note that it suffices to compute h with high relative accuracy, since (3.7) can be evaluated accurately when h is very small (indeed, (3.7) behaves like $\theta \approx h$ for small h).

If the domain P is singly elongated, a solution to the BVP is possible using Schwarz–Christoffel maps from a rectangle [47]. Recall that a rectangle is transformed to the upper half-plane by the elliptic function $\mathrm{sn}(z|m)$, where m is obtained in the SC solution. We can further construct a Möbius transformation that maps the half-plane image of x to the center of the unit disk, and then accurately measure h. Of course, such a solution is not available when P is multiply elongated and the rectangle map fails.

We assume throughout that x is not too close to any edge of the polygon. If x is close to an edge, this creates a different problem with relative accuracy that is not addressed by the techniques in this section, though it could be addressed by other means.

We start by running CRDT on P and computing the affine constants as above. Let Q_1 be a quadrilateral having s as an edge. (We assume for now that s is an unsplit edge of the original polygon and deal with the case of split s below.) Let Q_m be a quadrilateral in P that contains the point x. Note that we can accurately invert the Schwarz–Christoffel map using embedding E_m to find $\xi = f_m^{-1}(x)$. Furthermore, ξ will not be close to the boundary of the disk.

Let $Q_1, Q_2, \ldots, Q_m$ be a path in the quadrilateral graph. In the embedding E_2, we can compute the distance $|z_p - z_q|$ accurately, because either z_p or z_q is a vertex of Q_2. Thus we can compute the cross-ratio ρ_2 of z_p, z_q, and two more prevertices of Q_2, including the prevertex that is not shared with Q_1. Now consider the embedding E_3. In this embedding, z_p and z_q may be crowded. However, we still know the cross-ratio ρ_2 of z_p, z_q, and two prevertices of Q_3. Hence of the four lengths that are factors in the formula for $|\rho_2|$, only one may be small, so we can compute it accurately using ρ_2 and the other lengths. By the same token, we can accurately compute ρ_3, the cross-ratio of z_p, z_q, and two other prevertices of Q_3, including the one not shared by Q_2. As in the previous example, we are computing a very small quantity (given by (3.7)) by repeated multiplication of small numbers rather than by subtracting two nearby complex numbers.

We can continue this procedure through ρ_m, a cross-ratio involving z_p, z_q, and two prevertices of Q_m. Finally, we find the image of these points under the transformation of the type (3.4) that moves ξ to the origin. Since ξ is not close to the boundary of the disk, the well-separated points will remain uncrowded. The invariance of ρ_m under fractional linear transformation allows us to recover $|z_p - z_q|$, which is now the desired h.

In the situation where the segment s is split by CRDT, we simply add up the contributions to $u(x)$ separately from each subsegment using the preceding algorithm. A Laplace solution is linear in the boundary data, and there cannot be

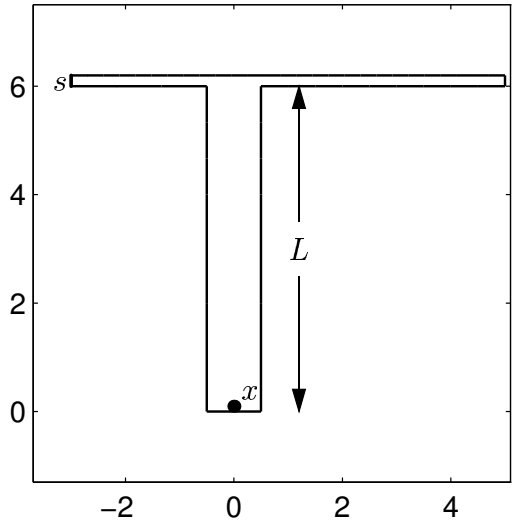


Figure 3.10: The BVP of Section 3.7 is solved on this T-shaped region. The length L is allowed to vary, and the forced edge is denoted by s. The solution is harmonic in the region and zero on the boundary. The point x at which the solution is sought is at $0.1i$. The dimensions of the horizontal bar are 8×0.2.

any cancellation at this step because each contribution is positive.

We use the method described above to solve the BVP on the T-shaped region of Figure 3.10. The region is parameterized by L, the length of the long arm of the T, and the forced edge s lies at the end of the shortest arm. The point x lies along the centerline of the long arm and at a distance 0.1 from the end.

In Figure 3.11 we plot the solution $u(x)$ for L up to 20. After a transient phase (shown in the inset), the behavior very quickly approaches $ke^{-\pi L}$ for some constant k. This is because as L grows, the influence of the configuration at the top of the T becomes negligible, and the solution is like that for a rectangle with forced edge at the top. The rectangle aspect ratio L is asymptotically related to the sn parameter m by

$$L = \frac{1}{\pi} \ln \left(\frac{4}{\sqrt{m}} \right) + O(m).$$

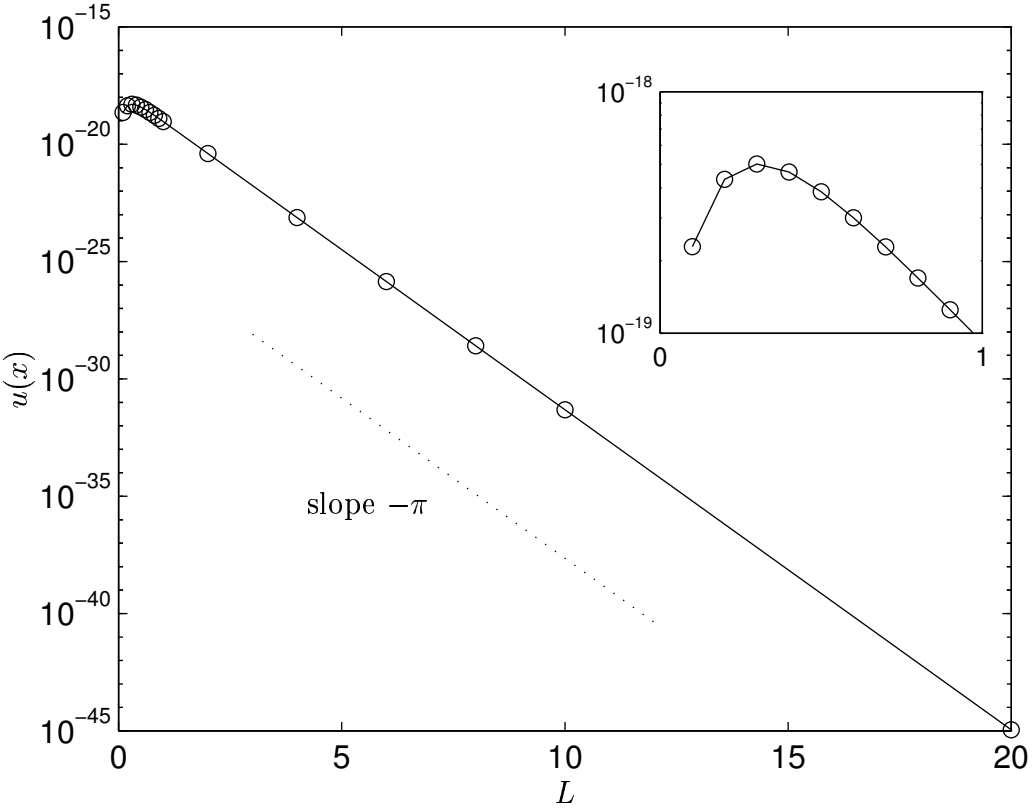


Figure 3.11: Solution of the BVP for the region in Figure 3.10. The inset shows the solution at points for $L < 1$. As L grows, the curve quickly approaches $ke^{-\pi L}$, corresponding to the leading-order behavior of the solution for a rectangle.

By further transformation to the unit disk, we find that to leading order, $u(x) \approx \sqrt{m} \approx e^{-\pi L}$.

One issue that arises immediately is verification of the computed solutions. We do this by noting that the BVP is essentially singly elongated, even when P is not; the elongations not containing x nor the forced edge are largely irrelevant to the solution. For the T-shaped polygon in Figure 3.10, for example, we can shrink the right branch of the crossbar, decreasing the BVP solution by an exponentially small

amount. Once this branch is sufficiently small, we can compute a rectangle map and solve the approximate BVP as described above. We have done this for several values of L and verified the CRDT solutions to at least 10 digits. We have also confirmed 12 digits for BVPs on other regions. The advantage of the CRDT method is that the solution is found for the original region directly, without introducing any approximations, and that the method can map to polygons that require a great deal of user intervention via continuation in the SC Toolbox.

The two applications presented in this section share the idea of composing a chain of operations via a path in the quadrilateral graph. By following such a path, we can take advantage of the overlap between neighboring quadrilaterals to ensure that each link in the chain is fairly well-conditioned, even though the global results of following the chain may vary over many orders of magnitude. We believe that this technique is central to the application of CRDT to potential theory problems.

3.8 Summary

We have introduced CRDT, a new algorithm for finding the Schwarz–Christoffel prevertices on the unit circle for arbitrary bounded polygonal regions. The classical crowding problem is avoided through conformally equivalent re-embeddings of the prevertices so that the numerical mapping is always locally accurate. The embeddings are defined via an overlapping decomposition so that neighboring embeddings are uniquely related. The nonlinear system chosen for numerical solution apparently has a monotonicity property that makes it easier to solve numerically than the systems posed by the SC Toolbox, although we cannot verify that this is so for all polygons. While we conjecture that the CRDT algorithm will always converge to the correct solution, we have been unable to prove this.

The polygon is first split so that long, narrow regions can be represented piece-

wise by well-conditioned triangles. A Delaunay triangulation of the resulting n-vertex polygon is computed and used to define $n-3$ quadrilaterals, whose diagonals appear as internal sides in the triangulation. The primitive variables of the nonlinear system are logarithms of the cross-ratios of the prevertices of those $n-3$ quadrilaterals. These cross-ratios define an infinite set of conformally equivalent configurations of the prevertices, each of which produces an S-C map to the same image polygon. The imposed constraints are on the magnitudes of the cross-ratios of the quadrilaterals in the image polygon.

The CRDT algorithm generally compares favorably with the SC Toolbox in numerical experiments. The principal exceptions are those regions which require a great many extra vertices to be added in the splitting phase of the algorithm. We do not know if there is a more effective splitting procedure. On the other hand, CRDT has no difficulty finding the prevertices for arbitrarily elongated polygons, something which no previous algorithm can claim.

We demonstrate the use of CRDT in applications. Figure 3.9 shows the rectangle map to a multiply elongated polygon for which we believe no other existing algorithm would work. We also illustrate how to use CRDT to solve a particular elliptic boundary value problem. In each case, the key is to follow a path in the quadrilateral graph of the polygon, derived from the dual of the triangulation. When this is done, the distance separating prevertices can be computed without cancellation, even when the distance is extraordinarily small.

Once some of the unresolved matters already discussed have been more satisfactorily settled, especially issues of using CRDT for grid generation, CRDT will be included as part of the SC Toolbox. In addition, there are open questions about possible extensions of CRDT to polygons with infinite vertices, circular-arc polygons [45], or multiply-connected regions [19, 48].

Chapter 4

A nonoverlapping method for Symm's equation for conformal mapping

Numerical methods for the Schwarz–Christoffel formula are appealing because of their speed and high accuracy. However, they are fundamentally limited by their restriction to polygons. While solutions of generalizations of the Schwarz–Christoffel formula for circular-arc polygons [10, 45] and even more general regions [20, 27, 39, 40, 83, 89] have been devised, the methods lose much of their simplicity, speed, and reliability.

There are many methods for conformal mapping that apply to a simply connected region bounded by a general piecewise analytic curve [30, 35, 39]. Most of these are based on integral equations. The best-known is due to Symm [49, 79] and has been efficiently and elegantly implemented as the public-domain package `CONFPACK` [42].

Symm's method deals primarily with the map Riemann map g from a simply

connected region Ω to the unit disk. This has two consequences. The first is that a linear equation can be derived, as seen from the Green's function for Ω. The Green's function with pole at $z = \zeta$ is proportional to $-\log|g(z)|$, where $g(\zeta) = 0$. On the other hand, it is also proportional to $-\log|z - \zeta|$ plus a function harmonic in Ω. Thus the problem of finding g is reduced to the linear problem of finding this harmonic remainder. The second significant consequence of working with g rather than $g^{-1} = f$ as in eariler chapters is that the impact of crowding is more manageable. In a crowded portion of Ω, f' is very large, but g' is very small, and g can still be found accurately *relative to the global map.*

There has been little study of domain decomposition in conformal mapping [64, 65], and apparently none relating to integral equations. This is true despite the fact that Symm's method is essentially an indirect boundary element method, and domain decomposition is common in boundary element methods generally [12, 51]. Any region with a long, narrow channel, or with structures on different scales, is a candidate for decomposition. The basic idea is to introduce a few extra unknowns on the interfaces between subregions in order to introduce sparse structure into the system matrix, corresponding to a localization of the boundary integrals. An additional benefit is that deadwater regions can be treated once, while regions of greater activity can be resolved adaptively with a minimum of interaction with other areas.

In this chapter we introduce Symm's interior equation and briefly summarize existing numerical methods, including `CONFPACK`. We then show how to incorporate domain decomposition in such a way that the accurate representation of corners in `CONFPACK` is still applicable. We experiment with an implementation of the domain decomposition method for polygons for both fixed and adaptive strategies.

4.1 Symm's equation

We begin with a summary of the mathematics behind Symm's method. This material can be found in more detail in, e.g., [39], Section 16.6.

Let Ω be a simply connected open region in $\mathbf{C}$ such that $\Gamma = \partial\Omega$ is a finite collection of analytic curves, and let $z_0 \in \Omega$. We do not allow Γ to have interior or exterior cusps. Let g be a conformal map of Ω to D, the unit disk, that maps z_0 to the origin. (The choice of g has one rotational degree of freedom.)

Because $g'(z_0) \neq 0$, the function $g(z)/(z - z_0)$ is nonzero throughout Ω.[1] Thus $g(z)/(z - z_0)$ has a logarithm in Ω, so we can write

$$g(z) = (z - z_0) \exp\left(u(z) + iv(z)\right), \tag{4.1}$$

where u and v are real harmonic conjugates. Furthermore, since $|g(z)| = 1$ for $z \in \Gamma$, the function u is a solution of the BVP

$$\Delta u(z) = 0, \qquad z \in \Omega, \tag{4.2a}$$

$$u(z) = -\log|z - z_0|, \qquad z \in \Gamma. \tag{4.2b}$$

Because u is harmonic, we can express it in terms of a single-layer potential [12]:

$$u(z) = -\frac{1}{2\pi} \int_\Gamma \sigma(\zeta) \log|z - \zeta| \, d\Gamma. \tag{4.3}$$

By considering the known values of u on the boundary, we arrive at *Symm's equation* for interior mapping,

$$\log|z - z_0| = \frac{1}{2\pi} \int_\Gamma \sigma(\zeta) \log|z - \zeta| \, d\Gamma, \qquad z \in \Gamma. \tag{4.4}$$

[1] In Symm's method, the emphasis is on finding a map from a region to a disk. Thus we use the z-plane to denote the polygon and the w-plane for the disk, reversing the notation of earlier chapters.

This integral equation of the first kind can always be solved uniquely for the potential σ, provided the capacity of Γ is not equal to one [31]. In fact, σ has a special meaning—it is closely related to the *boundary correspondence function*, which is defined as follows. Suppose that $\zeta = \zeta(s)$, $0 \leq s \leq L$, is a piecewise analytic parameterization of the boundary Γ. The boundary correspondence function is defined to be

$$\theta(s) = \arg(g(\zeta(s))), \tag{4.5}$$

for any choice of the arg that makes $\theta(s)$ continuous. The potential σ, regarded as a function of s, is

$$\sigma(s) = |\zeta'(s)|\theta'(s), \tag{4.6}$$

where $\zeta'(s)$ can be taken as an arbitrary finite number at a corner of Γ. If arc length is used to parameterize the boundary, σ and θ' are identical. Equation (4.6) is useful in the construction of accurate numerical methods, as we shall see.

Once σ is known, the conjugate pair u and v can be found up to an additive constant in v by replacing $\log|z-\zeta|$ by $\log(z-\zeta)$ in (4.3), taking care to make the logarithm continuous in Ω. This in turn determines g by (4.1), up to a rotation of the unit disk.

There are other integral equation formulations for interior conformal mapping [30, 39], including those of Berrut [9], Warschawski [77], Gerschgorin [39], and those based on the Szegő kernel [52, 63]. There are also methods for the inverse mapping, including conjugate function approaches [35] such as Theodorsen's [87] method and Wegmann's method [88].

4.2 Numerical solutions of Symm's equation

To approximate the solution of the integral equation (4.4), the typical procedure is to discretize the boundary, choose a representation for the approximation to σ, and replace the integral equation by a linear system of integral equations. This is the approach originally taken by Symm [79], who chose a piecewise constant representation for σ. Hayes *et al.* [36] use a piecewise quadratic approximation. Hough and Papamichael [41, 43] use information about the behavior of $\theta'(s)$ at the corners of Γ to demonstrate that Jacobi weight functions give the correct singular behavior of σ at the corners. This is the approach implemented in `CONFPACK` [42].

All these methods can be viewed as indirect boundary element methods [12]. In most contexts, indirect methods are not as popular as the direct methods, because the potential σ being solved for has no direct physical meaning. However, in this context, the use of σ allows the representation of corner singularities by the Jacobi weight method of `CONFPACK`, which greatly increases the accuracy of the method near corners.

Other approaches are possible. Berrut [8] and Reichel [70] use FFT-based iterative methods to solve (4.4). Amano [1] uses a charge-simulation method in which point charges outside the domain are used to approximate the solution to (4.2).

4.3 Domain decomposition

If two portions of Ω are separated by a long, thin channel, the influence of data on one end of the channel may be exceedingly weak at the other end. Thus there seems little point in having to compute the influence of all of Γ on all of Γ. By introducing an interface in the channel, we aim to concentrate the influence of the far region there, relieving us of the need to find this effect explicitly.

In terms of the linear system resulting from discretization, we hope to introduce a few additional unknowns along the interface that give the system matrix a block sparsity structure. In fact, the structure we shall arrive at is the Schur complement form, which is familiar from domain decomposition in boundary value problems [14]. This structure will yield large computational savings in both setting up and solving the system.

A natural first idea would be to introduce potentials on the interface as we did on the boundary. A potential on each "side" of Γ would be mathematically necessary to keep the system square. Thus, for $k = 1, 2$, we could define σ_k on all of Γ_k, define a function u_k on Ω_k by (4.3), and supplement equation (4.4) with equations requiring the u_k and their normal derivatives to match along $\Gamma_1 \cap \Gamma_2$.

This is fairly easy to do with piecewise constant boundary elements, for example, and the resulting linear system is indeed ready for the Schur complement form. But the resulting σ_k have no obvious relationship to the global potential σ. One can still use the individual potentials to compute g in the subregions, but the boundary correspondence function is not known. More seriously, the new potentials σ_k do not retain the same behavior at the corners as σ, and therefore the specialized treatment of corners used by `CONFPACK` cannot be applied. This makes calculations of high accuracy difficult. We now derive an alternative formulation that avoids this problem.

4.3.1 Integral equations

In what follows, we adopt the notation

$$G(z, \zeta) = \frac{1}{2\pi} \log |z - \zeta| \tag{4.7}$$

for the fundamental solution for the Laplace operator.

Let

$$\tilde{u}(z) = u(z) + \log|z - z_0|. \tag{4.8}$$

From (4.2) we see that

$$\Delta\tilde{u}(z) = 2\pi\delta(z - z_0), \qquad z \in \Omega, \tag{4.9a}$$

$$\tilde{u}(z) = 0, \qquad z \in \Gamma. \tag{4.9b}$$

Hence $\tilde{u}(z)$ is a multiple of the Green's function for Ω with pole at z_0. We shall now apply a direct integral equation to solve for $\tilde{u}$ [51]. Let $z \in \Omega$. By Green's second identity,

$$\int_\Omega \varphi\Delta\psi \, d\Omega - \int_\Omega \psi\Delta\varphi \, d\Omega = \int_\Gamma \varphi\frac{\partial\psi}{\partial n} \, d\Gamma - \int_\Gamma \psi\frac{\partial\varphi}{\partial n} \, d\Gamma, \tag{4.10}$$

with $\psi = \tilde{u}(\zeta)$ and $\varphi = G(z, \zeta)$, we have

$$\log|z - z_0| - \tilde{u}(z) = \int_\Gamma G(z, \zeta)\frac{\partial\tilde{u}}{\partial n}(\zeta) \, d\Gamma.$$

Now suppose z approaches a point on Γ. The $\tilde{u}(z)$ term on the left will need to be multiplied by a constant depending on the angle subtended by Γ at the limit point, but in any case, $\tilde{u}(z) \to 0$. Thus we have

$$\log|z - z_0| = \int_\Gamma G(z, \zeta)\frac{\partial\tilde{u}}{\partial n}(\zeta) \, d\Gamma. \tag{4.11}$$

By comparison to (4.4), we see that $\partial\tilde{u}/\partial n$ is equal to the potential σ.

We wish to find $\partial\tilde{u}/\partial n$ on Γ using domain decomposition. Let us suppose that Ω is divided into two nonoverlapping regions Ω_1 and Ω_2, and let Γ_1 and Γ_2 be the positively oriented boundaries of these subregions. The interface Γ_{12} is equal to $\Gamma_1 \cap \Gamma_2$, and is taken to be oriented positively with respect to Ω_1. Assume without loss of generality that $z_0 \in \Omega_1$. See Figure 4.1.

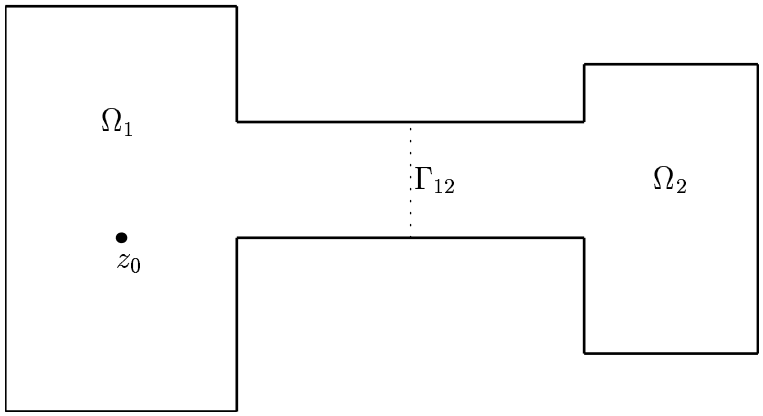


Figure 4.1: Typical domain decomposition for the method of this chapter.

By applying Green's identity (4.10) to the subregions, inserting (4.9), and letting $z \in \Gamma$, we arrive at the system

$$c_1(z)\tilde{u}_1(z) - \log|z - z_0| = \int_{\Gamma_1} \tilde{u}_1(\zeta)\frac{\partial G}{\partial n}(z,\zeta)\,d\Gamma_1 - \int_{\Gamma_1} G(z,\zeta)\frac{\partial \tilde{u}_1}{\partial n}(\zeta)\,d\Gamma_1, \tag{4.12a}$$

$$c_2(z)\tilde{u}_2(z) = \int_{\Gamma_2} \tilde{u}_2(\zeta)\frac{\partial G}{\partial n}(z,\zeta)\,d\Gamma_2 - \int_{\Gamma_2} G(z,\zeta)\frac{\partial \tilde{u}_2}{\partial n}(\zeta)\,d\Gamma_2, \tag{4.12b}$$

where $\tilde{u}_{1,2}$ represent the values of $\tilde{u}$ on the two subregions and $c_k(z)$ is the interior angle subtended by Γ_k about z, normalized by 2π. (For z on a smooth portion of Γ_k, $c_k(z) = \frac{1}{2}$.) Of course, $\tilde{u}_1$ and $\tilde{u}_2$ are not independent; they are related by

$$\tilde{u}_1 = \tilde{u}_2, \tag{4.13a}$$

$$\frac{\partial \tilde{u}_1}{\partial n} = -\frac{\partial \tilde{u}_2}{\partial n} \qquad \text{on } \Gamma_{12}. \tag{4.13b}$$

We have replaced the Fredholm equation of the first kind (4.4) with a system of Fredholm equations of the second kind. The kernel $\partial G/\partial n$ is equal to the *Neumann kernel* given by [39]

$$\frac{\partial G}{\partial n}(z,\zeta(s)) = \frac{1}{2\pi}\,\mathrm{Im}\left(\frac{\zeta'(s)}{|\zeta'(s)|(\zeta(s)-z)}\right),$$

for any parameterization $\zeta(s)$ of a boundary. This kernel is nominally more singular than the logarithmic one arising from the fundamental solution. However, $\tilde{u}_i(\zeta) = 0$ for $\zeta \in \Gamma_i \backslash \Gamma_{12}$; furthermore, if we assume that the interface is a straight line segment, then the kernel will be zero on Γ_{12} when $\zeta, z \in \Gamma_{12}$. Hence no singular integrals need be computed. Note that this essentially returns (4.12) to a system of equations of the first kind.

4.3.2 Representation of solutions

We now recall and extend the representation of unknowns in Symm's equation described by Hough [41, 42]. Let $\zeta(s)$ be a local parameterization of a typical analytic arc of Γ with endpoints $\zeta(-1) = z_-$ and $\zeta(1) = z_+$ such that $|\zeta'(s)|$ is bounded and never zero. Let Γ have exterior turning angles (as defined in Chapter 2) $\pi\beta_-$ and $\pi\beta_+$ at the endpoints, respectively. We represent $\partial\tilde{u}/\partial n$ on this arc by

$$\frac{\partial \tilde{u}}{\partial n}\big(\zeta(s)\big) = \phi(s)w(s) = \phi(s)(1+t)^{\alpha_-}(1-t)^{\alpha_+}, \tag{4.14}$$

where the *Jacobi indices* $\alpha_\pm$ are given by

$$\alpha_\pm = -1 + \frac{1}{1+\beta_\pm}.$$

Hough [41] has shown that while $\partial\tilde{u}/\partial n$ is only in $L^2[-1,1]$ and is infinite at a reentrant corner, ϕ is Hölder continuous on $[-1,1]$ with index greater than $\frac{1}{2}$.

We approximate ϕ by

$$\phi(s) \approx \sum_{j=0}^{m-1} q_j P_j(s), \tag{4.15}$$

where the P_j's are Jacobi polynomials associated with the weight function $w(s)$. To match the m unknown coefficients, we choose m collocation points z_j. These are $\zeta(s_j)$, $j = 1, \ldots, m$, where s_j is the jth zero of P_m.

We also need a representation of $\tilde{u}$ on the interface Γ_{12}. Let $\zeta(s)$ now be a parameterization of the interface. It seems natural to suppose that the appropriate weight function will be as in (4.14) with the Jacobi indices increased by one. So we define

$$\tilde{u}(\zeta(s)) = \psi(s)(1+t)^{1+\alpha_-}(1-t)^{1+\alpha_+}. \tag{4.16}$$

We now prove that ψ is as smooth as ϕ. This theorem is the analog of Proposition 2.1 in [41].

Theorem 4.1. *With the notation of this section, the quotient function $\psi(s)$ is in $H^\nu[-1,1]$, where ν is given by*

$$\nu = \begin{cases} \min\big((1+\beta_-)^{-1}, (1+\beta_+)^{-1}\big), & \text{if } \max(\beta_-,\beta_+) > 0, \\ 1, & \text{if } \max(\beta_-,\beta_+) < 0, \\ 1-\epsilon, & \text{for any } \epsilon > 0, \text{ otherwise.} \end{cases}$$

Proof. Let $\pi\gamma$ be the interior angle at the left endpoint z_-; i.e., $\gamma = 1+\beta_-$. From [66] we have two expansions of $g(z)$ near z_-:

(i) If $\gamma = p/q$, with p and q relatively prime, then

$$g(z) - g(z_-) = \sum_{j,k,l} B_{jkl}(z-z_-)^{j+k/\gamma}\big(\log(z-z_-)\big)^l, \tag{4.17a}$$

where j, k, and l run over integers such that $j \geq 0$, $1 \leq k \leq p$, and $0 \leq l \leq q$, and where $B_{010} \neq 0$.

(ii) If γ is irrational, then

$$g(z) - g(z_-) = \sum_{j,k} B_{jk}(z-z_-)^{j+k/\gamma}, \tag{4.17b}$$

where $j \geq 0$, $k \geq 1$, and $B_{01} \neq 0$.

Note from (4.1) and (4.8) that $\tilde{u}(z) = \log|g(z)|$. Hence for z near z_-,

$$\tilde{u}(z) = \log|g(z_-) + S(z)| \leq |S(z)| + O\big(|S(z)|^2\big),$$

where $S(z)$ represents the sum on the right-hand side of the appropriate member of equations (4.17). We set $z = \zeta(s)$ and observe that $|\zeta(s) - \zeta(-1)|$ and $(s+1)$ have the same asymptotic behavior. We wish to factor out $(s+1)^{1/\gamma}$ from $S(\zeta(s))$ and show that the quotient is appropriately well-behaved for s near -1; i.e., it is in $H^\nu[-1, -1+\delta]$ for some $\delta > 0$. We consider three cases.

(a) Suppose $1 < \gamma < 2$. If γ is irrational, then clearly $(s+1)^{1/\gamma}$ is the most singular term after the factorization. But the same is true if $\gamma = p/q$, since necessarily $p > 1$ and the B_{020} term appears in (4.17a). We observe that $(s+1)^{1/\gamma}$ is in $H^{1/\gamma}[-1, -1+\delta]$ for any $\delta > 0$.

(b) Suppose $\gamma = 1$. Then from (4.17a) we see that the most singular term in the quotient is $(s+1)\log(s+1)$, which is in $H^{1-\epsilon}$ for any $\epsilon > 0$.

(c) Suppose $0 < \gamma < 1$. If γ is irrational, again it is clear that $(1+s)^{1/\gamma}$ is the most singular term. If $\gamma = p/q$, then $(1+s)$ is the most singular term, because $q > 1$ and the logarithmic terms in (4.17a) cannot appear until $j \geq 2$. In either case, the quotient is in H^1.

Similar reasoning applies near z_+. □

The natural representation of ψ is analogous to (4.15), with the Jacobi polynomials being taken with respect to the new weight function in (4.16), that is, with Jacobi indices increased by 1.

We have seen that on each analytic arc of Γ_k, we have unknown coefficients from the Jacobi expansion (4.15) defining $\partial\tilde{u}/\partial n$ on that arc. These coefficients are globally collected into a vector $q^{(k)}$. Let us suppose the elements of $q^{(k)}$ are ordered

such that the coefficients relating to values on Γ_{12} are last, so that

$$q^{(k)} = \begin{bmatrix} q_p^{(k)} \\ q_s^{(k)} \end{bmatrix}.$$

(The subscripts are meant to convey "private" and "shared.") Let the length of $q_p^{(k)}$ be N_{ip}, the length of $q_s^{(k)}$ be N_{ks}, and N_k be $N_{kp} + N_{ks}$. We also have the N_{ks}-vector $v_s^{(k)}$ defining $\tilde{u}_k$ on Γ_{12}. (For symmetry of notation, we can define $v_p^{(k)}$, all of whose entries are zero.) Similarly, we define $z^{(k)}$ as a vector of N_{kp} collocation points on $\Gamma_k \setminus \Gamma_{12}$ and N_{ks} collocation points on Γ_{12}. We require that $N_{1s} = N_{2s} = N_s$, and that corresponding entries of the s-vectors have the same meaning globally on Γ_{12}.

4.3.3 Numerical method

We are now prepared to state the discretized form of (4.12). Each integral on the right-hand side of the equations is approximated by a matrix operating on $q^{(k)}$ or $v^{(k)}$ to produce a vector of the integral evaluated at the collocation points $z^{(k)}$. We denote these matrices by $A^{(k)}$ and $B^{(k)}$, respectively. Each identity operator on the left-hand side of the equations is also approximated by a matrix $\tilde{I}^{(k)}$ operating on $v^{(k)}$; because of the representation chosen, these will not be identity matrices. All the matrices can be written in block form to yield

$$\begin{bmatrix} B_{pp}^{(k)} & B_{ps}^{(k)} \\ B_{sp}^{(k)} & B_{ss}^{(k)} \end{bmatrix} \begin{bmatrix} q_p^{(k)} \\ q_s^{(k)} \end{bmatrix} - \begin{bmatrix} A_{pp}^{(k)} - \frac{1}{2}\tilde{I}_{pp}^{(k)} & A_{ps}^{(k)} \\ A_{sp}^{(k)} & A_{ss}^{(k)} - \frac{1}{2}\tilde{I}_{ss}^{(k)} \end{bmatrix} \begin{bmatrix} 0 \\ v_s^{(k)} \end{bmatrix} = \begin{bmatrix} b_p^{(k)} \\ b_s^{(k)} \end{bmatrix},$$

where $b^{(1)} = \log|z^{(1)} - z_0|$ and $b^{(2)} = 0$. Singular integrals for the B matrices, which involve boundary integrals with the kernel $\log|z - \zeta|$, are performed as in `CONFPACK`, splitting out the singularity by subtraction and making calls to `QUADPACK`. Nonsingular integrals are calculated by compound Gauss–Jacobi integration. The integrals for the A matrices need be computed only over the interface Γ_{12}. As

was pointed out above, no singular integrals are necessary as long as Γ_{12} is a line segment, since in that case $A_{ss}^{(k)} = 0$.

We shall now pose a single system. Relations (4.13) imply

$$\begin{aligned} v_s^{(1)} &= v_s^{(2)}, \\ q_s^{(1)} &= -q_s^{(2)}. \end{aligned}$$

All together we have

$$\begin{bmatrix} B_{pp}^{(1)} & 0 & B_{ps}^{(1)} & -A_{ps}^{(1)} \\ 0 & B_{pp}^{(2)} & -B_{ps}^{(2)} & -A_{ps}^{(2)} \\ B_{sp}^{(1)} & 0 & B_{ss}^{(1)} & \frac{1}{2}\tilde{I}_{ss}^{(1)} \\ 0 & B_{sp}^{(2)} & -B_{ss}^{(2)} & \frac{1}{2}\tilde{I}_{ss}^{(2)} \end{bmatrix} \begin{bmatrix} q_p^{(1)} \\ q_p^{(2)} \\ q_s^{(1)} \\ v_s^{(1)} \end{bmatrix} = \begin{bmatrix} b_p^{(1)} \\ 0 \\ b_s^{(1)} \\ 0 \end{bmatrix}. \tag{4.18}$$

This matrix is in Schur complement form. The advantage of this form is significant when N_s is much smaller than N_{1p} and N_{2p}. In this case, the system (4.18) is solved by first solving for $q_p^{(1)}$ and $q_p^{(2)}$ from the first and second block rows respectively, and substituting into the equations for the interface variables:

$$\begin{aligned} \left(\begin{bmatrix} B_{ss}^{(1)} & \frac{1}{2}\tilde{I}_{ss}^{(1)} \\ -B_{ss}^{(2)} & \frac{1}{2}\tilde{I}_{ss}^{(2)} \end{bmatrix} - \begin{bmatrix} B_{sp}^{(1)} \\ 0 \end{bmatrix} \left[B_{pp}^{(1)}\right]^{-1} \begin{bmatrix} B_{ps}^{(1)} & -A_{ps}^{(1)} \end{bmatrix} \right. \\ \left. - \begin{bmatrix} 0 \\ B_{sp}^{(2)} \end{bmatrix} \left[B_{pp}^{(2)}\right]^{-1} \begin{bmatrix} -B_{ps}^{(2)} & -A_{ps}^{(2)} \end{bmatrix} \right) \begin{bmatrix} q_s^{(1)} \\ v_s^{(1)} \end{bmatrix} \\ = \begin{bmatrix} b_s^{(1)} - B_{sp}^{(1)} \left[B_{pp}^{(1)}\right]^{-1} b_p^{(1)} \\ 0 \end{bmatrix}. \end{aligned}$$

Once $q_s^{(1)}$ and $v_s^{(1)}$ have been found, they can be substituted back to solve for $q_p^{(1)}$ and $q_p^{(2)}$.

Approximations to $\tilde{u}_1(z)$ and $\tilde{u}_2(z)$ for $z \in \Omega$ could now be obtained from (4.12), with $c_i(z) = 1$, by integration around Γ_1 or Γ_2. However, we also need the harmonic

conjugate $v(z)$ of $u(z)$ in order to compute g. It is therefore convenient to augment the integration kernels in (4.12) by their harmonic conjugates:

$$u_k(z) + iv_k(z) = l_k(z) + \frac{1}{2\pi} \int_{\Gamma_k} \left[\frac{-it_k(\zeta)}{\zeta - z} \tilde{u}_k(\zeta) - \log(z - \zeta) \frac{\partial \tilde{u}_k}{\partial n}(\zeta) \right] d\Gamma_k, \quad (4.19)$$

where $t_k(\zeta)$ is the unit tangent to Γ_k at ζ, $l_1(z) = 0$, and $l_2(z) = -\log(z - z_0)$. If z is pathologically close to Γ or Γ_{12}, we can proceed as in `CONFPACK` by analytically continuing the boundary parameterization and the representation of unknowns; see [42].

We note that the domain decomposition procedure is straightforwardly generalized to regions decomposed into three or more subdomains. One forms Schur complements for the subregions adjacent to each interface, solves for the interface quantities simultaneously, and substitutes back to find the primary quantities.

Finally, we conclude with a word about inverting the map. `CONFPACK` is able to treat the inverse map analogously to the forward map, by means of an inverse boundary correspondence function on the unit circle. We do not see a natural generalization of the domain decomposition technique to this formulation. One could use the obtained $\partial\tilde{u}/\partial n$ in the `CONFPACK` formulation. Alternatively, one could attempt to use a nonlinear iteration on the forward map, as when numerically inverting the Schwarz–Christoffel map.

4.3.4 Computational efficiency

The domain decomposition procedure requires $O\big(N_{1p}^2 + N_{2p}^2 + N_s(N_{1p} + N_{2p})\big)$ flops to form a single linear system and $O\big(N_{1p}^3 + N_{2p}^3 + N_s(N_{1p}^2 + N_{2p}^2 + N_s^2)\big)$ to solve the system by direct Schur complementation. The undecomposed version requires $O\big((N_{1p} + N_{2p})^2\big)$ flops to form and $O\big((N_{1p} + N_{2p})^3\big)$ to solve one system. Asymptotically, the domain decomposition method takes a constant fraction of the time needed by the direct solution, assuming a uniform growth in N_{1p}, N_{2p}, and N_s. However, the

constant of proportionality for forming the system is generally much larger than that of solving the system, so that the benefits of a domain decomposition in a practical serial computation may depend more on the weaker $O(N^2)$ dependence than on the $O(N^3)$ dependence.

A practical method for solving Symm's equation must be adaptive. `CONFPACK`, for example, initially chooses a uniform distribution of unknowns along the boundary arcs, solves the discretization of Symm's equation, and adjusts the number of unknowns on each arc based on the size of the coefficients in the orthogonal expansion (4.15). The system is updated and solved again, leading to an iterative process. During an update, only those columns and rows corresponding to new unknowns and collocation points need be computed, but the entire linear system must be solved from scratch.

Domain decomposition allows adaptation to proceed more efficiently. Consider the polygon in Figure 4.1, with the channel made more narrow. `CONFPACK` quickly detects that relatively few unknowns are needed on the arcs of Γ_2, but it must repeatedly solve for a linear system including those unknowns as it adds resolution to Γ_1. If a domain decomposition method is used, the block $B_{pp}^{(2)}$ need be formed and LU-factored only once. Hence a good choice of decomposition can reduce the amount of computational overhead required by deadwater regions in the adaptive solution.

Parallelization is trivially achieved in both the system formation and private block inversion stages that consume most of the computational effort. The load balancing depends on the evenness of the distribution of boundary unknowns among the subregions. This can be a delicate matter for an adaptive strategy, in which unknowns tend to concentrate near the conformal center.

Evaluation of the map at points in Ω_k requires $O(N_{kp} + N_s)$ flops, as opposed to $O(N_{1p} + N_{2p})$. Over many evaluations, this can be a substantial savings, especially

within deadwater regions where N_{kp} is ideally rather small.

4.3.5 Numerical experiments

The domain decomposition version of Symm's method described above has been implemented in MATLAB, except for singular integration against the logarithmic kernel, for which `QUADPACK` is used. For simplicity of coding, the implementation so far accepts only polygonal domains.

We begin with nonadaptive computations performed serially on a SPARC-10. Our first example is a 16×1 rectangle with 34 vertices distributed evenly around the boundary. We consider decompositions into $d = 2$, $d = 4$, and $d = 8$ equally sized subregions by splitting in the long direction. Figure 4.2 shows the CPU time required in each case, expressed as a fraction of the time needed for a single domain solution, as a function of the number of unknown Jacobi polynomial coefficients per side or interface. Even without parallelization, we see the advantage of forming and solving several smaller systems rather than a single large one. The $d = 4$ and $d = 8$ version are slower than the others for small systems, because of additional overhead, but overtake the other versions as the number of unknowns grows.

In Figure 4.3 we revisit the partially constructed Koch snowflake. In this case there is a natural decomposition into seven subregions, one of which has exclusively interface unknowns. The results of timings are shown in Figure 4.4. (These calculations make no use of symmetry, which could render the domain decomposition superfluous in this case.)

We now consider adaptive computations, performed in parallel on an IBM SP2.[2] In the adaptive strategy, the solution is initially obtained by a uniform distribution

[2]The parallel MATLAB computations were done in the MultiMATLAB environment [80].

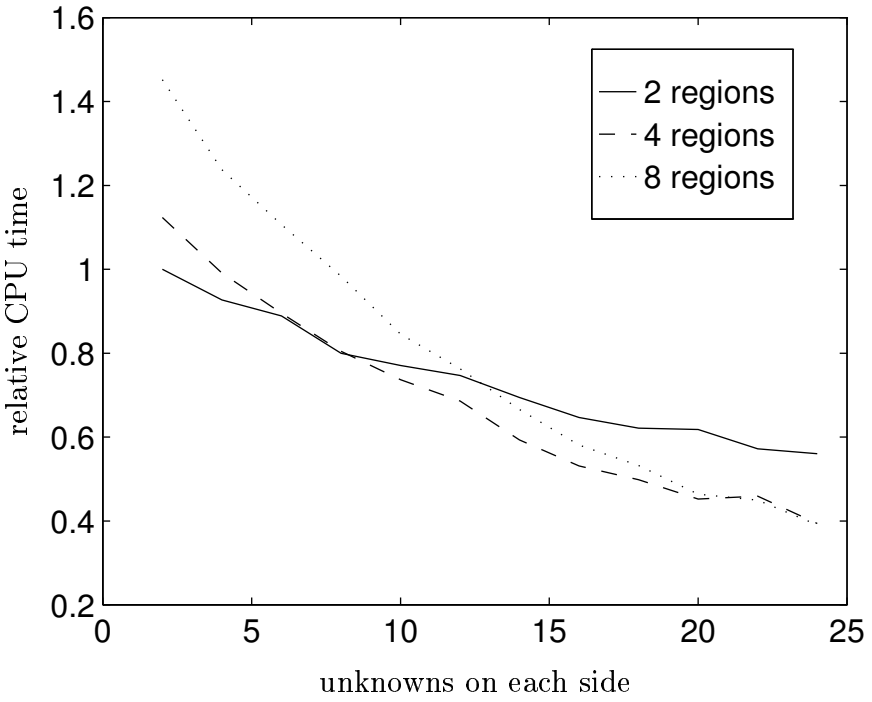


Figure 4.2: Relative CPU time for forming and solving the decomposed Symm's equation for a rectangle. The timings for decompositions into 2, 4, and 8 subdomains are shown as a fraction of the time needed by the undecomposed version.

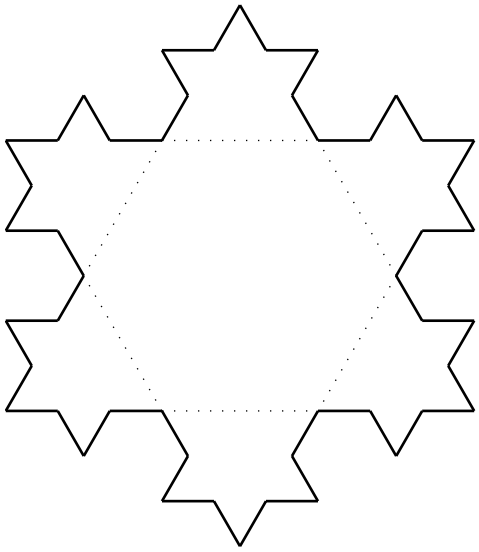


Figure 4.3: Subdivision of an early stage in the Koch snowflake.

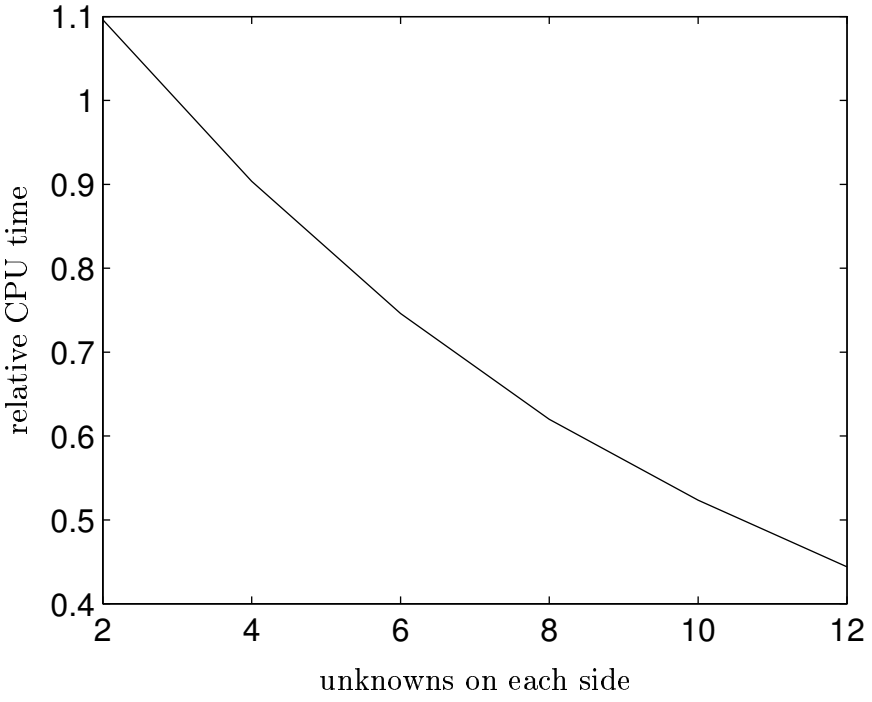


Figure 4.4: Relative CPU time for the decomposed algorithm on the Koch curve.

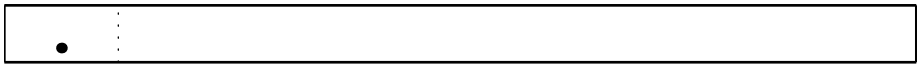


Figure 4.5: Rectangular region used in computational tests. The aspect ratio of the rectangle is 16, the conformal center is one unit to the right and a quarter unit above the lower left corner, and the interface is two units from the left edge.

of unknowns per side. Since the Jacobi polynomials are orthogonal with respect to the weight functions used, the size of the coefficients on a side is assumed to be a fair indicator of the error on that side. The number of unknowns is updated based on an exponential fit of the coefficients, and the system or systems are updated and solved. The individual subdomain matrix computations, inversions, and Schur complementation are performed in parallel, while the interface solution, subdomain back-substitutions, and adaptive refinement decisions are performed by a master process, since these steps take negligible time. We measure the total elapsed time required for a solution at each requested accuracy, and report speedup and parallel efficiency. Note that in light of Figures 4.2 and 4.4, a parallel efficiency greater than 100% is possible.

We consider the 16×1 rectangle (depicted in Figure 4.5) with conformal center one unit from the left edge and a quarter unit from the bottom edge. We put a single vertical interface two units from the left edge to achieve much better load balancing than by placing the interface in the middle of the rectangle. In Figure 4.6 we display the speedup and efficiency for several accuracies. Because the rectangle map is known exactly, we can verify that the requested accuracy is met; in fact, it is exceeded in all of these examples.

In Figure 4.7 we show a spiral region with 36 vertices and 5 subdomains. We also define $d = 3$ subdomains by eliminating the interfaces closest to the conformal center. The results are shown in Figure 4.8. Initially the subdomain containing the

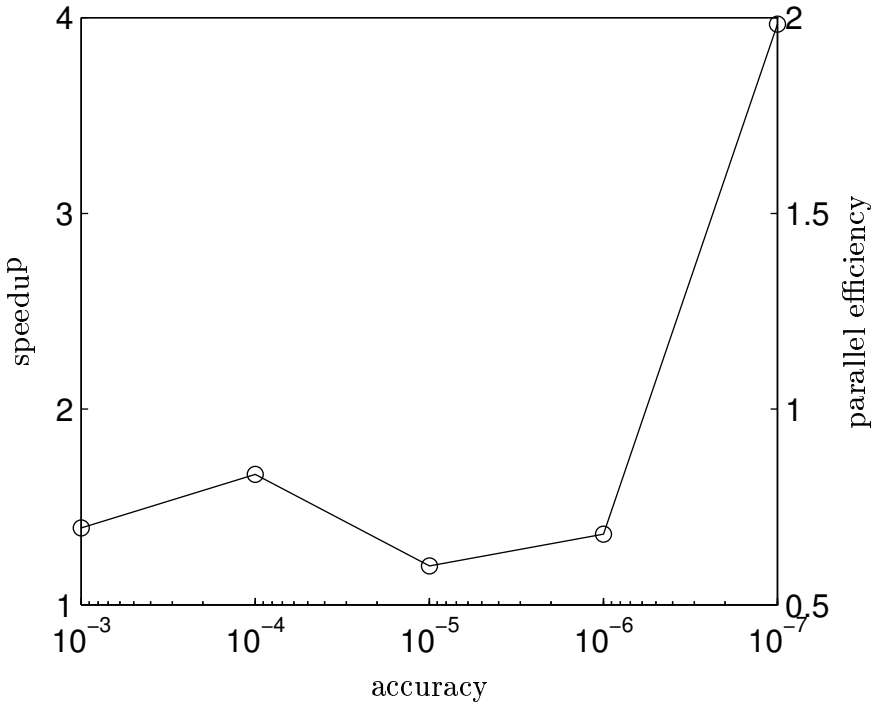


Figure 4.6: Speedup and efficiency for parallel adaptive Symm computations on the rectangle in Figure 4.5. The parallel efficiency exceeds 100% because the serial computations for the decomposed region can be faster than the undecomposed case.

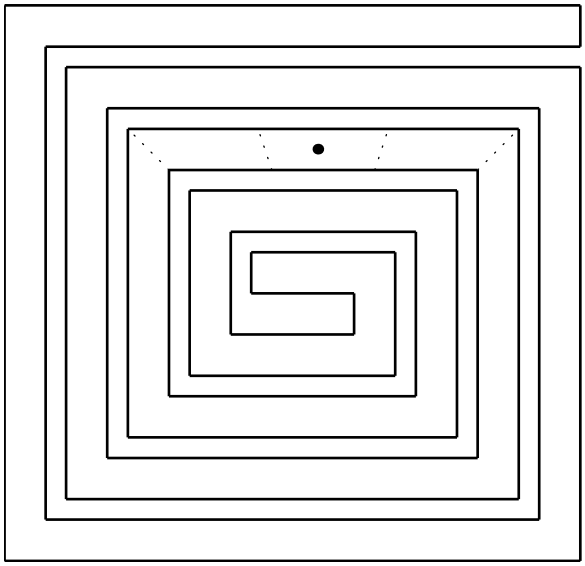


Figure 4.7: Spiral region with 36 vertices and 3 subdomains.

conformal center has relatively few unknowns, but the adaptive refinement quickly pushes the distribution into more of a balance. Parallel efficiency for $d = 3$ rises and then falls, perhaps due to poor balancing. The efficiency for $d = 5$ starts at about 25% and increases to 75%.

4.4 Summary

Symm's equation plays a major role in numerical conformal mapping. `CONFPACK` is a software package for Symm's equation on a single domain that exploits the structure of the conformal mapping problem to produce fast and accurate solutions. The use of domain decomposition, however, has not previously been explored. By working with the Green's function for the region instead of just its harmonic part, one can write a system of integral equations for a decomposed region so that the accurate representation of singularities in `CONFPACK` may still be used and extended.

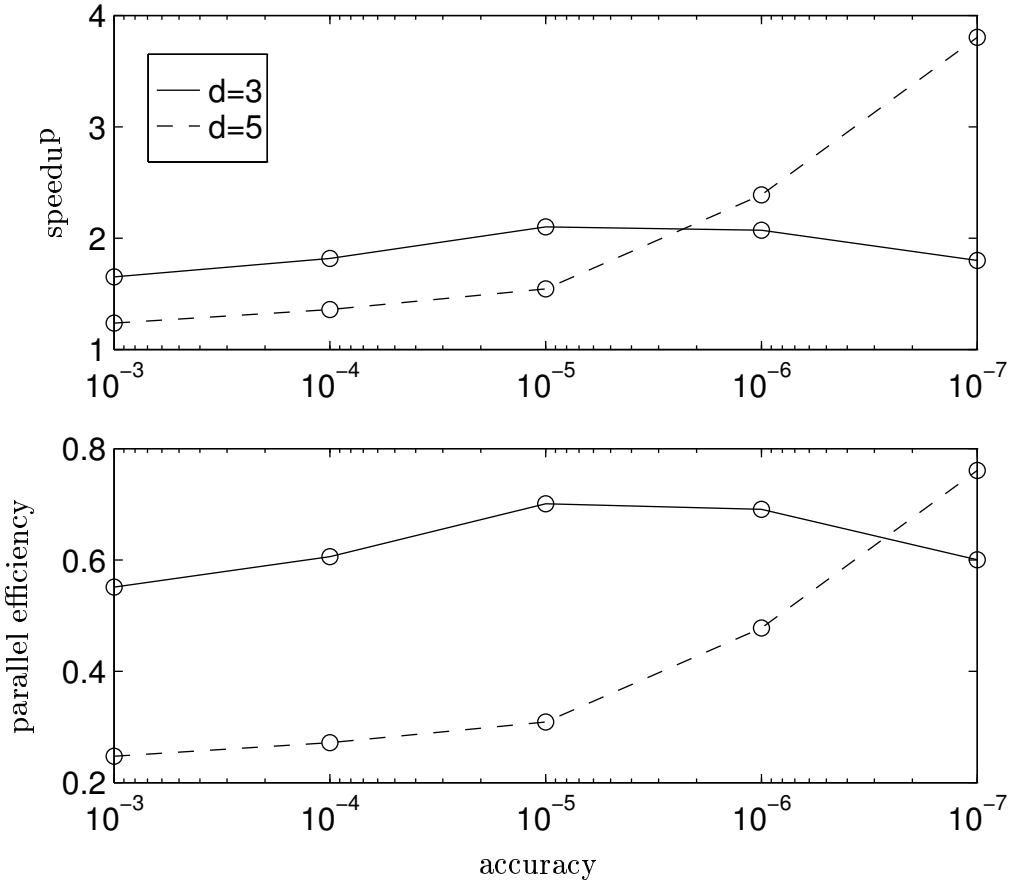


Figure 4.8: Speedup and efficiency for the spiral in Figure 4.8.

The resulting algorithm is more efficient than the undecomposed version as the number of unknowns increases, or as the adaptive accuracy requirements become more stringent. The method is easily parallelized and yields good parallel efficiencies in experiments with some elongated regions.

One issue we have not considered is the use of iterative solvers. In the face of increasingly large and dense systems, iterative solution should replace direct elimination. In the domain decomposition context, the use of an iterative method would dispose of the need to form the Schur complement, which also has a cost that is cubic in the number of uniformly distributed unknowns.

An open question is whether the domain decomposition can lead to more accurate results, in the sense of local relative accuracy. Eariler we observed that in crowded areas g' is small, and therefore we do not lose accuracy in a global sense even if we cannot resolve g' or θ' accurately. However, frequently the map $f = g^{-1}$ is also important, and without a locally accurate determination of g' we will be unable to compute f accurately. Consider the region of Figure 4.5. The solution on the right edge of the rectangle should have order of magnitude roughly $\exp(-16\pi) \approx 10^{-22}$. Since the solution on the left edge is $O(1)$ or $O(0.1)$, there is no hope of resolving the right edge in double precision with a single domain solution. (Note that the solution is still accurate relative to the entire boundary.) If an interface were placed halfway along the rectangle, however, where the solution is of an intermediate size, it is plausible that the correct order of magnitude could be reached. To exploit the situation appropriately, one probably needs to introduce scaling on the interfaces, because all interfaces are solved in a single system.

Chapter 5

A high-accuracy nonoverlapping method for the Laplace eigenvalue problem on polygons*

In this chapter we consider the eigenvalues and eigenfunctions of the Laplace operator in a planar region, subject to Dirichlet boundary conditions. This is a well-known problem with applications in acoustics [26], electromagnetics [18, 61], heat flow [18, 61], quantum mechanics [37], and queueing theory [57].

In 1966 Mark Kac [50] posed the question, "Can one hear the shape of a drum?" He was asking whether the Laplacian operator with Dirichlet boundary conditions could have identical spectra on two distinct planar regions. In 1992 Gordon, Webb, and Wolpert [32] answered the question negatively by an elegantly constructed counterexample, justifiably attracting a great deal of attention [15, 17, 68]. The simplest form of their example is a pair of regions bounded by eight-sided polygons,

*Portions reprinted with permission from T. A. Driscoll, "Eigenmodes of isospectral drums," *SIAM Review*, volume 39, number 1. Copyright 1997 by the Society for Industrial and Applied Mathematics. All rights reserved.

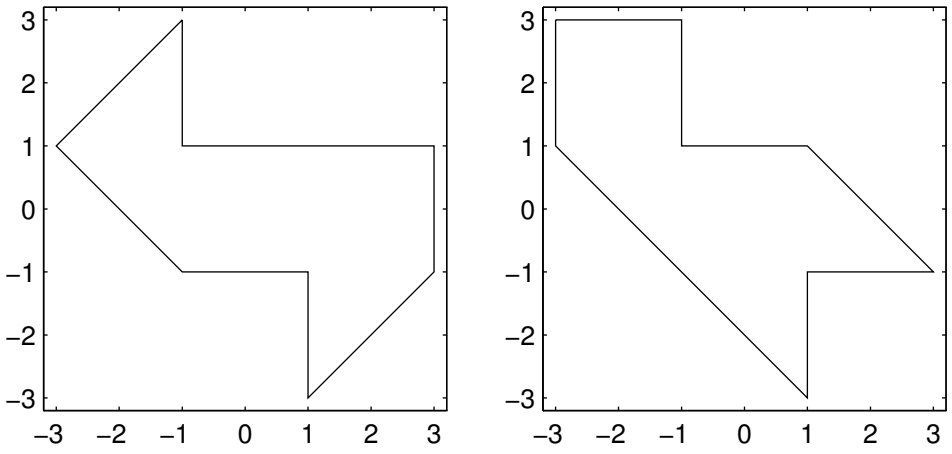


Figure 5.1: The isospectral drums discovered by Gordon, Webb, and Wolpert.

henceforth called the GWW isospectral drums; see Figure 5.1. Numerous similar examples have since been discovered [13].

There is a remarkably short and simple proof of the isospectrality of the drums [15, 17]. Yet there is no analytical method that can produce the eigenvalues themselves, or even accurate estimates. The best one can do is to use a form of Weyl's formula (see Section 5.3), which reveals the asymptotics of the spectrum. While experimentation can reveal fair estimates of the eigenvalues [75], a highly accurate determination for the exact mathematical problem calls for computation.

The most obvious numerical approach is the finite element method. There is a problem in the presence of reentrant corners, which degrade the convergence rate, but by using adaptive meshes or special functions at the corners, full convergence can be restored [78]. Even with careful treatment of the corners, though, the convergence obtained is limited by the quality of shape functions used in the approximation basis. Thus with classical piecewise linear elements, the error in the smallest eigenvalue decreases as the square of the mesh size. For modest accuracy in the eigenvalue, this is usually acceptable. But for accuracy comparable to double precision, the resulting

meshes themselves, to say nothing of the associated linear systems, become too large and unwieldy. One can turn to h-p methods [2] and spectral element methods [72], both of which accelerate convergence significantly at the cost of increased algorithm complexity and systems which are more expensive to set up and solve.

However, because the Laplace eigenmodes on polygons have certain structure, there are special numerical methods for its solution that are far superior to the general-purpose algorithms. The earliest and best-known of these is the method of particular solutions (MPS), first used on an L-shaped region by Fox, Henrici, and Moler [29].[1] While the MPS was a dramatic improvement over all previous methods for this region, it is unreliable for use with general polygons.

The MPS makes global use of information known about the solution near the corners. A localized use of this information, involving domain decomposition, was described by Descloux and Tolley [22]. We will describe an improvement that doubles the accuracy of this algorithm.

In this chapter we begin with the definition of the problem and an important feature of its solution, the Fourier–Bessel expansion. We then briefly summarize the MPS and describe its limitations. Next we describe the method of Descloux and Tolley and our improvement. Finally, we conclude with the results of high-precision computations of the eigenmodes of isospectral drums.

[1] Their calculations are the basis of the logo of The MathWorks, Inc., and can be demonstrated with the `membrane` command in MATLAB.

5.1 Fourier–Bessel expansion

Given a planar region Ω with polygonal boundary $\partial\Omega$, our goal is to find approximations to one or more eigenpairs $(\lambda, u) \in (\mathbf{R}, C(\overline{\Omega}))$ satisfying

$$\Delta u + \lambda u = 0 \qquad \text{in } \Omega, \tag{5.1a}$$

$$u = 0 \qquad \text{on } \partial\Omega. \tag{5.1b}$$

The nature of exact solutions to (5.1) has been extensively studied [18, 61]. All of the eigenvalues λ are positive, and they form a sequence that diverges to ∞. Furthermore, in the vicinity of a corner of the boundary $\partial\Omega$ with interior angle π/α, we have the *Fourier–Bessel expansion*

$$u(r, \varphi) = \sum_{n=1}^{\infty} c_n J_{n\alpha}(\sqrt{\lambda} r) \sin(n\alpha\varphi), \tag{5.2}$$

where (r, φ) are suitably oriented polar coordinates originating from the corner, and J_ν is a Bessel function of the first kind. This expression, which is essentially just a Fourier series, is valid within a radius at least as large as the distance to the nearest other corner of $\partial\Omega$. Note that when $\alpha < 1$, i.e., when the corner is reentrant, the solution just fails to be in the Sobolev space $H^{1+\alpha}(\Omega)$. This has important consequences for finite element methods, causing a decreased rate of convergence compared to the convex case, where the solution is in H^2 [78]. The convergence can be restored by the use of adaptive grid generation or the inclusion of special functions in the approximation basis.

The explicit knowledge of the Fourier–Bessel expansion (5.2) suggests its use in a numerical method. The simplest use would be as additional basis functions in a finite element calculation. However, in the next section we present two simpler alternatives.

5.2 Numerical algorithms

5.2.1 The method of particular solutions

One approach to exploiting the Fourier–Bessel expansion is the method of particular solutions (MPS), also known as point matching [29, 56]. In the MPS, a truncated form of (5.2) is taken about the reentrant corner(s). Eigenvalues are determined by requiring the trial solution to be zero at collocation points along the boundary $\partial\Omega$. In practice, this reduces to detecting the rank deficiency of a square or rectangular matrix $A(\lambda)$ whose columns are evaluations of the individual Fourier–Bessel terms at the collocation points.

The MPS works very well for the L-shaped region. A problem arises, however, when the MPS is applied to a region with more than one reentrant corner. Fourier–Bessel expansions must be made about each such corner, but the convergence of an expansion at another singularity can be very slow. In Figure 5.2 we show the result of applying the MPS to find the first eigenvalue of the left-hand region in Figure 5.1. We use N Fourier–Bessel terms at each reentrant corner, choose $2N$ collocation points evenly spaced on the boundary, and minimize the smallest singular value of the resulting $2N \times 2N$ matrix as a function of λ. The convergence to the eigenvalue as determined in Section 5.3.2 is slow and erratic as the number of expansion functions increases. No further expansion functions can be included, because the matrix is numerically singular for most values of λ and roundoff noise completely masks the smallest singular value.

5.2.2 A domain decomposition approach

In the MPS the essentially local eigenfunction expansion (5.2) is treated globally, with ill effects. We now describe a more local approach, first developed by De-

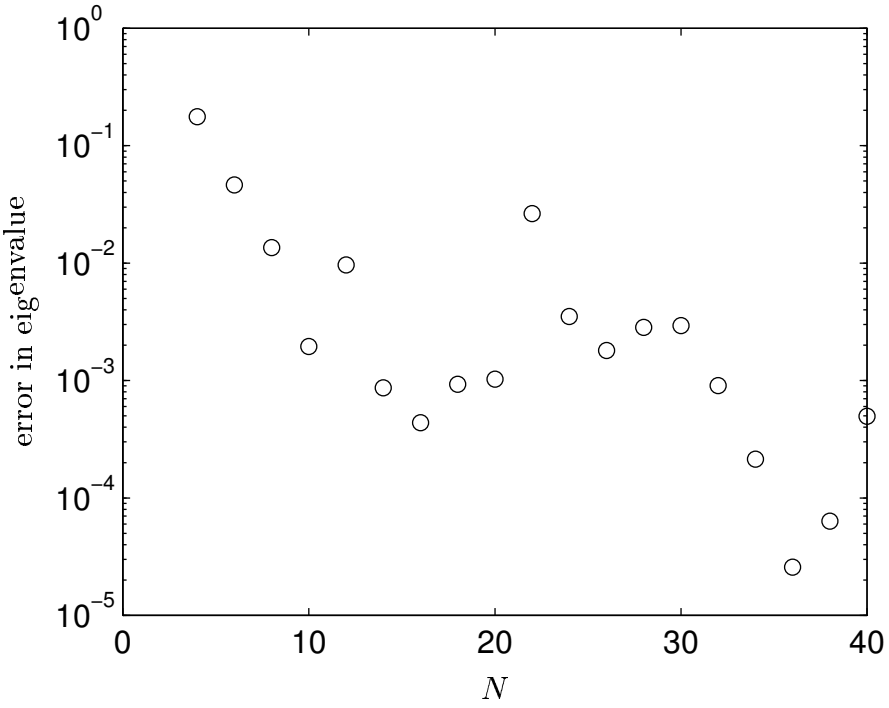


Figure 5.2: Convergence of the MPS to the first eigenvalue of the first GWW drum. There are N Fourier–Bessel terms from (5.2) at each reentrant corner and $2N$ evenly spaced collocation points. The convergence is slow because of the singularities at the reentrant corners.

scloux and Tolley [22]. Let the polygon Ω be subdivided into several nonoverlapping pieces Ω_j, $j = 1, \ldots, N$. We denote each interface $\partial\Omega_j \cap \partial\Omega_k$, which may be empty, by Γ_{jk}. Suppose that, given a scalar λ, we can find a set of N "subfunctions" $u_j \in C(\overline{\Omega_j})$ such that (λ, u_j) is a certain eigenpair on subregion Ω_j:

$$\Delta u_j + \lambda u_j = 0 \qquad \text{in } \Omega_j, \tag{5.3a}$$

$$u_j = 0 \qquad \text{on } \partial\Omega \cap \partial\Omega_j. \tag{5.3b}$$

(Note the difference between boundary conditions (5.1b) and (5.3b).) It is well known that λ is an eigenvalue for the whole region Ω if and only if along each nonempty interface Γ_{jk}, the subfunctions u_j and u_k and their normal derivatives match continuously.

Now let us further assume that the boundary of each Ω_j includes a portion of the entire boundary $\partial\Omega$ in such a way that exactly one vertex V_j with interior angle θ_j of the original polygon is in $\partial\Omega_j$. We describe a simple procedure for producing such a decomposition:

1. Compute a constrained Delaunay triangulation of Ω.

2. Choose an arbitrary point inside each triangle, and connect these points according to their adjacency in the dual graph of the triangulation (see Chapter 3).

3. For each triangle, connect the point chosen in step 2 to any point on each boundary edge which is an edge of the triangle.

See Figure 5.3 for an example. We do not formally prove that this procedure yields the desired decomposition, nor do we specify how to choose the points in steps 2 and 3. Centroids and midpoints may be acceptable, or one may want to numerically optimize the theoretical convergence rate (see (5.8)). A decomposition based on Voronoi diagrams [69] may also prove to be useful.

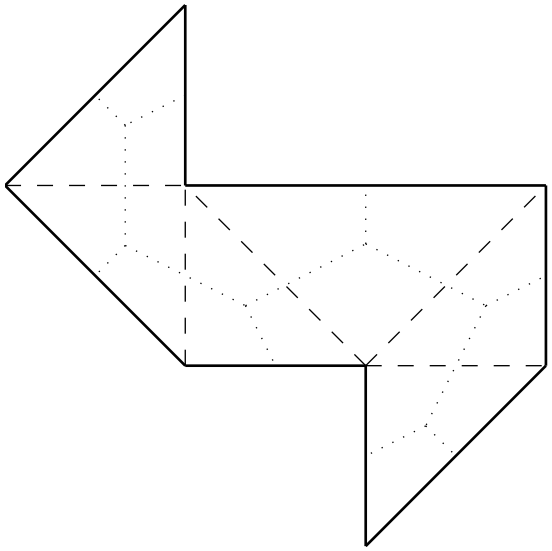


Figure 5.3: Domain decomposition by Delaunay triangulation.

We could enforce the matching conditions at collocation points on the interfaces, in the manner of the method of particular solutions. However, Descloux and Tolley [22] report the familiar problem with location of the singularity of the resulting matrix, and instead propose a method that employs finite element ideas within the domain decomposition framework. At the heart of the algorithm are the functionals

$$R(\lambda; u_1, \ldots, u_N) = \sum_{j<k} \int_{\Gamma_{jk}} \left[(u_j - u_k)^2 + |\nabla u_j - \nabla u_k|^2 \right] ds, \tag{5.4}$$

$$M(\lambda; u_1, \ldots, u_N) = \sum_{j=1}^{N} \iint_{\Omega_j} u_j^2 \, dx \, dy. \tag{5.5}$$

For fixed λ, let $\mu(\lambda)$ be the minimum of the quotient R/M over all choices of subfunctions. Now $\mu(\lambda) = 0$ if and only if λ is an eigenvalue of (5.1) and each u_j is the restriction of the eigenfunction u to Ω_j.[2]

We now use the finite element idea of replacing a minimization over infinite-

[2]Descloux and Tolley were able to prove convergence of their algorithm only when gradients, rather than normal derivatives, appear in (5.4).

dimensional function spaces by minimization over a nested family of finite-dimensional approximations. As bases for these spaces we choose terms of the local Fourier–Bessel expansions; that is, each subfunction u_j is expressed as a combination of n_j terms of the expansion (5.2) about V_j. This guarantees that the subproblem (5.3) is satisfied, even when λ is not an eigenvalue of Ω.[3] For optimal performance, n_j should be proportional to θ_j, the interior angle at V_j.

The corresponding approximation to $\mu(\lambda)$ now becomes the solution to a generalized matrix eigenproblem:

$$A(\lambda)v(\lambda) = \mu(\lambda)B(\lambda)v(\lambda). \tag{5.6}$$

The matrix A is computed by evaluating (5.4) by Gauss–Legendre quadrature with q nodes on each interface, and (5.5) is approximated by integrals over circular sectors so as to make the mass matrix B diagonal. This choice makes it convenient to replace the generalized eigensystem by the standard eigenproblem for $B^{-1/2}AB^{-1/2}$. Finally, a value of λ which minimizes our approximation to μ is taken as an estimate of an eigenvalue of (5.1).

Here is where we improve upon Descloux and Tolley's original algorithm. Suppose we can compute μ only to accuracy ϵ, which is on the order of machine precision. Because of the quadratic nature of μ near a minimum, a straightforward minimization gives an accuracy in λ of only order $\sqrt{\epsilon}$. If instead we seek solutions to $\dot{\mu}(\lambda) = 0$, the linearity of $\dot{\mu}$ near a minimum allows us to find λ to an accuracy comparable to that of μ. Figure 5.4 illustrates the situation for an estimate $\tilde{\lambda}$ of the first eigenvalue of the GWW drums. By differentiating (5.6) with respect to λ

[3]Actually, $\partial\Omega \cap \partial\Omega_j$ may contain isolated points where the boundary condition (5.3b) is not explicitly satisfied. However, the matching conditions do enforce this condition, and experiments show that slight changes in the Ω_j that avoid this problem do not substantially affect the performance of the algorithm.

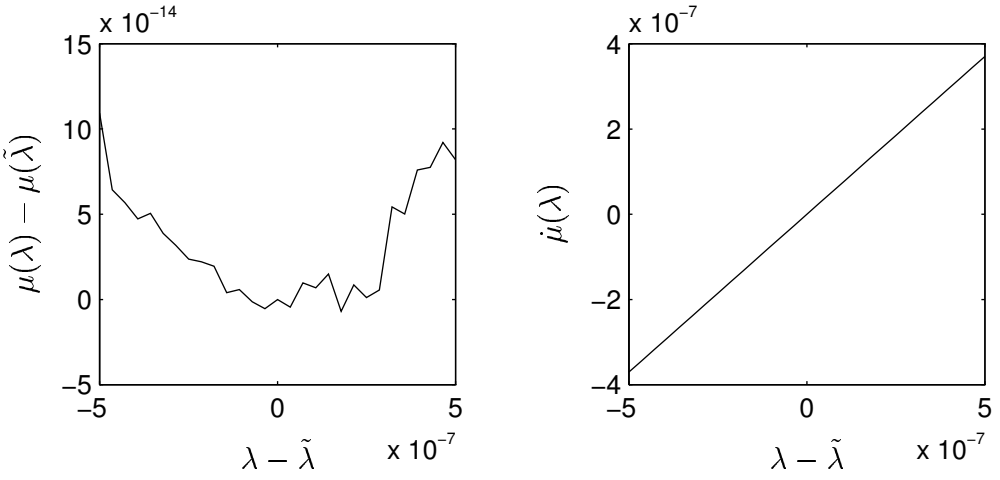


Figure 5.4: Comparison of $\mu(\lambda)$ with $\dot{\mu}(\lambda)$ near a minimum $\tilde{\lambda}$, which is an estimate of the first GWW eigenvalue. Rounding errors produce visible noise in the first curve; a method based on that curve can be accurate only to the square root of the roundoff error.

and left-multiplying through by v^T, we see that

$$\dot{\mu}(\lambda) = \frac{v^T(\dot{A} - \mu\dot{B})v}{v^T B v}. \tag{5.7}$$

The matrices $\dot{A}$ and $\dot{B}$ can be computed in a straightforward manner. A simple calculation shows that combining $\dot{\mu}(\lambda) = 0$ with (5.7) and (5.6) is equivalent to the Euler–Lagrange conditions for minimizing R, subject to $M = 1$, over λ and the coefficients of the truncated expansions.

To summarize, the algorithm can be viewed as an iteration in the parameter λ whose convergence is dictated by domain decomposition considerations. Each step of the iteration is computed approximately by a large singular finite element method, where the basis functions depend nonlinearly on the parameter λ. Improved accuracy is achieved by increasing the number of basis functions in the inner step.

Descloux and Tolley [22] were able to prove geometric convergence of this algorithm. Suppose that n_j, the number of expansion functions in subregion j, is equal

to $m\nu_j$, for given fixed ν_j and integer m. A key parameter is defined by

$$\omega = \max_j \left[\left(\frac{\max_{z \in \Omega_j} |z - V_j|}{\min_{k \neq j} |V_k - V_j|} \right)^{\nu_j \alpha_j} \right]. \tag{5.8}$$

When $\omega < 1$, the error in an eigenvalue estimate is no larger than a constant times $(\omega + \epsilon)^m$, for any $\epsilon > 0$. We have observed that ω is a pessimistic bound for the convergence rate in many cases.

If $\omega > 1$, convergence is not guaranteed, and we have seen apparent failures to converge in some cases. It is possible to change ω by modifying the decomposition used. If desired, extra edge vertices (with $\alpha = 1$) or interior elements (with cosine terms as well as sine terms in the Fourier–Bessel expansion) may be added. We do not have a practical algorithm that always produces a decomposition having $\omega < 1$.

5.3 Eigenmodes of isospectral drums

5.3.1 Background

We now return to the isospectral drums of Figure 5.1 discovered by Gordon, Webb, and Wolpert.

The simplest and most versatile proof of the isospectrality employs "transplantation" [5] of the eigenfunctions. The regions are shown to be (or are constructed to be) made up of nonoverlapping translations, rotations, and reflections of a single shape, such as a triangle. Given an eigenfunction on one region, one can prescribe a function over the other region whose values over each piece are linear combinations of the eigenfunction values over several of the pieces of the first region. The combinations are chosen to satisfy the boundary conditions and to match values and derivatives at interfaces between pieces, and the interior equation is satisfied by superposition. Hence the result is an eigenfunction of the second region having

the same eigenvalue. To complete the proof of isospectrality, one need only check that the procedure is invertible. Note that the proof is nonconstructive and other information, particularly the actual values of the eigenvalues, remains unknown.

The first successful determination of the spectra of the GWW drums was by Sridhar and Kudrolli [75], who used an experimental approach. They constructed microwave cavities in the shapes of the polygons and measured resonances in transverse magnetic waves, which obey the Helmholtz equation. In this manner they obtained the first 54 eigenvalues to within about 0.3%. The accuracy and versatility of this method are limited primarily by the fabrication of the cavities.

Wu, Sprung, and Martorell [90] describe a numerical method which they call mode matching, also rooted in domain decomposition. In this method, the expansion (5.2) is not used. Instead, analytic expressions of the solutions to (5.3) must be known throughout each subdomain Ω_j. For the GWW drums, it is possible to accomplish this by dividing each drum into five pieces, each of which is a square or a $(45°, 45°, 90°)$ triangle. Because of the simple shapes, the eigenfunctions are differences between products of sine functions. A subfunction u_j is expanded as a combination of these functions with unknown coefficients, the expansions are truncated, and the requirement of functions and derivatives matching at interfaces becomes a linear system in these coefficients. As with the method of particular solutions, an eigenvalue is a value of λ for which the matrix of this system becomes singular. In this case, however, Wu *et al.* report no difficulty with numerical near-singularity. We shall show that the figures computed by Wu *et al.* are accurate to about four digits.

A significant shortcoming of the mode matching method is that it is not universally applicable. In general we cannot expect Ω to admit a simple decomposition for which the eigenfunctions of the individual pieces can be explicitly written in a convenient and usable form. By contrast, the expansion (5.2) is available for any

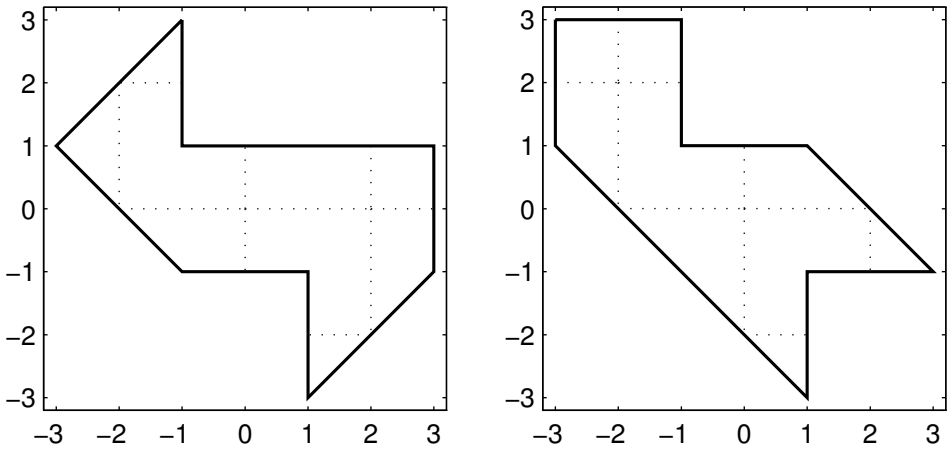


Figure 5.5: Subdivisions of the GWW drums used for the domain decomposition method.

polygonal domain.

5.3.2 Results

In Figure 5.5 we illustrate domain decompositions used for the GWW isospectral polygons. In Table 5.1 we list our estimates of the first 25 eigenvalues of the GWW isospectral drums. For these calculations we used $n_j = 36/\alpha_j = 36\theta_j/\pi$ basis functions in region Ω_j and $q = 40$ Gauss quadrature points on each interfacial line segment. The results for the two drums agree with each other nearly to machine precision.

In Figure 5.6 we compare $N(\lambda)$, the number of eigenvalues (counting multiplicity) less than λ, to the corrected Weyl's formula, which for a polygon is [3]

$$N(\lambda) \approx \frac{A}{4\pi}\lambda - \frac{P}{4\pi}\sqrt{\lambda} + \sum_{j=1}^{N} \frac{1}{24}\left(\alpha_j - \alpha_j^{-1}\right), \tag{5.9}$$

where A is the area of the region Ω and P is its perimeter. (By isospectrality, A and P are necessarily the same for the two drums.) The formula agrees excellently

Table 5.1: The first twenty-five eigenvalues of the GWW isospectral drums. All digits shown are believed to be correct.

2.53794399980	9.20929499840	14.3138624643	20.8823950433	24.6740110027
3.65550971352	10.5969856913	15.8713026200	21.2480051774	26.0802400997
5.17555935622	11.5413953956	16.9417516880	22.2328517930	27.3040189211
6.53755744376	12.3370055014	17.6651184368	23.7112974848	28.1751285815
7.24807786256	13.0536540557	18.9810673877	24.4792340693	29.5697729132

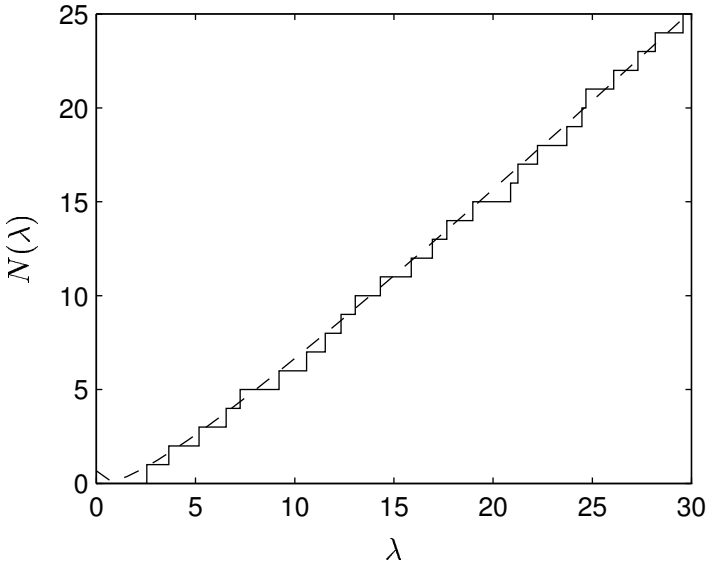


Figure 5.6: Integrated eigenvalue density for the GWW drums. The solid stairstep is the actual density, compared with the dashed line representing Weyl's formula in (5.9).

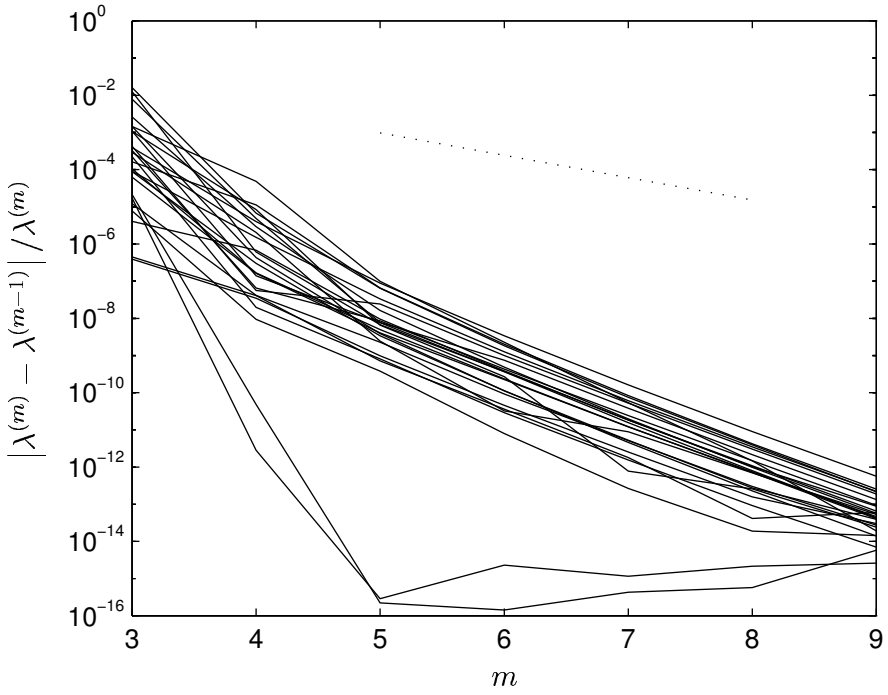


Figure 5.7: Convergence of the eigenvalue estimates. The dotted line shows ω^m, with ω given by (5.8). See text for comments about the two dramatically decreasing curves.

with the exact $N(\lambda)$.

We believe that the entries of Table 5.1 are accurate to all digits shown. As support for this claim, in Figure 5.7 we present the convergence history of the estimates with respect to $n_j = m\nu_j = 4m/\alpha_j$. For each value of m, we find that the estimates of any eigenvalue for the two drums agree essentially to machine precision, and we use $\lambda^{(m)}$ to denote this common number. Figure 5.7 shows the relative change in the successive estimates $\lambda^{(m)}$ as m varies. The convergence bound defined by (5.8) is $\omega = 1/4$ for both regions. However, the observed convergence is much better, at roughly twice that rate. In general, curves which are higher on the graph correspond to higher eigenvalues, the two "superconvergent" curves being

exceptions which are discussed below. Note that all the convergence curves end at less than 10^{-12}.

As mentioned previously, a feature of the results is the dramatic agreement for all the estimates of the two drums, regardless of their accuracy. In fact, all the eigenvalues of the generalized system (5.6), and hence $\mu(\lambda)$, are numerically identical for the two regions for any m. Presumably this occurs because the subdivisions of Figure 5.5 respect the transplantation symmetries between the regions. As a further check on our results, we applied the algorithm using the less regular subdivisions depicted in Figure 5.8. The estimates for the two regions now differ by amounts consistent with their apparent accuracy. In Figure 5.9 we compare the difference between corresponding values at each m to the rate ω^m, where now $\omega \approx 0.52$. The estimates for $m = 11$ all agree with the numbers of Table 5.1 to the full 12 digits.

Figures 5.10–5.11 show in detail the first eight eigenfunctions of the GWW drums, including the nodal lines (cf. Figure 2 of Sridhar and Kudrolli [75]). Figure 5.12 shows contours for the ninth mode, which is clearly derived from the first mode on a $(45°, 45°, 90°)$ triangle. This triangle is the fundamental shape that forms the basis of the transplantation proof. The exact eigenvalue in this case is $5\pi^2/4$, which agrees with our computed values to 15 digits. A similar phenomenon occurs at the twenty-first mode, which is equivalent to the second mode on the triangle with eigenvalue $10\pi^2/4$. In fact, these two modes account for the "superconvergent" curves of Figures 5.7 and 5.9. We hypothesize that the accelerated convergence occurs because the symmetries of these eigenfunctions about the corners cause many of the Fourier coefficients in (5.2) to be exactly zero.

In Figure 5.13 we compare our results to other published determinations of the first 25 eigenvalues. Based on their microwave experiments, Sridhar and Kudrolli report these eigenvalues to an rms relative accuracy of about 0.3%. We observe that the error in their estimates is frequently much larger than the agreement between

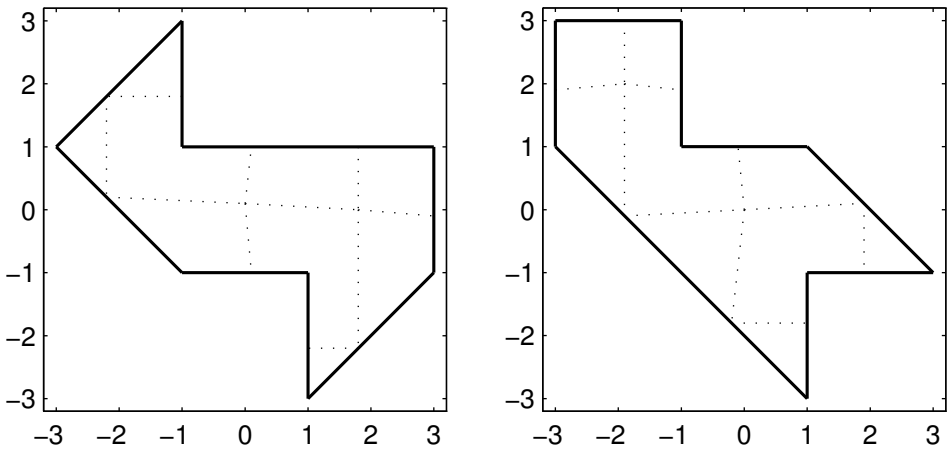


Figure 5.8: An alternate subdivision of the GWW drums.

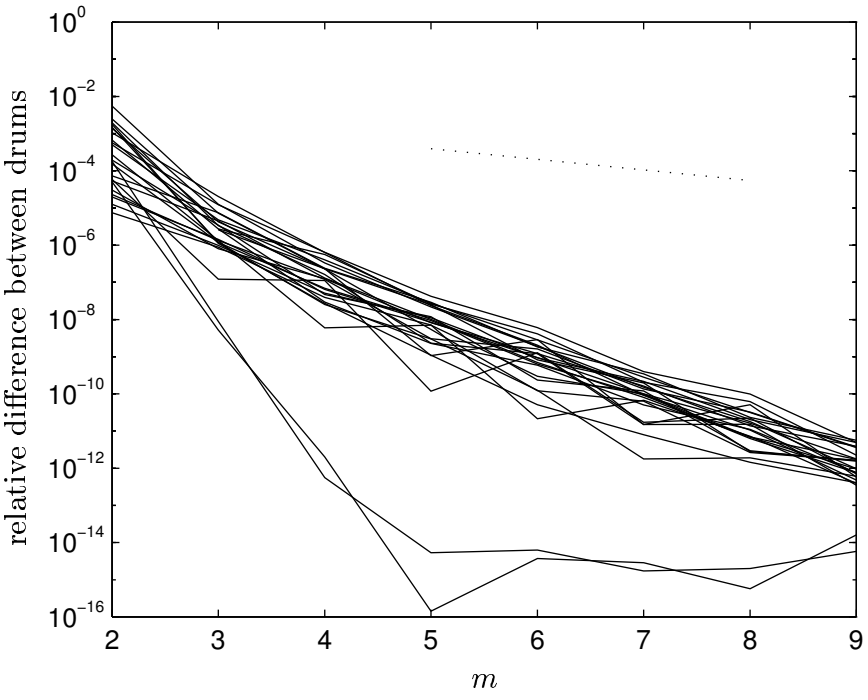


Figure 5.9: Difference in estimates for the two drums at each m when using the subdivisions of Figure 5.8.

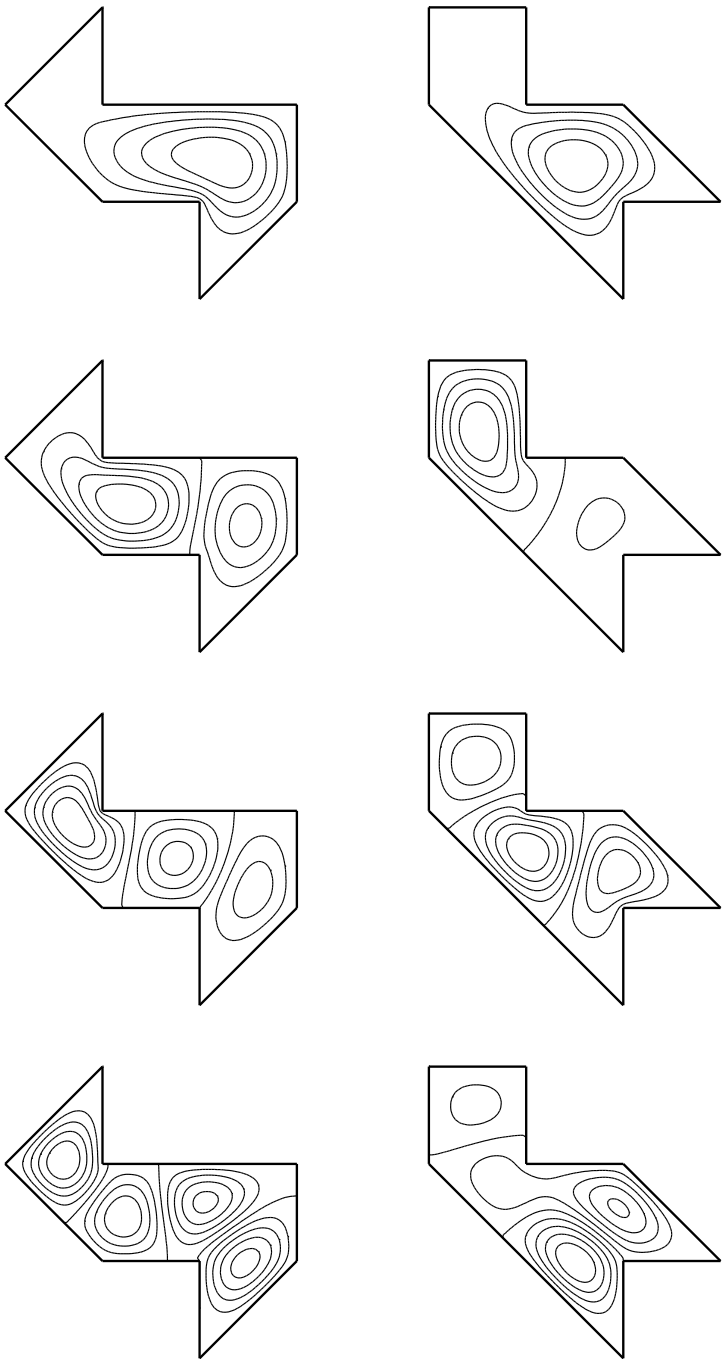


Figure 5.10: First four eigenfunctions of the GWW isospectral drums. Each is normalized to have unit amplitude. The contours are at levels $-0.8, -0.6, \ldots, 0.8$.

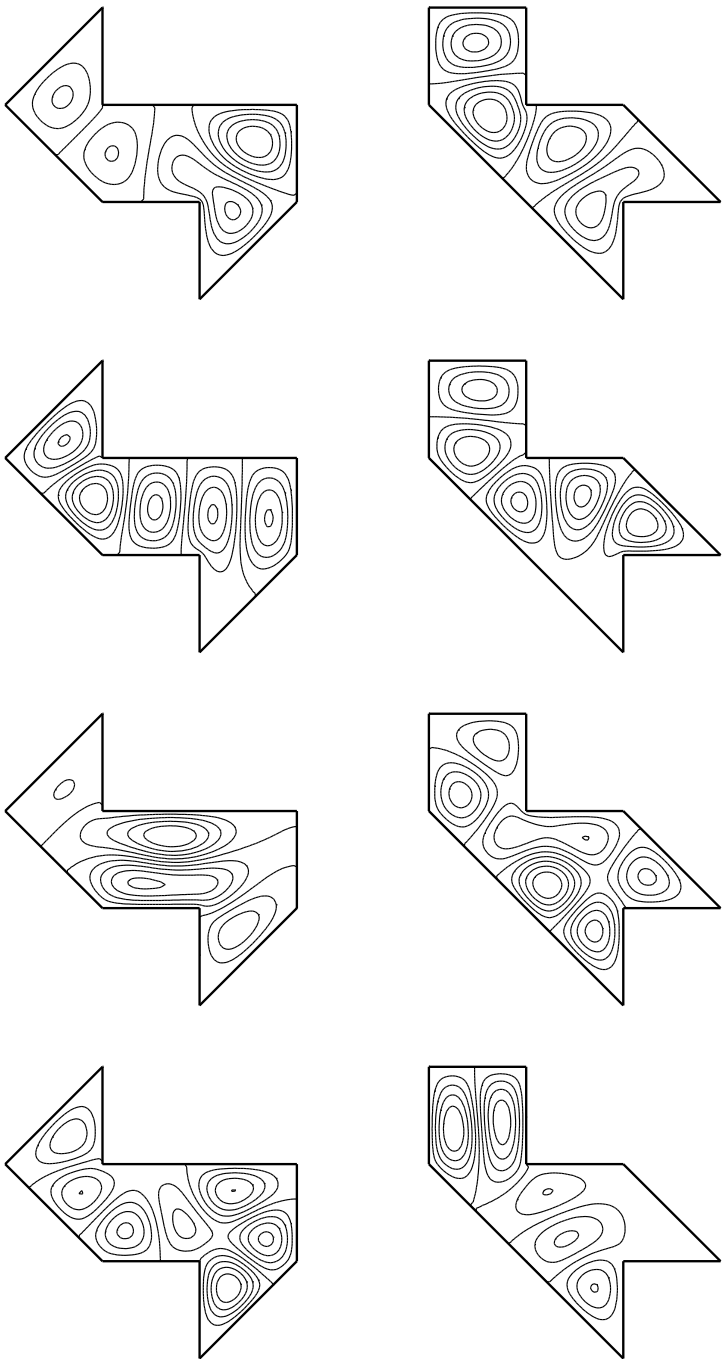


Figure 5.11: Eigenfunctions 5–8 of the GWW isospectral drums.

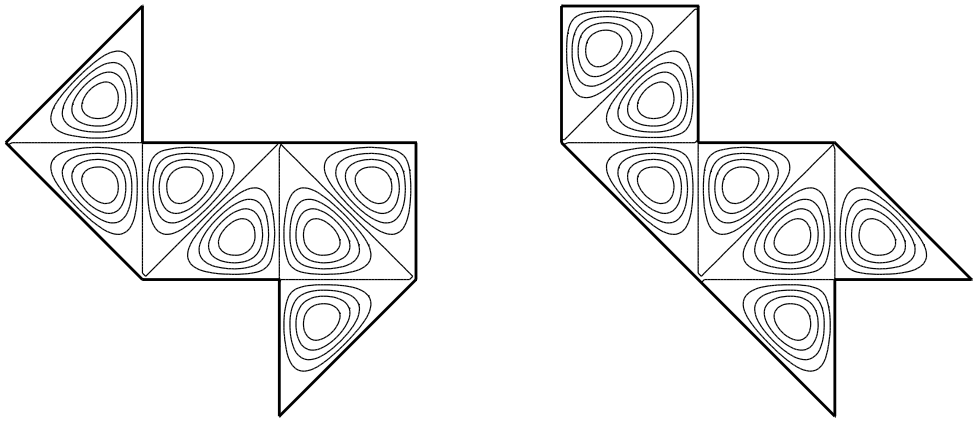


Figure 5.12: The ninth mode of the GWW drums. This corresponds to the first mode on a $(45°, 45°, 90°)$ triangle.

their values, but there is no clear explanation of this phenomenon.[4] The eigenvalues obtained by Wu, Sprung, and Martorell [90] by extrapolation of results from finite differences and mode matching agree with our results to about 3 and 4 digits, respectively.[5]

A direct numerical approach to this problem is to use a finite element software package. We first chose PLTMG [4] because of its widespread availability and automatic adaptive mesh refinement capabilities. PLTMG regards the linear problem (5.1) as a nonlinear continuation problem with parameter λ and functional $\rho(u) = ||u||_{L^2}$. The procedure, which is outlined in Section 4.6.2 of [4], is to track the zero solution for varying λ until a bifurcation point in the λ-ρ plane is found, at which point the bifurcating branch with constant λ (the eigenfunction) is followed. The grid is adaptively improved and the estimate for λ updated until the desired

[4]Kudrolli has suggested that the finite skin depth of the copper used for the cavities may be responsible [55]. However, that does not explain the fairly sudden transition in behavior seen starting at the 14th eigenvalue.

[5]In the mode matching method, the degenerate modes 9 and 21 are not computed but are taken exact.

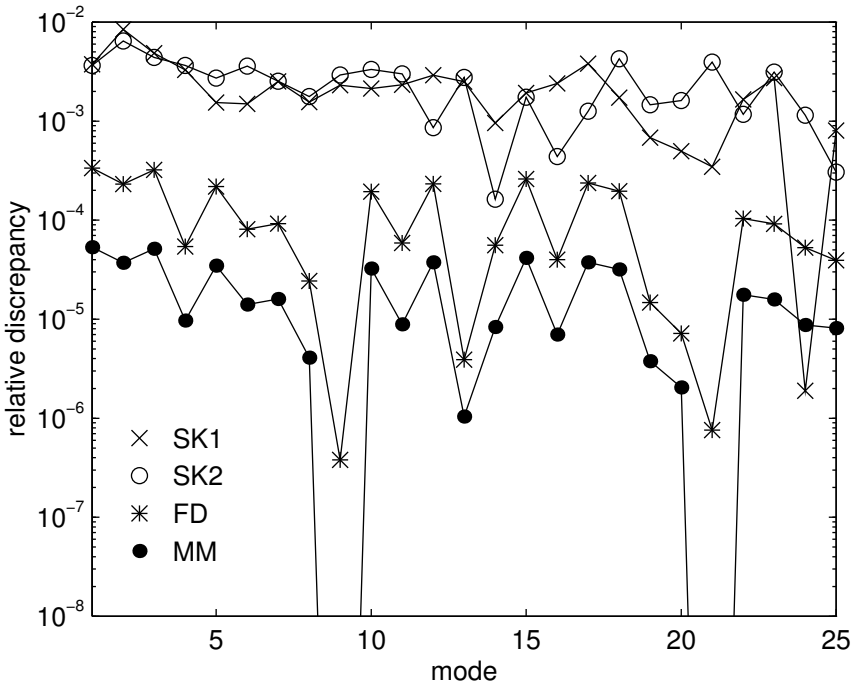


Figure 5.13: Comparison with other determinations of the spectra. Shown are the two sets obtained by Sridhar and Kudrolli by microwave experiments, and the results of finite differences and mode matching reported by Wu *et al.*

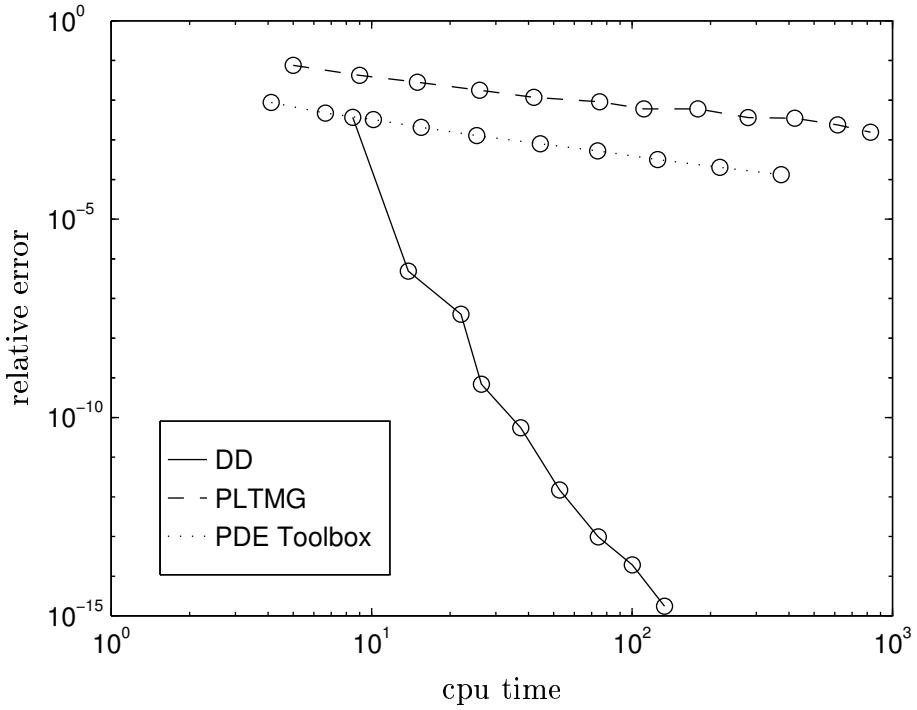


Figure 5.14: Comparison of PLTMG and the PDE Toolbox with the domain decomposition method. The data are based on the computation of the first eigenvalue of the first GWW drum on a Sun SPARCstation 10.

accuracy is apparently achieved.

Because the use of the nonlinear continuation method is atypical for the linear eigenvalue problem, we have additionally applied to this problem the PDE Toolbox for MATLAB, which also uses piecewise linear finite elements. Here the eigenvalue estimates come from the solution of a generalized matrix eigenproblem in the usual way. As with PLTMG, we adaptively refine the mesh based on *a posteriori* error estimates of the most recent solution.

In Figure 5.14 we compare the efficiency of the domain decomposition method to computation of the eigenvalues in PLTMG and the PDE Toolbox. Figure 5.15 shows the adaptive grid process for the first eigenvalue on the first drum. The

finest structure occurs near the corner where the eigenfunction is large. At this stage, there are 966 triangles and 402 vertices, and the eigenvalue estimate is about 2.5659. If one were to try to obtain 12 digits via either of the finite element methods, storage as well as computational time would become a serious obstacle.

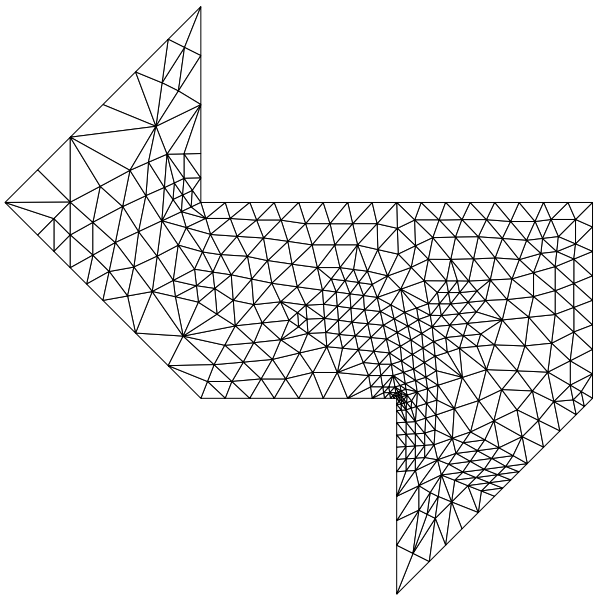


Figure 5.15: Adaptive mesh refinement by PLTMG. The mesh is most refined near one of the reentrant corners.

One advantage of the domain decomposition method over physical experiment and mode matching is its flexibility in application to other polygons. We have applied the domain decomposition method to another pair of isospectral drums, depicted in Figure 5.16. These regions were constructed using the techniques of Buser *et al.* [13]. The fundamental unit of the construction is a $(30°, 70°, 80°)$ triangle, which renders the mode matching method impractical. The first ten eigenvalues of these regions are listed in Table 5.2 to nine digits. In Figure 5.17 we present the convergence history analogous to Figure 5.7. Here, the value of ω is about 0.48;

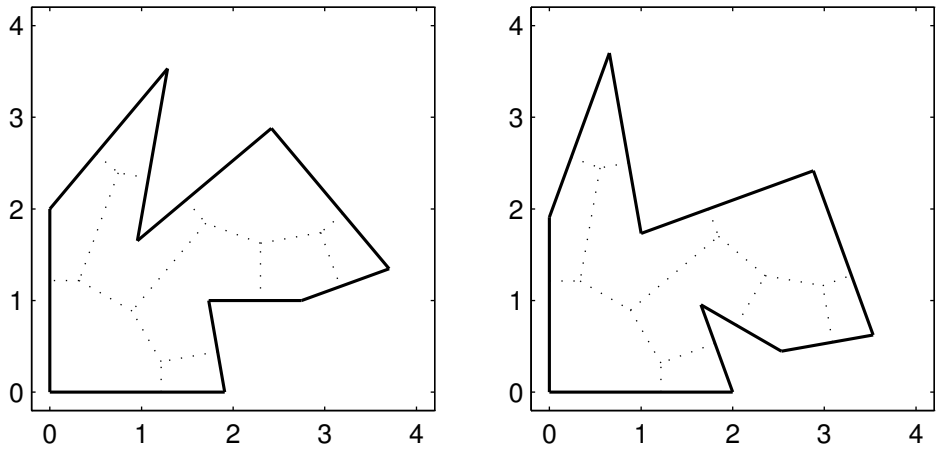


Figure 5.16: Two more isospectral regions, and the subdivisions used in the domain decomposition method.

Table 5.2: First ten eigenvalues of the isospectral drums of Figure 5.16.

5.63126379	18.8537757
7.18148848	19.8509471
12.7905748	24.1803291
13.0935554	27.5379471
17.0680091	30.0098327

however, the actual convergence is again much better. Selected eigenfunctions for these drums are displayed in Figure 5.18.

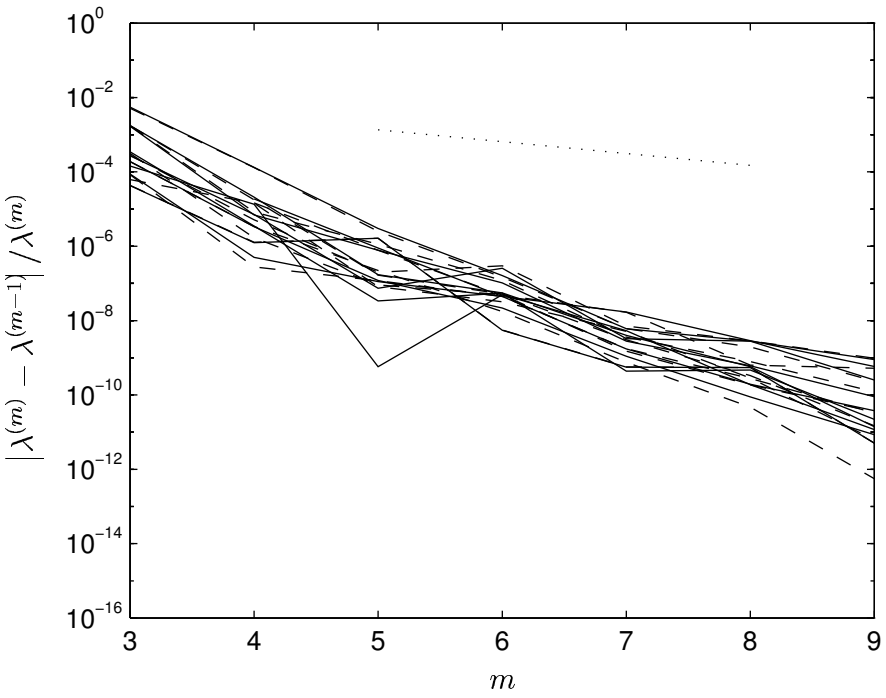


Figure 5.17: Convergence of the eigenvalue estimates for the second pair of drums. Solid lines are for the first drum, dashed lines are for the second drum, and the dotted line is a multiple of ω^m, where $\omega \approx 0.48$.

5.4 Summary

The Laplace eigenvalue problem for polygons is important in applications and plays a role in the celebrated discovery of isospectral drums. Standard algorithms to compute the eigenmodes converge fairly slowly when compared to special-purpose methods that exploit the structure of the solutions.

One such method, a modification of that described by Descloux and Tolley [22],

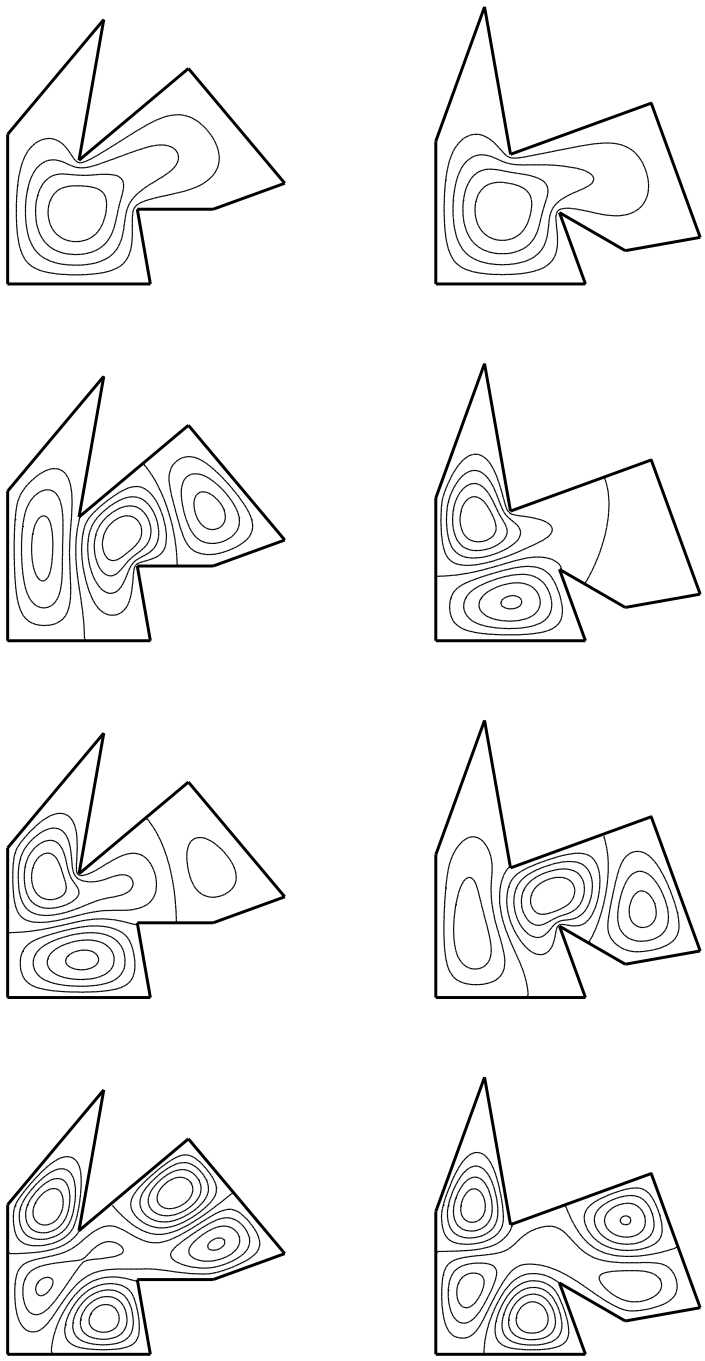


Figure 5.18: Eigenmodes 1, 3, 4, and 6 of the second pair of isospectral drums. Modes have unit amplitude, and contours are drawn at $-0.8, -0.6, \ldots, 0.8$.

is described in this chapter. It is modeled after domain decomposition but, unlike other domain decomposition algorithms for elliptic eigenvalue problems [11, 25, 58], uses the Fourier–Bessel expansion particular to the Laplace problem. The resulting algorithm is efficient, accurate, and useful for arbitrary polygonal regions.

Using this algorithm, we have made the first high-precision determinations of the eigenvalues of the GWW isospectral drums. We have also demonstrated that the method is flexible enough to be applied to other instances of polygonal isospectral regions.

To see color depictions of the eigenfunctions of the GWW drums and some animations of vibrations based on selected combinations of the modes [23], use a Web browser such as Netscape to open the URL `http://cam.cornell.edu/~driscoll/`.

Bibliography

[1] K. Amano. A charge simulation method for the numerical conformal mapping of interior, exterior, and doubly-connected domains. *J. Comp. Appl. Math.*, 53:353–370, 1994.

[2] I. Babuška and M. Suri. The P and H-P versions of the finite element method, basic principles and properties. *SIAM Review*, 36:578–632, 1994.

[3] H. P. Baltes and E. R. Hilf. *Spectra of Finite Systems.* Bibliographisches Institut, Mannheim, 1976.

[4] R. Bank. *PLTMG Users' Guide 7.0: A Software Package for Solving Elliptic Partial Differential Equations.* SIAM, 1994.

[5] P. Bérard. Transplantation et isospectralité. *Math. Ann.*, 292:547–559, 1992.

[6] M. Bern and D. Eppstein. Mesh generation and optimal triangulation. In D. Z. Du and F. Hwang, editors, *Computing in Euclidean Geometry.* World Scientific, 1992. Also Xerox Palo Alto Research Center Technical report CSL-92-1.

[7] M. Bern, D. Eppstein, and J. Gilbert. Provably good mesh generation. In *Proc. 31st IEEE Symp. Foundations of Computer Science*, pages 231–241, 1990.

[8] J.-P. Berrut. Numerische Lösung der Symmschen Integralgleichung durch Fourier-Methoden. Master's thesis, ETH Zürich, 1976.

[9] J.-P. Berrut. A Fredholm integral equation of the second kind for conformal mapping. *J. Comp. Appl. Math.*, 14:99–110, 1985.

[10] P. Bjørstad and E. Grosse. Conformal mapping of circular-arc polygons. *SIAM J. Sci. Stat. Comput.*, 8:19–32, 1987.

[11] F. Bourquin and F. d'Hennezel. Numerical study of an intrinsic component mode synthesis method. *Comp. Meth. Appl. Mech. Eng.*, 97:49–76, 1992.

[12] C. A. Brebbia, J. C. F. Telles, and L. C. Wrobel. *Boundary Element Techniques: Theory and Applications in Engineering.* Springer, 1984.

[13] P. Buser, J. Conway, P. Doyle, and K. Semmler. Some planar isospectral domains. *Int. Math. Res. Not.*, to appear.

[14] Tony F. Chan and Tarek P. Mathew. Domain decomposition algorithms. *Acta Numerica*, pages 61–143, 1994.

[15] S. J. Chapman. Drums that sound the same. *Amer. Math. Mon.*, 102:124–138, 1995.

[16] L. P. Chew. Guaranteed-quality triangular meshes. Technical Report 89–983, Department of Computer Science, Cornell University, 1989.

[17] B. Cipra. You can't hear the shape of a drum. *Science*, 255:1642–1643, 1992.

[18] R. Courant and D. Hilbert. *Methods of Mathematical Physics*, volume 2. Interscience, 1953.

[19] H. D. Däppen. *Die Schwarz–Christoffel-Abbildung für zweifach zusammenhängende Gebiete mit Anwendungen.* PhD thesis, ETH Zürich, 1988.

[20] R. T. Davis. Numerical methods for coordinate generation based on Schwarz–Christoffel transformations. In *4th AIAA Comput. Fluid Dynamics Conf.*, pages 1–15, Williamsburg, VA, 1979.

[21] J. E. Dennis and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Prentice-Hall, 1983.

[22] J. Descloux and M. Tolley. An accurate algorithm for computing the eigenvalues of a polygonal membrane. *Comp. Meth. App. Mech. Eng.*, 39:37–53, 1983.

[23] T. A. Driscoll and B. Land. Vibrations of isospectral drums. Computer animation, 1995.

[24] T. A. Driscoll and S. A. Vavasis. Numerical conformal mapping using cross-ratios and Delaunay triangulation. Submitted to SIAM J. Sci. Comp.

[25] C. Farhat and M. Geradin. On a component mode synthesis method and its application to incompatible substructures. *Comp. Struct.*, 51:459–473, 1994.

[26] N. H. Fletcher and T. D. Rossing. *The Physics of Musical Instruments.* Springer-Verlag, 1991.

[27] J. M. Floryan. Conformal-mapping-based coordinate generation method for flows in periodic configurations. *J. Comp. Phys.*, 62:221–247, 1986.

[28] J. M. Floryan and C. Zemach. Schwarz–Christoffel mappings: A general approach. *J. Comp. Phys.*, 72:347–371, 1987.

[29] L. Fox, P. Henrici, and C. Moler. Approximations and bounds for eigenvalues of elliptic operators. *SIAM J. Numer. Anal.*, 4:89–102, 1967.

[30] D. Gaier. *Konstruktive Methoden der konformen Abbildung.* Springer-Verlag, 1964.

[31] D. Gaier. Integralgleichungen erster Art und konforme Abbildung. *Math. Z.*, 147:113–129, 1976.

[32] C. Gordon, D. Webb, and S. Wolpert. Isospectral plane domains and surfaces via Riemannian orbifolds. *Invent. Math.*, 110:1–22, 1992.

[33] I. S. Gradshteyn and I. M. Ryzhik. *Table of Integrals, Series, and Products.* Academic Press, 5th edition, 1994.

[34] M. H. Gutknecht. An iterative method for solving linear equations based on minimum norm Pick-Nevanlinna interpolation. *Approx. Theory V*, pages 371–374, 1986.

[35] M. H. Gutknecht. Numerical conformal mapping methods based on function conjugation. *J. Comp. Appl. Math.*, 14:31–77, 1986.

[36] J. K. Hayes, D. K. Kahaner, and R. G. Kellner. An improved method for numerical conformal mapping. *Math. Comp.*, 26:327–334, 1972.

[37] W. D. Heiss, editor. *Chaos and Quantum Chaos*, Blydepoort, South Africa, 1992. Eighth Chris Engelbrecht Summer School on Theoretical Physics, Springer.

[38] P. Henrici. *Applied and Computational Complex Analysis*, volume 1. Wiley, 1974.

[39] P. Henrici. *Applied and Computational Complex Analysis*, volume 3. Wiley, 1986.

[40] M. Hoekstra. Coordinate generation in symmetrical interior, exterior, or annular 2D domains, using a generalized Schwarz–Christoffel transformation. In J. Hauser and C. Taylor, editors, *Numerical Grid Generation in Computational Fluid Mechanics.* Pineridge Press, 1986.

[41] D. M. Hough. Conformal mapping and Fourier–Jacobi approximations. ETH Zürich IPS Research Report 89-06, 1989.

[42] D. M. Hough. User's guide to CONFPACK. ETH Zürich IPS Research Report 90-11, 1990.

[43] D. M. Hough and N. Papamichael. An integral equation method for the numerical conformal mapping of interior, exterior, and doubly-connected domains. *Numer. Math.*, 41:287–307, 1983.

[44] L. H. Howell. *Computation of Conformal Maps by Modified Schwarz–Christoffel Transformations*. PhD thesis, MIT, 1990.

[45] L. H. Howell. Numerical conformal mapping of circular arc polygons. *J. Comput. Appl. Math.*, 46:7–28, 1993.

[46] L. H. Howell. Schwarz–Christoffel methods for multiply-elongated regions. In *Proc. of the 14th IMACS World Congress on Computation and Applied Mathematics*, 1994.

[47] L. H. Howell and L. N. Trefethen. A modified Schwarz–Christoffel transformation for elongated regions. *SIAM J. Sci. Stat. Comput.*, 11:928–949, 1990.

[48] C. Hu. User's guide to DSCPACK. Nat. Inst. Aviation Res. 95-1, Wichita State Univ., 1995.

[49] M. A. Jaswon and G. T. Symm. *Integral equation methods in potential theory and electrostatics*. Academic Press, 1977.

[50] M. Kac. Can one hear the shape of a drum? *Amer. Math. Monthly*, 73 part II:1–23, 1966.

[51] J. H. Kane. *Boundary Element Analysis in Engineering Continuum Mechanics*. Prentice Hall, 1994.

[52] N. Kerzman and M. R. Trummer. Numerical conformal mapping via the Szegő kernel. *J. Comp. Appl. Math.*, 14:125–142, 1986.

[53] D. E. Keyes. Domain decomposition: A bridge between nature and parallel computers. Technical Report 92-44, ICASE, 1992.

[54] B. C. Krikeles and R. L. Rubin. On the crowding of parameters associated with Schwarz–Christoffel transformations. *Appl. Math. Comp.*, 28:297–308, 1988.

[55] A. Kudrolli. Private communication, Feb. 1995.

[56] J. R. Kuttler and V. G. Sigillito. Eigenvalues of the Laplacian in two dimensions. *SIAM Review*, 26:163–193, 1984.

[57] G. Louchard, R. Schott, M. Tolley, and P. Zimmerman. Random walks, heat equation and distributed algorithms. *J. Comp. Appl. Math*, 53:243–274, 1994.

[58] J. C. Luo. A domain decomposition method for eigenvalue problems. In D. E. Keyes, T. F. Chan, G. A. Meurant, J. S. Scroggs, and R. G. Voigt, editors, *Proceedings of the Fifth International Conference on Domain Decomposition Methods*, pages 306–321. SIAM, 1992.

[59] R. Menikoff and C. Zemach. Methods for numerical conformal mapping. *J. Comp. Phys.*, 36:366–410, 1980.

[60] S. A. Mitchell and S. A. Vavasis. Quality mesh generation in three dimensions. In *Proc. 8th ACM Symp. on Computational Geometry*, pages 212–221, 1992.

[61] P. M. Morse and H. Feshbach. *Methods of Theoretical Physics.* McGraw–Hill, 1953.

[62] Z. Nehari. *Conformal Mapping.* Dover, 1952.

[63] S. T. O'Donnell and V. Rokhlin. A fast algorithm for the numerical evaluation of conformal mappings. *SIAM J. Sci. Stat. Comput.*, 10:475–487, 1989.

[64] N. Papamichael and N. S. Stylianopoulos. A domain decomposition method for conformal mapping onto a rectangle. *Constr. Approx.*, 7:349–379, 1991.

[65] N. Papamichael and N. S. Stylianopoulos. A domain decomposition method for approximating the conformal modules of long quadrilaterals. *Numer. Math.*, 62:213–234, 1992.

[66] N. Papamichael, M. K. Warby, and D. M. Hough. The treatment of corner and pole-type singularities in numerical conformal mapping techniques. *J. Comp. Appl. Math.*, 14:163–191, 1986.

[67] K. Pearce. A constructive method for numerically computing conformal mappings for gearlike domains. *SIAM J. Sci. Stat. Comput.*, 12:231–246, 1991.

[68] I. Peterson. Beating a fractal drum: How a drum's shape affects its sound. *Science News*, 146:184–185, 1994.

[69] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction.* Springer–Verlag, 1985.

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

[71] K. Reppe. Berechnung von Magnetfeldern mit Hilfe der konformen Abbildung durch numerische Integration der Abbildungsfunktion von Schwarz–Christoffel. *Siemens Forsch. u. Entwickl. Ber.*, 8:190–195, 1979.

[72] S. Sherwin and G. Karniadakis. Triangular and tetrahedral spectral elements. Brown University Technical Report CFM 95-9, 1995.

[73] B. Smith, P. Bjørstad, and W. Gropp. *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge University Press, 1996.

[74] K. P. Sridhar and R. T. Davis. A Schwarz–Christoffel method for generating two-dimensional flow grids. *J. Fluids Eng.*, 107:330–337, 1985.

[75] S. Sridhar and A. Kudrolli. Experiments on not "hearing the shape" of drums. *Phys. Rev. Let.*, 72:2175–2178, 1994.

[76] G. Starke and R. S. Varga. A hybrid Arnoldi–Faber iterative method for nonsymmetric systems of linear equations. *Numer. Math.*, 64:213–240, 1993.

[77] E. Stiefel. On solving Fredholm integral equations: Applications to conformal mapping and variational problems of potential theory. *SIAM J. Numer. Anal.*, 4:63–85, 1956.

[78] G. Strang and G. J. Fix. *An Analysis of the Finite Element Method.* Prentice-Hall, 1973.

[79] G. T. Symm. An integral equation method in conformal mapping. *Numer. Math.*, 9:250–258, 1966.

[80] A. E. Trefethen, V. S. Menon, C.-C. Chang, G. J. Czajkowski, C. Myers, and L. N. Trefethen. MultiMATLAB: MATLAB on multiple processors. Technical Report CTC96TR239, Cornell Theory Center, 1996.

[81] L. N. Trefethen. Numerical computation of the Schwarz-Christoffel transformation. *SIAM J. Sci. Stat. Comput.*, 1:82–102, 1980.

[82] L. N. Trefethen. Analysis and design of polygonal resistors by conformal mapping. *Z. Angew. Math. Phys.*, 35:692–704, 1984.

[83] L. N. Trefethen, editor. *Numerical Conformal Mapping.* North-Holland, 1986.

[84] L. N. Trefethen. SCPACK User's Guide. MIT Numerical Analysis Report 89-2, 1989.

[85] L. N. Trefethen. Numerical construction of conformal maps. Appendix to *Fundamentals of Complex Analysis for Mathematics, Science, and Engineering*, by E. B. Saff and A. D. Snider, Prentice Hall, 1993.

[86] L. N. Trefethen. Schwarz-Christoffel mapping in the 1980's. Cornell University Computer Science Department Technical Report TR 93-1381, 1993.

[87] S. Warschawski. On Theodorsen's method of conformal mapping of nearly circular regions. *Quart. J. Appl. Math.*, 3:12–28, 1945.

[88] R. Wegmann. An iterative method for conformal mapping. *J. Comp. Appl. Math.*, 14:7–18, 1986.

[89] L. C. Woods. *The Theory of Subsonic Plane Flow.* Cambridge Univ. Press, 1961.

[90] H. Wu, D. W. L. Sprung, and J. Martorell. Numerical investigation of isospectral cavities built from triangles. *Phys. Rev. E*, 51:703–708, 1995.