NUMERICAL COMPUTING WITH FUNCTIONS ON THE SPHERE AND DISK

by

Heather Denise Wilber

A thesis

submitted in partial fulfillment

of the requirements for the degree of

Master of Science in Mathematics

Boise State University

August 2016

© 2016

Heather Denise Wilber

ALL RIGHTS RESERVED

BOISE STATE UNIVERSITY GRADUATE COLLEGE

DEFENSE COMMITTEE AND FINAL READING APPROVALS

of the thesis submitted by

Heather Denise Wilber

Thesis Title: Numerical Computing with Functions on the Sphere and Disk

Date of Final Oral Examination: 18 May 2016

The following individuals read and discussed the thesis submitted by Heather Denise Wilber, and they evaluated her presentation and response to questions during the final oral examination. They found that the student passed the final oral examination.

Grady Wright, Ph.D.	Co-Chair, Supervisory Committee
Alex Townsend, Ph.D.	Co-Chair, Supervisory Committee
Jodi L. Mead, Ph.D.	Member, Supervisory Committee

The final reading approval of the thesis was granted by Grady Wright, Ph.D., Chair of the Supervisory Committee. The thesis was approved for the Graduate College by Jodi Chilson, M.F.A., Coordinator of Theses and Dissertations.

dedicated to Grady

ACKNOWLEDGMENTS

My efforts have been generously supported by fellowships granted through the NASA Idaho Space Grant Consortium and the Boise State University Mathematics Department and Graduate College.

This work would not be possible without Nick Trefethen and the many researchers associated with Chebfun: their discoveries and ideas are what led me to study numerical analysis. The encouragement and feedback I've received from the Chebfun development team have been critical to the creation of this thesis.

I am appreciative of Dr. Jodi Mead, who once advised me to "always pick the hard problems." I have taken this to heart. Dr. Alex Townsend is a vision of creativity and energy; the investment of his characteristic zest into my development as a researcher has been an incredible and lucky gift. Dr. Grady Wright is an extraordinary and thoughtful mathematician. He has been my teacher, in the deepest sense of the word, and this is an immeasurable fortune to have gained. My gratitude is inexpressible.

My friends and family have filled my time with encouragement, care, and countless kindnesses. I am especially thankful for Don and Janet, Ty and Kelsey, my parents, and Ashton Byers. My beautiful son, James, adds perpetual effervescence and levity to serious places, and this has been especially important. Finally, I thank my husband, Daniel, who shares with me the deepest form of friendship I know, and endlessly inspires me.

ABSTRACT

A new low rank approximation method for computing with functions in polar and spherical geometries is developed. By synthesizing a classic procedure known as the double Fourier sphere (DFS) method with a structure-preserving variant of Gaussian elimination, approximants to functions on the sphere and disk can be constructed that (1) preserve the bi-periodicity of the sphere, (2) are smooth over the poles of the sphere (and origin of the disk), (3) allow for the use of FFT-based algorithms, and (4) are near-optimal in their underlying discretizations. This method is used to develop a suite of fast, scalable algorithms that exploit the low rank form of approximants to reduce many operations to essentially 1D procedures. This includes algorithms for differentiation, integration, and vector calculus. Combining these ideas with Fourier and ultraspherical spectral methods results in an optimal complexity solver for Poisson's equation, which can be used to solve problems with 10^8 degrees of freedom in just under a minute on a laptop computer. All of these algorithms have been implemented and are publicly available in the open-source computing system called Chebfun [21].

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Polar and spherical geometries occupy a central role in scientific computing and engineering, with applications in fluid dynamics [39, 55], optics [41], astrophysics [3, 11, 29, 51, 63], weather forecasting and climate modeling [17, 20, 25, 40, 45, 49, 59], and geophysics [24,80]. Advances in these areas increasingly require accurate and effective approximation methods for functions defined on the unit disk or the surface of the unit sphere. To take advantage of convenient algorithms, coordinate transforms are often applied that map functions in polar and spherical geometries to rectangular domains. Unfortunately, this method has several significant drawbacks: it introduces one or more artificial singularities into the problem, making smoothness over the poles of the sphere (and origin of the disk) difficult to enforce. Interpolation schemes derived from these mappings typically oversample the function near the singularities, and this can severely hamper computational efficiency. Additionally, rectangular coordinate transforms for functions on the sphere destroy the inherent bi-periodicity of such functions, making the FFT inapplicable in one direction.

This thesis presents a new method for computing with functions in polar and spherical geometries. By synthesizing a classic technique known as the double Fourier sphere (DFS) method with a new structure-preserving Gaussian elimination procedure, many of the issues associated with rectangular coordinate transforms for polar and spherical geometries are effectively overcome. This results in an efficient and adaptive method of approximation for functions on the sphere and disk, with approximants that enjoy a plurality of desirable attributes: 1) A structure that allows the use of FFT-based algorithms in both variables, 2) smoothness over the poles of the sphere (and origin of the disk), 3) stability for differentiation, and 4) an underlying interpolation grid that rarely oversamples the function.

This approach is used to create an integrated computational framework for working with functions in polar and spherical settings, including the development and implementation of algorithms for integration, function evaluation, vector calculus, and the solving of Poisson's equation, among many other things. These ideas are implemented in a software package that is publicly available through the open-source Chebfun software system, which is written in MATLAB [21]. This development allows investigators to compute on the sphere and disk without concern for the underlying discretization or procedural details, providing an intuitive platform for data-driven computations, explorations, and visualizations, all occurring at an accuracy near machine precision.

This thesis is organized as follows: A review of global polynomial and trigonometric interpolation methods in the 1D setting is given in Section 2.1, as this forms a basis for our approach. The concept of low rank approximation for 2D functions is reviewed in Section 2.2, followed by an overview of existing methods of approximation for functions on the sphere (see Section 2.3) and the disk (see Section 2.4).

In Chapter 3, a classic procedure known as the double Fourier sphere (DFS) method, as well as its disk analogue, are used to introduce the concept of *block-mirror-centrosymmetric* (BMC) structure (see Section 3.1). A new, BMC structure-preserving Gaussian elimination (GE) procedure is described in Section 3.2, and this method is used to construct low rank approximations to functions on the sphere and the disk. The relationship between this GE procedure, BMC structure, and parity properties of functions in polar and spherical geometries is discussed in Section 3.4. Section 3.5 provides theoretical results related to convergence of the GE procedure, and this is followed by a discussion of near-optimal convergence behaviors observed in practice (see Section 3.6).

Chapter 4 describes a collection of algorithms for computing on the sphere (see Section 4.2) and disk (see Section 4.3) through the use of low rank approximations. These algorithms have been implemented in the Chebfun computing system and are available for exploration at `www.chebfun.org`.

Chapter 5 applies the DFS method in conjunction with Fourier and ultraspherical spectral methods to develop optimal Poisson solvers for both the sphere (see Section 5.1) and the disk (see Section 5.2).

Appendix A offers a collection of observations on the properties of BMC functions, including a discussion of linear algebra related to discretizations of BMC functions (see Section A.1), and a derivation of the SVD for BMC functions (see Section A.2). Appendix B provides an overview of the ultraspherical spectral method, which is used to formulate a Poisson solver on the disk.

Author Contributions and Related Publications

This thesis is the result of the collective efforts of myself, Prof. Grady Wright (Boise State Univ.), and Prof. Alex Townsend (MIT). Results related to the sphere and the initial conception of BMC structure-preserving GE were largely collaborative. With the guidance of my advisors, I independently extended these concepts to the disk, developed the algorithms for numerically computing with functions on the disk, and created the Diskfun software. I was also the lead author of our paper about numerical computing with functions on the disk [71]. While writing this thesis and working closely with my advisors, I additionally developed several new results and observations that apply to functions on the sphere and the disk. This includes a proof that the BMC structure-preserving GE procedure in Chapter 3 converges *geometrically* (see Section 3.5), an explicit formulation and formal proof that the GE algorithm preserves BMC structure (see Section 3.2), and an explicit description of precisely why approximants that have BMC structure are differentiable on the sphere and disk (see Sections 4.2.4 and 4.3.4).

Overall, two papers related to this research have been produced [70, 71]; this work complements these papers by providing additional insights, results, and examples. This includes an extended discussion of key notions from approximation theory and alternate methods of approximation in polar and spherical geometries in Chapter 2, new results related to the convergence properties of the GE algorithm and parity properties associated with BMC functions in Chapter 3, the development of a *weighted* SVD algorithm for functions on the sphere, as well as additional details concerning differentiation on the sphere in Chapter 4, and an extended description of a highly optimized Poisson solver for the disk in Chapter 5. These insights are supplemented by additional materials contained in the appendices, including observations on BMC matrices and an explicit derivation of the SVD for BMC functions (Appendix A), and an overview of the ultraspherical spectral method (Appendix B).

CHAPTER 2

BACKGROUND

In [4], global polynomial interpolants are used to create a framework for numerically computing with 1D functions, and this idea is extended to periodic 1D functions in [81] through the use of trigonometric interpolants. These ideas are implemented within the software system Chebfun. The extension of these ideas to 2D is presented in [67], and in this thesis, we develop an analogous approximation method for computing with functions on the sphere and disk. Using this method, many operations on bivariate functions can be performed through essentially 1D procedures involving univariate Fourier and Chebyshev expansions. This makes the method especially amenable to implementation within Chebfun, where highly optimized algorithms for a range of operations involving 1D functions are available. We have implemented our approximation method and the associated algorithms in the Spherefun and Diskfun software systems, which are available as an integrated part of Chebfun.

Our approximation method requires the use of global polynomial and trigonometric interpolants to 1D functions, and relies on the convergence properties associated with these interpolants (see Section 2.1). We also use recent developments in low rank approximation methods for 2D functions (see Section 2.2.2), and classic techniques associated with numerically representing functions on the sphere and disk (see Sections 2.3 and 2.4, respectively). This chapter reviews these important ideas, and also gives a broad overview of alternative techniques currently used for computing with functions in polar and spherical geometries.

2.1 1D Function Approximation

2.1.1 Trigonometric Polynomial Interpolation

For approximations to functions on the sphere and disk, we will require the use of interpolants for periodic functions on $[-\pi, \pi]$. Let $f : [-\pi, \pi] \to \mathbb{C}$ be a 2π-periodic function that is Lipschitz continuous. Then, f has a unique Fourier series that converges absolutely and uniformly:

$$f(x) = \sum_{k=-\infty}^{\infty} \tilde{c}_k e^{ikx}, \qquad x \in [-\pi, \pi], \tag{2.1}$$

where $\tilde{c}_k = \frac{1}{2\pi}\int_{-\pi}^{\pi} f(x)e^{-ikx}\,dx$ are the Fourier coefficients of f. Truncating (2.1) gives an approximation to f:

$$f_m(x) = \sum_{k=-m}^{m} \tilde{c}_k e^{ikx}, \qquad x \in [-\pi, \pi]. \tag{2.2}$$

The function f_m is a trigonometric polynomial of degree m, referred to as the degree m *Fourier projection* of f. To compute (2.2), we can approximate the integrals defining each $\tilde{c}_k$ by using the trapezoidal rule with the following $2m + 1$ quadrature points:

$$x_k = -\pi + \frac{2k\pi}{2m+1}, \qquad 0 \le k \le 2m. \tag{2.3}$$

However, as described in [81], this is equivalent to interpolating f at the points given in (2.3) with the basis functions $\{e^{ikx}\}$. This finds $p_m(x)$, the *trigonometric interpolant* to f of degree m, which can be written as

$$p_m(x) = \sum_{k=-m}^{m} c_k e^{ikx}, \qquad x \in [-\pi, \pi]. \tag{2.4}$$

The coefficients $\{c_k\}$ and $\{\tilde{c}_k\}$ are related to one-another through an *aliasing* formula, so that theoretical results for (2.2) have analogous interpretations for (2.4) [81]. This is useful because (2.2) is convenient for analytical work, but it is easier numerically to use (2.4). Here, we list key convergence properties for p_m (see [81] and [72] for related results on f_m). The first result is related to the bounded total variation of a function, defined as follows:

Definition 2.1 (Bounded total variation)**.** *A function $f(x)$, $x \in [a,b]$ is said to be of V bounded total variation if*

$$V = \int_a^b |f'(x)|\, dx < \infty.$$

This notion can be used to understand the rate of convergence for trigonometric interpolants of differentiable functions. In the following theorems, $||\cdot||_\infty$ is the infinity norm, i.e., $||f||_\infty = \sup\{|f(x)| : x \in [-\pi,\pi]\}$.

Theorem 2.1 (Convergence for differentiable periodic functions)**.** *For $\nu \geq 1$, let f be ν times differentiable, 2π-periodic function on the interval $[-\pi,\pi]$, with $f^{(\nu)}$ of bounded total variation V. Let p_m be the degree m trigonometric interpolant of f as in* (2.4)*. Then, for $m \geq \nu$,*

$$||f - p_m||_\infty \leq \frac{2V}{\pi \nu m^\nu},$$

i.e., $||f - p_m||_\infty = \mathcal{O}(m^{-\nu})$.

If f is analytic, convergence depends on the region for which f is analytically continuable in the complex plane.

Theorem 2.2 (Convergence for analytic periodic functions)**.** *If f is a 2π-periodic function on the interval $[-\pi,\pi]$, $f(z)$ is analytic, and for some finite constant $M > 0$,*

$|f(z)| \leq M$ in an open strip of half-width $\alpha > 0$ around the real axis in the complex z-plane, then

$$||f - p_m||_\infty \leq \frac{4Me^{-\alpha m}}{e^\alpha - 1},$$

i.e., $||f - p_m||_\infty = \mathcal{O}(e^{-\alpha m})$.

The proofs for these theorems are given in [81], and rely on related theorems about the rates of decay of the coefficients in (2.4). These theorems state specific results associated with a general maxim in approximation theory: the smoother a function is, the faster its approximants converge. Trigonometric interpolants to periodic functions that are ν-times differentiable (with a ν^{th} derivative of bounded total variation) converge at algebraic rates. Interpolants to functions that are analytic converge at geometric rates, and interpolants to functions that are entire converge super-geometrically. Thus, for sufficiently smooth periodic functions, trigonometric interpolation offers an excellent method of approximation.

Henrici describes a procedure for evaluating p_m at any point $x^* \in [-\pi, \pi]$ in only $\mathcal{O}(N)$ operations [35], where $N = 2m+1$ is the number of interpolation points in (2.3). Alternatively, p_m can be uniquely represented by the set of coefficients $\{c_k\}_{k=-m}^{m}$. Given the values of $f(x_k)$ at each x_k in (2.3), the FFT finds $\{c_k\}_{k=-m}^{m}$ in only $\mathcal{O}(N \log N)$ operations. Likewise, if $\{c_k\}_{k=-m}^{m}$ are known, the inverse FFT provides an efficient way to sample f, givings its values at each x_k in (2.3) in $\mathcal{O}(N \log N)$ operations.

2.1.2 Chebyshev Polynomial Interpolation

For numerically representing functions on the disk, we require approximations to functions defined on the interval $[-1, 1]$.

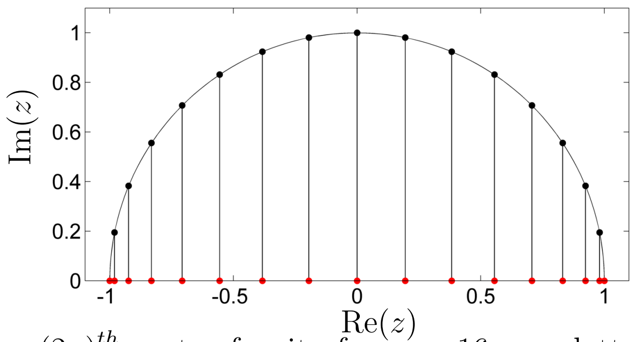


Figure 2.1: The $(2n)^{th}$ roots of unity for $n = 16$ are plotted on the unit circle. Their projection to the real axis results in the Chebyshev points (red) on the interval $[-1, 1]$.

Let $f : [-1, 1] \to \mathbb{C}$ be any continuous function. Without the assumption of periodicity, interpolants to f at equally-spaced points over $[-1, 1]$ become exponentially ill-conditioned, with interpolants at n points potentially producing numerical errors of size $\mathcal{O}(2^n)$, even in cases where f is analytic on $[-1, 1]$ [74, Ch. 13]. For this reason, we seek an alternate set of interpolation points.

Consider the set of $n + 1$ equally-spaced angles $\{\theta_k\}_{k=0}^{n}$, $\theta_k \in [0, \pi]$. These angles are the arguments for the $(2n)^{th}$ roots of unity $\{z_k = e^{ik\pi/n}\}_{k=0}^{n}$. The points $x_k = \text{Re}(z^k)$ are called the *Chebyshev* points, and can be expressed conveniently as

$$x_k = -\cos\left(\frac{k\pi}{n}\right), \qquad 0 \le k \le n. \tag{2.5}$$

Interpolating f at the $n + 1$ Chebyshev points gives an approximation to f that is closely related to the expansion of f in the Chebyshev polynomial basis. The Chebyshev polynomial of degree n is defined as

$$T_n(x) = \cos(n\theta), \qquad \theta = \cos^{-1}(x), \qquad x \in [-1, 1].$$

Chebyshev polynomials are orthogonal with respect to the weight function $(1-x^2)^{-1/2}$, so that

$$< T_m, T_n >_w = \frac{2}{\pi}\int_{-1}^{1}\frac{T_m(x)T_n(x)}{\sqrt{1-x^2}} = \begin{cases} 2, & m = n = 0, \\ 1, & m = n,\ m, n \neq 0 \\ 0, & m \neq n. \end{cases} \tag{2.6}$$

These polynomials form a complete basis for functions on $[-1, 1]$ that are square-integrable with respect to (2.6), and for such a function f, there exists a unique series

$$f(x) = \sum_{k=0}^{\infty} \tilde{a}_k T_k(x), \qquad x \in [-1, 1], \tag{2.7}$$

that converges absolutely and uniformly to f. Truncating this series to $n + 1$ terms forms the approximant f_n, known as the degree n Chebyshev *projection* of f. As with trigonometric polynomial approximation, it can be more convenient computationally to consider the unique interpolant to f at the $n + 1$ Chebyshev points:

$$p_n(x) = \sum_{k=0}^{n} a_k T_k(x), \qquad x \in [-1, 1]. \tag{2.8}$$

Here, each a_k is related to $\tilde{a}_k$ through an aliasing formula [74], and theoretical results for f_n correspond closely to results for p_n. Chebyshev interpolants are known to have very good convergence properties for functions with some degree of smoothness, and we give two essential results here.

Theorem 2.3 (Convergence for differentiable functions)**.** *Let f be $\nu \geq 1$ times differentiable on $[-1, 1]$ with $f^{(\nu)}$ of bounded variation V. If p_n is the degree n Chebyshev interpolant of f given by* (2.8)*, then for any $n \geq \nu$,*

$$||f - p_n||_\infty \leq \frac{4V}{\pi\nu(n-\nu)^\nu},$$

i.e., $||f - p_n||_\infty = \mathcal{O}(n^{-\nu})$.

If f is analytic on $[-1, 1]$, the rate of convergence depends on the region for which f is analytically continuable; this can be formalized through the concept of *Berstein*

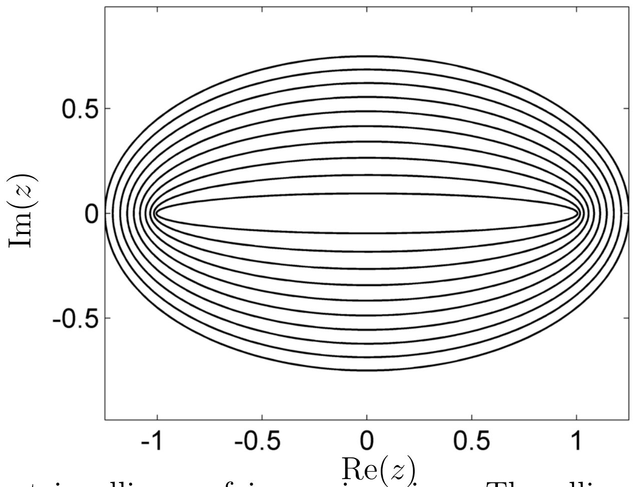


Figure 2.2: Berstein ellipses of increasing size. The ellipses correspond to the respective parameters $\rho = 1.1, 1.2, \ldots 1.8, 2$, with innermost ellipse having $\rho = 1.1$.

ellipses, defined below.

Definition 2.2 (Berstein ellipse). *The Berstein ellipse E_ρ, $\rho > 1$, is the open region of the complex plane bounded by an ellipse with foci at ± 1 and a semimajor and semiminor axis that sum to ρ.*

Figure 2.2 displays Berstein ellipses for several choices of ρ. Using Berstein ellipses, we can precisely describe the convergence behavior of Chebyshev interpolants to analytic functions.

Theorem 2.4 (Convergence for analytic functions). *Let f be analytic on $[-1, 1]$ and analytically continuable to the Berstein ellipse E_ρ, satisfying $|f| < M$ for some constant $M > 0$. If p_n is the degree n Chebyshev interpolant to f, then*

$$||f - p_n||_\infty \leq \frac{4M\rho^{-n}}{\rho - 1},$$

i.e., $||f - p_n||_\infty = \mathcal{O}(\rho^{-n})$.

Proofs of Theorems 2.3 and 2.4 can be found in [74]. These theorems show that for sufficiently smooth functions, Chebyshev interpolants have excellent convergence properties. In fact, Chebyshev interpolants offer a *near-best* approximation for continuous functions. In [74, Ch. 16], it is shown that if f is continuous on $[-1,1]$ and p_n^* is the best degree n polynomial approximation to f with respect to the infinity norm, then

$$||f - p_n||_\infty \leq \big(\frac{2}{\pi}\log(n+1) + 2\big)||f - p_n^*||_\infty.$$

This inequality states that the difference between the best approximation to f and the Chebyshev interpolant to f is at maximum $\mathcal{O}(\log n)$. Asymptotically, one cannot do better than this: it is shown in [12] that for any set S of $n+1$ distinct points on $[-1,1]$, there always exists a continuous function f such that the polynomial interpolant $p_n^\dagger$ to f on S satisfies

$$||f - p_n^\dagger||_\infty \geq \big(1.52125 + \frac{2}{\pi}\log(n+1)\big)||f - p_n^*||_\infty.$$

It is in this sense that Chebyshev interpolants offer a *near-best* approximation to f, and this idea is made precise in [74, Ch. 15].

Chebyshev interpolants possess one more important quality: Given the values of f at the $n+1$ Chebyshev points, the coefficients in (2.8) can be computed through the fast cosine transform in only $\mathcal{O}(n\log n)$ operations. The inverse of this operation also costs only $\mathcal{O}(n\log n)$ operations, providing a convenient way to sample f when the coefficients in (2.8) are known.

2.2 Low Rank Approximation for 2D Functions

In [67], a general approach for computing with 2D functions over a bounded rectangular domain is established through methods of low rank approximation, and we will use this idea in Chapter 3 to develop a low rank approximation method for functions in spherical and polar geometries. Here, we review the concepts from [67].

Let $f(x,y)$ be a bivariate function on $[-1,1]^2$, and note that any function on a more generalized rectangular domain can be mapped to $[-1,1]^2$ by a change of variables. A nonzero function f is called a rank 1 function if it can be written as a product of two univariate functions, i.e., $f(x,y) = c(y)r(x)$. A function is of at most rank K if it can be written as a sum of K rank 1 functions. While most functions are mathematically of infinite rank, smooth functions can typically be approximated to machine precision by a rank K truncation, i.e.,

$$f(x,y) \approx \underbrace{\sum_{j=1}^{K} c_j(y) r_j(x),}_{f_K} \tag{2.9}$$

for some relatively small K [64]. In practice, we are interested in the *numerical rank* of f. This is the minimum rank required to approximate f within some tolerance, such as machine epsilon, using any bounded function on $[-1,1]^2$ of finite rank. For a prescribed value $\epsilon > 0$, this is given by

$$k_\epsilon = \min\{K \in \mathbb{N} : \inf_{f_K} ||f - f_K||_\infty \leq \epsilon ||f||_\infty\},$$

where the inner infimum is taken over the set of bounded rank K functions on $[-1,1]^2$ [64].

The primary advantage of using low rank approximations is evident in [67] and in Chapter 4 of this thesis, where several algorithms are devised that exploit the low rank form in order to use highly efficient 1D procedures. To use this form of approximation effectively, several questions are in order: (1) Can every function be written as an expansion of rank 1 terms? (2) Is there an efficient mechanism for constructing low rank approximations to an arbitrary function? (3) What are the convergence properties of low rank approximations to functions? We will additionally be concerned with whether an adaptive low rank approximation method can be developed that preserves inherent geometric features of functions defined in polar and spherical geometries.

2.2.1 The Singular Value Decomposition

We examine questions (1)-(3) by considering the *best* rank K approximation to f. For the L_2 norm, this is given by the *singular value decomposition* (SVD), also called the Karhunen–Loève expansion, for bivariate functions [53]. It is shown in [33] that if f is Lipschitz continuous in both variables, then the SVD converges absolutely and uniformly to f. Then, f is expressed by

$$f(x,y) = \sum_{j=1}^{\infty} \sigma_j u_j(y) v_j(x), \qquad (x,y) \in [-1,1]^2. \tag{2.10}$$

Here, the *singular values* $\{\sigma_j\}_{j=1}^{\infty}$ are non-negative, real, nonincreasing, and $\lim_{k\to\infty} \sigma_k \to 0$. The sets of continuous *singular functions*, $\{u_j\}_{j=1}^{\infty}$ and $\{v_j\}_{j=1}^{\infty}$, are each orthonormal with respect to the L_2 inner product on $[-1,1]$. Furthermore, the set $\{\sigma_j\}$ is uniquely determined for f, and the singular functions corresponding to each simple σ_j are unique up to complex signs.

In [53], it is shown that the best rank K approximation to f in the L_2 norm is formed by truncating (2.10) to K terms. For this reason, the SVD is said to provide an optimal rank K approximation to f. In [69], it is shown that properties of convergence for the SVD are linked to the smoothness of the function. Given any fixed choice of $x^* \in [-1,1]$, suppose that $f(x^*, y)$ is ν times differentiable in y with a ν^{th} derivative of variation that is uniformly bounded with respect to x^*. If f_K is the rank K truncation of the SVD of f, then the error $||f - f_K||_\infty$ decays at the rate of $\mathcal{O}(K^{-\nu})$. If all $f(x^*, y)$ are analytically continuable to a Bernstein ellipse of parameter ρ in the complex plane that contains the interval $[-1, 1]$, then $||f - f_k||_\infty$ decays at a geometric rate prescribed by ρ [69].

For such functions, the continuous SVD can be explicitly constructed by first performing the continuous analogue to LU factorization on f, and then performing a QR factorization the resulting *quasimatrices* [69]. Alternatively, one could sample f and numerically compute the SVD of the resulting matrix using standard techniques. The cost of computing the SVD is $\mathcal{O}(N^3)$ operations, where N is the number of samples required to resolve f to a specified tolerance in both x and y.

The SVD shows us that at least one method of low rank approximation exists for a large class of functions. The approximants have very good convergence properties, but they are computationally expensive to construct. For this reason, we seek an alternative technique.

2.2.2 Iterative Gaussian Elimination on Functions

Given a matrix A of rank n, $K < n$ steps of Gaussian elimination (GE) with complete or rook pivoting can be used to construct a near-best rank K approximation to A, provided that the singular values of A decay sufficiently fast [28]. Using a variety of pivoting strategies, methods such as adaptive cross approximation [6], two-sided interpolative decomposition [32], and Geddes–Newton approximation [16] apply GE in the continuous setting to create low rank approximations to functions. In [67], an adaptive, iterative variant of GE with complete pivoting is used to develop a framework for computing with 2D functions in the Chebfun software system. Our procedure for approximating functions in polar and spherical geometries is based on an extension of this idea (see Chapter 3).

Standard GE on a matrix A with complete pivoting proceeds by choosing the entry with the maximal absolute value, $A(i, j)$, as a pivot. At each GE step, a rank 1 matrix is formed from this pivot and subtracted from A:

$$A \leftarrow A - A(:, j)A(i, :)/A(i, j),$$

where $A(:, j)$ is j^{th} column of A, and $A(i, :)$ is the i^{th} row of A. This step zeros out the row and column containing the pivot and also reduces the rank of A by one. The resulting matrix is called the *residual.* The process is then repeated on the residual, and the algorithm proceeds until the residual is the zero matrix. Summing the first K rank 1 matrices formed within this process provides a rank K approximation to A.

In a similar way, given a function $f(x, y)$, denote the maximum absolute value of f for $(x, y) \in [-1, 1]^2$ as $f(x^*, y^*)$. Replacing A with f, the GE step is given by

$$f(x,y) \quad \longleftarrow \quad f(x,y) - \underbrace{\frac{f(x^*,y)f(x,y^*)}{f(x^*,y^*)}}_{\text{A rank 1 approx. to } f}. \tag{2.11}$$

In this scheme, the functions $f(x^*, y)$ are referred to as "column slices" of f. Similarly, $f(x, y^*)$ are "row slices." As with GE on matrices, the GE step produces a residual such that $f(x^*, y) = f(x, y^*) = 0$. Unlike a matrix, f is typically of infinite rank, so the GE procedure is terminated after the absolute maximum of the residual falls below some specified relative tolerance, such as the product of machine epsilon and the (approximate) maximum value of the function. The number of steps required to achieve this is an upper bound on the numerical rank[1] of f, and as discussed in [64],

[1]This upper bound is a good estimate of the true numerical rank of f, and we use it interchangeably with the term *numerical rank* in the remainder of this thesis.

smooth functions are typically of low numerical rank.

Each time we apply a GE step to f, a rank 1 function is selected. After K steps, we can collect these rank 1 functions and construct a rank K approximation to f:

$$f(x,y) \approx \sum_{j=1}^{K} d_j c_j(y) r_j(x). \tag{2.12}$$

Here, d_j is a coefficient derived from the j^{th} pivot, and $c_j(y)$ and $r_j(x)$ are the j^{th} column and row slices, respectively, selected during the GE procedure. The computational cost of performing K steps of GE on f is $\mathcal{O}(K^3)$. Each univariate function in (2.12) is then adaptively resolved in $\mathcal{O}(K^2(m+n))$ operations, where m and n are the maximum number of samples required to resolve each $c_j(y)$ and $r_j(x)$, respectively [67].

In [64], it is shown that this GE procedure is a near-optimal and highly efficient method for approximating 2D functions in standard rectangular domains. We now turn to the specific challenges associated with numerically representing functions in polar and spherical geometries.

2.3 Existing Approximation Methods for Functions on the Sphere

To numerically represent 2D functions that are defined on the surface of the unit sphere, it is convenient to relate computations with functions on the sphere to functions defined over a rectangular domain. Given a function $f(x,y,z)$ in Cartesian coordinates, the spherical coordinate transform is given by

$$x = \cos\lambda \sin\theta, \quad y = \sin\lambda\sin\theta, \quad z = \cos\theta, \qquad (\lambda,\theta) \in [-\pi,\pi]\times[0,\pi], \tag{2.13}$$

where λ is the azimuth angle and θ is the polar (or zenith) angle (measured from the north pole). This change of variables allows one to compute with the function $f(\lambda, \theta)$, as opposed to $f(x, y, z)$ restricted to the surface of the sphere.

Unfortunately, this transform introduces artificial singularities at the north and south poles, since, by (2.13), for all $\lambda \in [-\pi, \pi]$, $(\lambda, 0)$ maps to $(0, 0, 1)$, and (λ, π) maps to $(0, 0, -1)$. Furthermore, the mapping does not preserve the natural periodicity of f in the latitude direction. The introduced singularities act as north and south boundaries on the rectangle, making smoothness over the poles difficult to enforce. This mapping is also problematic for interpolation schemes, as it results in grids on the sphere that are severely and redundantly clustered near the poles.

The approach we develop in Ch. 3 resolves these issues. Through the use of the double Fourier sphere (DFS) method (see Section 3.1), the periodicity of f in θ is recovered. We combine this scheme with a new, structure-preserving low rank approximation method, and this mitigates the unphysical effects of the artificial singularities, as well as issues associated with oversampling f near the poles. Below, we review some alternative techniques for computing on the sphere in order to put our new method in context.

2.3.1 Spherical Harmonic Expansions

Spherical harmonics are an appealing choice for representing functions on the surface of the sphere [2, Chap. 2] because they are analogous to trigonometric expansions for periodic functions. They have been used extensively and effectively in weather forecasting [20, Sec. 4.3.2], least-squares filtering [37], and for finding the numerical solution of separable elliptic equations. The truncated spherical harmonic expansion of f of degree N is

$$f(\lambda,\theta) \approx \sum_{\ell=0}^{N} \sum_{m=-\ell}^{\ell} c_{\ell,m} Y_\ell^m(\lambda,\theta), \tag{2.14}$$

where Y_ℓ^m is the spherical harmonic function with degree ℓ and order m [47, Sec. 14.30], and

$$c_{\ell,m} = \int_0^{2\pi} \int_0^{\pi} Y_\ell^m(\lambda,\theta) f(\lambda,\theta) \sin\theta \, d\theta \, d\lambda.$$

The truncated expansion (2.14) provides essentially uniform resolution of f over the sphere. The coefficients $0 \le \ell \le N$ and $c_{\ell,m}$ for $-\ell \le m \le \ell$ in (2.14) can be approximated with a discrete spherical harmonic transform, and there are fast $\mathcal{O}(N^2 \log N)$ complexity algorithms available for these transforms [46, 76]. However, highly adaptive discretizations are computationally unfeasible in this setting due to a high preprocessing cost associated with these algorithms. In our setting, fast and highly optimized algorithms are more readily available via the FFT.

2.3.2 Quasi-isotropic Grid-based Methods

Quasi-isotropic grid-based methods, such as those that use the "cubed-sphere" ("quad-sphere") [52, 62], geodesic (icosahedral) grids [5], or equal area "pixelations" [31], partition the sphere into (spherical) quads, triangles, or other polyhedra. This avoids introducing artificial singularities and results in far less oversampling of functions compared to grids designed by standard coordinate transforms. They are particularly useful for computations in which 3-5 digits of accuracy are sought and they are also highly amendable to parallelization. However, for 8-15 digits of accuracy, sample points usually become clustered along the artificial boundaries associated with these grids. Our GE procedure takes function samples that are adaptively selected during the approximation process, and our interpolation grids are composed of a sparse tensor product of 1D uniform grids (see Figure 3.6) that only cluster if the function itself requires it.

2.3.3 Radial Basis Functions

Radial basis functions (RBFs), or spherical basis functions [23], allow one to place sampling points at any location on the sphere. This is highly advantageous as sampling can be tailored to capture a function's features of interest, and these methods have been successfully applied in numerical weather prediction and solid earth geophysics calculations [24, 80].

While spectral accuracy is possible with these methods, the state-of-the-art algorithms have a computational complexity that grows cubically in the total degrees of freedom [27]. Consequently, these methods are currently too costly for general purpose computations with functions on the sphere.

2.3.4 The Double Fourier Sphere Method

The double Fourier sphere (DFS) method is a technique that can be used to recover the periodicity of the sphere in the latitude direction [45]. This method forms one of the pillars of our new approximation technique, so we hold off reviewing it until Chapter 3.

2.4 Existing Approximation Methods for Functions on the Disk

A function $g(x, y)$ defined in Cartesian coordinates on the unit disk can be converted to a function in polar coordinates, $g(\theta, \rho)$, through the transformation

$$x = \rho \cos\theta, \quad y = \rho \sin\theta, \qquad (\theta, \rho) \in [-\pi, \pi] \times [0, 1]. \tag{2.15}$$

This relates a function on the disk to a function defined over a rectangular domain, where more convenient algorithms can be applied. Unfortunately, this transform introduces an artificial singularity and unphysical boundary at $\rho = 0$. One solution involves expanding $g(\theta, \rho)$ in a basis that inherently enforces smoothness over $\rho = 0$, but such basis choices are not associated with fast transforms.

Unsatisfied with choosing between the use of fast algorithms and smoothness at the origin, we present an efficient method in Chapter 3 that prioritizes both. We expand g in the Fourier–Chebyshev basis so that fast, FFT-based algorithms are applicable, but combine an analogue of the DFS method (see Section 3.1) with a structure-preserving Gaussian elimination (GE) procedure to enforce that approximants are smooth over the origin. Below, we offer a brief discussion of alternative strategies.

2.4.1 Radial Basis Functions

As a mesh-free method, radial basis functions are useful for applications on a variety of geometries, such as the sphere and disk [26] (for more discussion, see Section 2.3.3). In [36] and [38], an algorithm for approximations on the disk is developed that arranges interpolation points so that the computational cost of the method reduces from $\mathcal{O}(N^3)$ to $\mathcal{O}(N \log N)$ operations, where N is the number of function samples taken. Unfortunately, ill-conditioning prevents this method from reaching machine precision, which is what we require.

2.4.2 Conformal Mapping

Using the inverse of the cosine leminiscate function, a function g on the unit disk can be mapped conformally to the unit square [1, 54]. This allows g to be expressed as a bivariate Chebyshev expansion so that FFT-based transforms are applicable.

Unfortunately, the mapping introduces four artificial singularities corresponding to the corners of the unit square. Interpolation grids based on this scheme are unnecessarily redundant near these singularities, and this adversely affects the computational efficiency gained from use of the FFT. In contrast, our approach also enables the use of FFT-based transforms, but we combine this with an adaptive low rank approximation procedure that does not oversample the function.

2.4.3 Basis Expansions

Observing in (2.15) that g is periodic in θ, an approximation to g can be obtained from a Fourier expansion in θ:

$$g(\theta, \rho) \approx \sum_{k=-m/2}^{m/2-1} \phi_k(\rho) e^{ik\theta}, \qquad (\theta, \rho) \in [-\pi, \pi] \times [0, 1], \tag{2.16}$$

where we assume m is an even integer[2] and $\phi_k(\rho)$ is a function to be selected. The most appropriate basis for expanding $\phi_k(\rho)$ is not immediately obvious. Below, we discuss three of the most common choices.

2.4.3.1 Cylindrical Harmonic Expansions

Cylindrical harmonics are the natural analogue of the spherical harmonics, as they are the eigenfunctions of the Laplace operator in cylindrical coordinates at a fixed

[2]Section 2.1.1 is a more generalized discussion of the properties of Fourier series, and there we use an index from $-m$ to m for convenience.

height [19]. If g is sufficiently smooth and satisfies $g(\theta, 1) = 0$, an approximation to g is given by

$$g(\theta, \rho) \approx \sum_{k=-m/2}^{m/2-1} \sum_{j=0}^{n-1} a_{jk} J_k(\omega_{kj}\rho) e^{ik\theta}, \qquad (\theta, \rho) \in [-\pi, \pi] \times [0, 1], \tag{2.17}$$

where J_k is the k^{th} order Bessel function, and ω_{kj} is the j^{th} positive root of J_k [19, Ch. 9]. This expansion can also be modified to include a inhomogeneous boundary condition. For (2.16) to be infinitely differentiable, each $\phi_k(\rho)$ must decay like ρ^k or faster near $\rho = 0$. The cylindrical harmonics are attractive because they inherently satisfy this condition. However, there is no readily available fast transform for computing the coefficients in (2.17), and for this reason, we do not use cylindrical harmonics in our approach.

2.4.3.2 One-sided Jacobi Polynomial Expansions

Expressing $\phi_k(\rho)$ with one-sided Jacobi polynomials results in an expansion of $g(\theta, \rho)$ in the Zernike polynomial basis, $Z_j^k(\theta, \rho)$ [10, 78]. This set of polynomials is considered theoretically analogous to the Legendre polynomials due to its orthogonality properties [7]. Using this expansion,

$$g(\theta, \rho) \approx \sum_{k=-m/2}^{m/2-1} \sum_{j=0}^{n-1} a_{jk} \rho^{|k|} P_j^{(0,k)}(2\rho^2 - 1) e^{ik\theta}, \tag{2.18}$$

where $P_j^{(0,k)}$ is the Jacobi polynomial of degree j with parameters $(0, k)$. Terms in this expansion inherently satisfy smoothness conditions because they include a k^{th} order zero at $\rho = 0$. The Zernike polynomials are often considered the basis of choice for approximation on the disk, and have been used to develop a method that results in sparse operators for solving the Helmholtz and Poisson equations [44]. Recently, a whole hierarchy of bases related to the one-sided Jacobi polynomials has been developed to capture the regularity of vector- and tensor-valued functions on the disk [77].

The primary disadvantage of using the one-sided Jacobi polynomials is the lack of a fast transform for computing the coefficients in (2.18). Since our approach relies on an interpolative, data-driven and adaptive scheme, we require fast transforms, and therefore do not make use of one-sided Jacobi expansions.

2.4.3.3 Fourier–Chebyshev Expansions

Expanding $\phi_k(\rho)$ in the Chebyshev basis gives the Fourier–Chebyshev approximation to g:

$$g(\theta, \rho) \approx \sum_{k=-m/2}^{m/2-1} \sum_{j=0}^{n-1} a_{jk} T_j(2\rho - 1) e^{ik\theta}, \qquad (\theta, \rho) \in [-\pi, \pi] \times [0, 1], \tag{2.19}$$

where $T_j(\cdot)$ is the degree j Chebyshev polynomial defined on $[-1, 1]$. By sampling g on an $n \times m$ Fourier–Chebyshev tensor product grid, the coefficients in (2.19) can be found in only $\mathcal{O}(mn \log(mn))$ using FFT-based algorithms. Problematically, this choice does not naturally impose smoothness over the origin, and grids based on this expansion unnaturally cluster near $\rho = 0$ [25]. Our approach resolves these drawbacks by combining the disk analogue to the DFS with a structure-preserving low rank construction procedure (see Chapter 3).

2.4.4 The Disk Analogue to the Double Fourier Sphere Method

There are several spectral collocation schemes involving Fourier–Chebyshev grids that alleviate the induced overresolution near $\rho = 0$ by double-sampling g so that $\rho = 0$ is not treated as a boundary [22, 25, 34]. In Section 3.1, we observe that this action is analogous to the DFS method. As it forms a crucial component of our new approximation strategy, we defer a detailed discussion until Chapter 3.

CHAPTER 3

LOW RANK APPROXIMATION OF FUNCTIONS IN SPHERICAL AND POLAR GEOMETRIES

In this chapter, we develop a new method for approximating functions on the sphere and disk. The double Fourier sphere (DFS) method (see Section 3.1) is applied to a function defined on the sphere or disk, and this introduces a crucial symmetry structure related to the geometry of the domain. In Section 3.2, we develop a variant of GE that generates low rank approximants while exactly preserving this structure, and in Sections 3.3-3.6, we analyze several features of the new GE procedure. This new method results in approximants with several desirable properties that can be exploited to create highly efficient algorithms (see Chapter 4).

We will begin by considering functions defined on the unit sphere. Each observation we make has an analogous interpretation for functions defined on the disk. At certain points in the chapter, it is more convenient to generalize by discussing a broader class of functions that is inclusive of functions on the sphere or disk.

3.1 Intrinsic Structures for Functions on the Sphere and Disk

Merilees [45] observed that a simple technique can be used to transform a function on the surface of the sphere to one on a rectangular domain while simultaneously preserving the periodicity of the function in the both the longitude *and* latitude directions. This idea, known as the double Fourier sphere (DFS) method, was developed further by Orszag [49], Boyd [8], Yee [82], Fornberg [25], and Cheong [17]. Transforming $f(x,y,z)$ on the sphere to a function $f(\lambda,\theta)$ in spherical coordinates using (2.13) yields

$$f(\lambda,\theta) = f(\cos\lambda\sin\theta, \sin\lambda\sin\theta, \cos\theta), \qquad (\lambda,\theta)\in[-\pi,\pi]\times[0,\pi].$$

With this transformation, the periodicity in the latitude direction has been lost. The DFS method proceeds by "doubling up" the function to form an extension of f on $(\lambda,\theta)\in[-\pi,\pi]\times[-\pi,\pi]$. The extension is given by

$$\tilde{f}(\lambda,\theta) = \begin{cases} p(\lambda+\pi,\theta), & (\lambda,\theta)\in[-\pi,0]\times[0,\pi], \\ q(\lambda,\theta), & (\lambda,\theta)\in[0,\pi]\times[0,\pi], \\ p(\lambda,-\theta), & (\lambda,\theta)\in[0,\pi]\times[-\pi,0], \\ q(\lambda+\pi,-\theta), & (\lambda,\theta)\in[-\pi,0]\times[-\pi,0], \end{cases} \tag{3.1}$$

where $p(\lambda,\theta) = f(\lambda-\pi,\theta)$ and $q(\lambda,\theta) = f(\lambda,\theta)$ for $(\lambda,\theta)\in[0,\pi]\times[0,\pi]$. The function $\tilde{f}$ is 2π-periodic in λ and θ, and is constant along the lines $\theta = 0$ and $\theta = \pm\pi$, corresponding to the poles. With a slight abuse of notation, we depict $\tilde{f}$ as

$$\tilde{f} = \begin{bmatrix} p & q \\ \texttt{flip}(q) & \texttt{flip}(p) \end{bmatrix}, \tag{3.2}$$

where `flip` refers to the MATLAB command that reverses the order of the rows of a matrix. The extension (3.1) is also called a glide reflection in group theory [42, §8.1].

As depicted in Figure 3.1 and seen in (3.2), the extended function $\tilde{f}$ has a structure close to a 2×2 centrosymmetric matrix, except that the last block row is flipped (mirrored). For this reason, we say that $\tilde{f}$ in (3.1) has a block-mirror-centrosymmetric (BMC) structure. This is formally defined as follows:

Definition 3.1. *(Block-mirror-centrosymmetric functions) Let $a, b\in\mathbb{R}^+$. A function $\tilde{f} : [-a,a]\times[-b,b]\to\mathbb{C}$ is a block-mirror-centrosymmetric (BMC) function if there* *are functions $p, q : [0,a] \times [0,b] \;\to \mathbb{C}$ such that*

$$\tilde{f}(\xi,\eta) = \begin{cases} p(\xi+a,\eta), & (\xi,\eta) \in [-a,0]\times[0,b], \\ q(\xi,\eta), & (\xi,\eta) \in [0,a]\times[0,b], \\ p(\xi,-\eta), & (\xi,\eta) \in [0,a]\times[-b,0], \\ q(\xi+a,-\eta), & (\xi,\eta) \in [-a,0]\times[-b,0]. \end{cases} \tag{3.3}$$

Using (3.1), every continuous function on the sphere can be extended to a continuous BMC function defined on $[-\pi,\pi]\times[-\pi,\pi]$ that is 2π-periodic (*bi-periodic*) in both variables. However, the converse is not true. It is possible to have a continuous BMC function that is bi-periodic but is not constant at the poles, i.e., along the lines $\theta = 0$ and $\theta = \pi$, and therefore is not associated with a continuous function on the sphere. For example, the function $f(\lambda,\theta) = \cos(2\theta)\cos(2\lambda)$ is a bi-periodic BMC function, but it is not constant when $\theta = 0$ or $\theta = \pi$. We capture this important aspect of BMC functions created through the DFS method with the following definition:

Definition 3.2 (BMC-I functions). *A function $\tilde{f} : [-a,a]\times[-b,b] \to \mathbb{C}$ is a Type-I BMC (BMC-I) function if it is a BMC function, and additionally, there are constants C_1 and C_2 such that $f(\cdot,0) = C_1$ and $f(\cdot,\pm b) = C_2$.*

By operating on the BMC-I function $\tilde{f}$, we can incorporate a structure associated with the sphere into our computations and relate the result back to f. It is fundamental that any approximant to $\tilde{f}$ possess BMC-I structure, and the structure-preserving GE procedure in Section 3.2 is designed with this in mind. Figure 3.2 illustrates the importance of this fact: pole singularities are introduced when the approximation strategy does not enforce BMC-I structure, and this reduces accuracy in subsequent computations, such as differentiation.

The DFS method can also be applied to functions on the disk. Consider the function $g(\theta,\rho)$ in (2.15), and recall that this transformation introduces an artificial

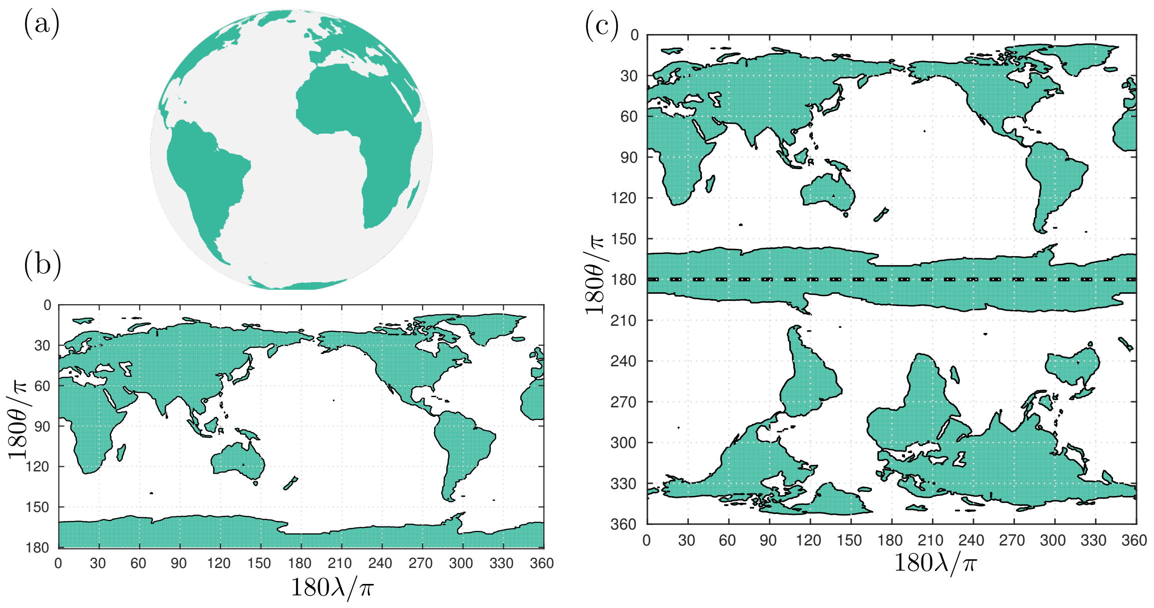


Figure 3.1: The DFS method applied to the globe. (a) An outline of the land masses on the surface of earth. (b) The projection of the land masses using latitude-longitude coordinates. (c) Land masses after applying the DFS method, shown in extended coordinates with the dashed line indicating the south pole. This is a BMC-I "function" that is periodic in longitude and latitude.

boundary at $\rho = 0$ (see Section 2.4). By expanding the domain of ρ from $[0, 1]$ to $[-1, 1]$, g can be extended to form a new function, $\tilde{g} : [-\pi, \pi] \times [-1, 1] \to \mathbb{C}$, so that $\rho = 0$ is no longer treated as a boundary. Mathematically, this extension of g can be expressed via functions $p(\theta, \rho)$ and $q(\theta, \rho)$, $(\theta, \rho) \in [0, \pi] \times [0, 1]$. The extended function $\tilde{g}$ is given by

$$\tilde{g}(\theta, \rho) = \begin{cases} q(\theta + \pi, \rho), & (\theta, \rho) \in [-\pi, 0] \times [0, 1], \\ p(\theta, \rho), & (\theta, \rho) \in [0, \pi] \times [0, 1], \\ q(\theta, -\rho), & (\theta, \rho) \in [0, \pi] \times [-1, 0], \\ p(\theta + \pi, -\rho), & (\theta, \rho) \in [-\pi, 0] \times [-1, 0], \end{cases} \tag{3.4}$$

and we observe in (3.4) that $\tilde{g}$ also possesses BMC symmetry (see Def. 3.3). This idea is conceptually analogous to the DFS method, although the original motivations for its development were quite different. On the disk, the method was devised in order

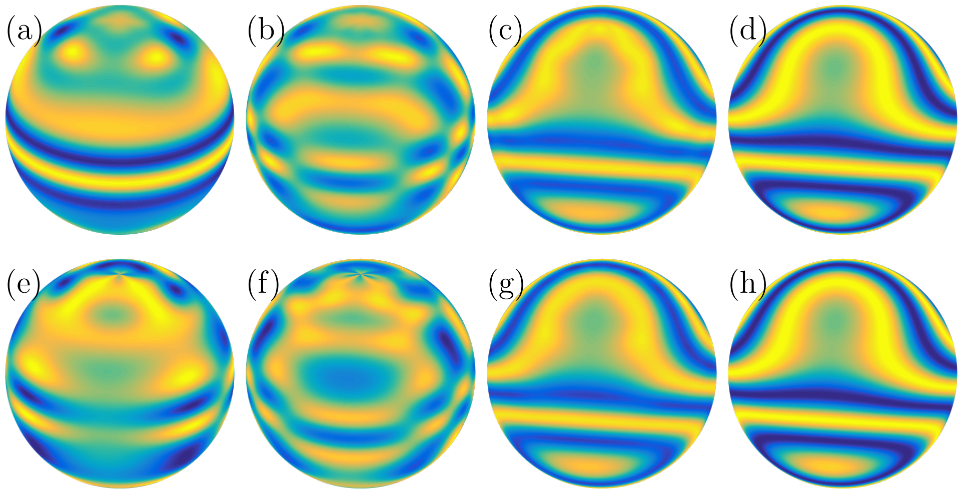


Figure 3.2: Low rank approximants to a function on the sphere, $\tilde{f} = \cos(1 + 2\pi(\cos\lambda\sin\theta + \sin\lambda\sin\theta) + 5\sin(\pi\cos\theta))$, $(\lambda,\theta) \in [-\pi,\pi]^2$. Figures (a)-(d) are the respective rank 2, 4, 8, and 16 approximants to $\tilde{f}$ constructed by the structure-preserving GE procedure in Section 3.2. Figures (e)-(h) are the respective rank 2, 4, 8, and 16 approximants to $\tilde{f}$ constructed by the standard GE procedure [67], which is not designed to preserve the BMC-I structure. In figures (e) and (f), one can see that a pole singularity is introduced when structure is not preserved.

to alleviate excessive grid clustering near $\rho = 0$ in Fourier–Chebyshev interpolation schemes [22, 25], and several variants have been proposed [10, 34].

Figure 3.3 depicts the disk analogue to the DFS method applied to the Nobel Prize medal. In addition to having BMC structure, $\tilde{g}$ must be constant along the line $\rho = 0$. This feature of $\tilde{g}$ is not shared by all BMC functions, but it is a crucial property of BMC functions associated with the disk. For this reason, we define a second variant of BMC functions:

Definition 3.3 (BMC-II function)**.** *A function $\tilde{g} : [-a,a] \times [-b,b] \to \mathbb{C}$ is a Type-II BMC (BMC-II) function if it is a BMC function and $g(\cdot, 0) = constant$.*

For BMC-II functions associated with the disk via (3.4), $a = \pi$, $b = 1$ and $\tilde{g}$ has the additional property of being 2π-periodic in θ. We now develop a procedure for

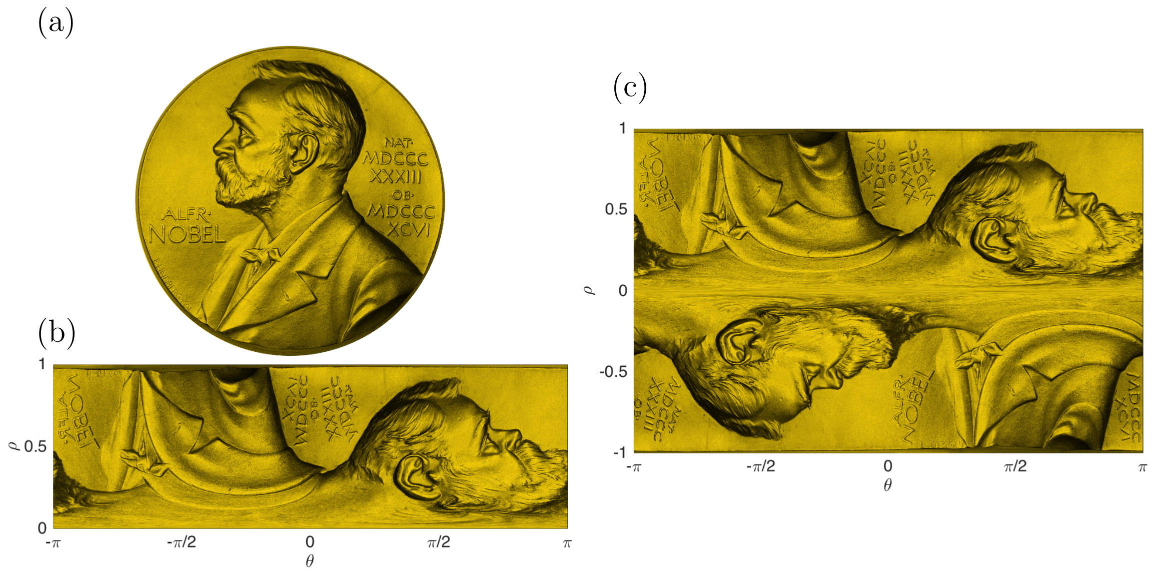


Figure 3.3: The disk analogue to the DFS method applied to the Nobel Prize medal. (a) The medal. (b) The projection of the medal using polar coordinates. (c) The medal after applying the disk analogue to the DFS method. This is a BMC-II "function" that is periodic in θ and defined over $\rho \in [-1, 1]$.

constructing approximants to BMC functions in a way that preserves BMC structure.

3.2 BMC Structure-preserving Gaussian Elimination

Using the DFS method and its disk analogue, we associate the functions f on the sphere and g on the disk with the BMC-I or BMC-II functions $\tilde{f}$ in (3.1) and $\tilde{g}$ in (3.4), respectively. If we construct approximants to these functions in a way that preserves BMC-I and BMC-II structures, we can perform computations that consistently remain associated with the geometry of the sphere or disk. In order to alleviate issues associated with oversampling the function (see Sections 2.3 and 2.4), we specifically seek a low rank approximation method.

Section 2.2.2 describes an efficient strategy for constructing low rank approximations to functions defined on rectangular domains. However, applying the GE step in (2.11) to a BMC function immediately destroys the BMC structure of the function, resulting in approximants that are rarely continuous on the sphere or disk (see Figure 3.2). To remedy this, we seek a new GE procedure that preserves BMC structure. We begin by considering a procedure applicable to *general* BMC functions, and then adapt the procedure to preserve additional features specifically associated with BMC-I or BMC-II functions.

3.2.1 A BMC Structure-preserving Gaussian Elimination Step

Without loss of generality, we use the BMC function $\tilde{f}$ on the sphere defined in (3.1) throughout the discussion. The procedure is the same for functions defined on the disk via (3.4).

Motivated by the pivoting strategy used for symmetric indefinite matrices [13], we employ GE with 2×2 pivots. Given $\tilde{f}$, a matrix M is constructed from values of $\tilde{f}$ so that for some $(\lambda^*, \theta^*) \in [0, \pi]^2$,

$$M = \begin{bmatrix} \tilde{f}(\lambda^* - \pi, \theta^*) & \tilde{f}(\lambda^*, \theta^*) \\ \tilde{f}(\lambda^* - \pi, -\theta^*) & \tilde{f}(\lambda^*, -\theta^*) \end{bmatrix} = \begin{bmatrix} p(\lambda^*, \theta^*) & q(\lambda^*, \theta^*) \\ q(\lambda^*, \theta^*) & p(\lambda^*, \theta^*) \end{bmatrix} = \begin{bmatrix} p^* & q^* \\ q^* & p^* \end{bmatrix}, \tag{3.5}$$

where $p(\lambda^*, \theta^*) = p^*$, $q(\lambda^*, \theta^*) = q^*$, and functions p and q are defined in (3.1).

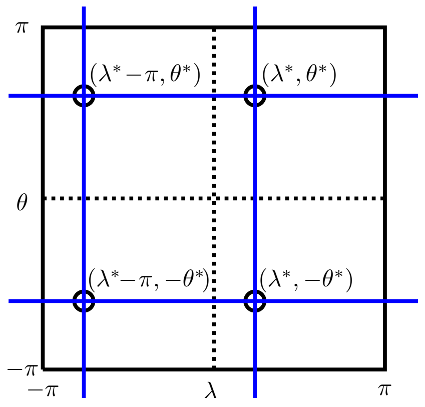


Figure 3.4: The entries of the 2×2 GE pivot matrix (black circles), and the corresponding row and column slices (blue lines) of $\tilde{f}$ containing these values. We only select pivots of this form during the GE procedure.

This choice of M comes from the observation that BMC symmetry is entirely characterized by the following two equalities: $\tilde{f}(\lambda^* - \pi, \theta) = \tilde{f}(\lambda^*, -\theta)$, $\theta \in [-\pi, \pi]$, and $\tilde{f}(\lambda, \theta^*) = \tilde{f}(\lambda - \pi, -\theta^*)$, $\lambda \in [-\pi, \pi]$. To preserve symmetry, a GE step that alters values along the slices $\tilde{f}(\lambda^* - \pi, \theta)$ and $\tilde{f}(\lambda, \theta^*)$ must alter $\tilde{f}(\lambda^*, \theta)$ and $\tilde{f}(\lambda, -\theta^*)$ in an identical way. Figure 3.4 shows that the location of the values of M correspond to the intersections of these row and column slices.

Assuming M in (3.5) is invertible, a GE step with the pivot matrix M is given by

$$\underbrace{\tilde{f}(\lambda, \theta)}_{\tilde{e}(\lambda,\theta)} \quad \longleftarrow \quad \tilde{f}(\lambda, \theta) - \underbrace{\begin{bmatrix} \tilde{f}(\lambda^* - \pi, \theta) & \tilde{f}(\lambda^*, \theta) \end{bmatrix} M^{-1} \begin{bmatrix} \tilde{f}(\lambda, \theta^*) \\ \tilde{f}(\lambda, -\theta^*) \end{bmatrix}}_{\tilde{s}(\lambda,\theta)}. \tag{3.6}$$

This GE step is analogous to the GE step in (2.11). In Lemma 3.1, we show that (3.6) zeros out the row and column slices displayed in Figure 3.4.

Lemma 3.1. *For a BMC function $\tilde{f}$, the GE step* (3.6) *with an invertible pivot matrix M as in* (3.5) *gives a residual $\tilde{e}(\lambda, \theta)$ such that $\tilde{e}(\lambda^*, \theta) = \tilde{e}(\lambda^* - \pi, \theta) = \tilde{e}(\lambda, \theta^*) = \tilde{e}(\lambda, -\theta^*) = 0$.*

Proof. Extending the results from [60, Ch. 3 Theorem 1.4] to the continuous setting, it follows that (3.6) zeros out the column and row slices displayed in Figure 3.4 that are associated with this choice of M. $\square$

We must also show that (3.6) preserves the BMC structure of $\tilde{f}$.

Lemma 3.2. *Given a BMC function $\tilde{f}$, the update $\tilde{s}(\lambda, \theta)$ in* (3.6) *is also a BMC function. That is, the GE step in* (3.6) *preserves BMC structure.*

Proof. To show that $\tilde{s}(\theta, \lambda)$ has BMC structure, we employ *quasimatrices.*[1]

[1]A quasimatrix A of size $[a, b] \times n$ is a matrix with n columns, where each column is a function defined on the interval $[a, b]$ [69].

Let J denote the 2×2 exchange matrix, so that for a matrix $A \in \mathbb{C}^{2\times n}$, JA reverses the rows of A. Let $\mathcal{J}$ be the reflection operator, $\mathcal{J} : \tilde{s}(\lambda, \theta) \to \tilde{s}(\lambda, -\theta)$. Now we use blocks of quasimatrices to rewrite $\tilde{s}$. Using the functions p and q in (3.3), let Q be the $[0, \pi] \times 2$ quasimatrix defined as $Q = \begin{bmatrix} p(\lambda^*, \theta) & | & q(\lambda^*, \theta) \end{bmatrix}$. Let P be the $[0, \pi] \times 2$ quasimatrix defined as $P = \begin{bmatrix} p(\lambda, \theta^*) & | & q(\lambda, \theta^*) \end{bmatrix}$. Then, $\tilde{s}$ in (3.6) can be written as

$$\tilde{s} = \begin{bmatrix} Q \\ \mathcal{J}(QJ) \end{bmatrix} M^{-1} \begin{bmatrix} P^T & JP^T \end{bmatrix}. \tag{3.7}$$

Since M^{-1} is centrosymmetric, it commutes with J. Using this fact, (3.7) becomes

$$\tilde{s} = \begin{bmatrix} QM^{-1}P^T & QM^{-1}JP^T \\ \mathcal{J}(QM^{-1}JP^T) & \mathcal{J}(QM^{-1}P^T) \end{bmatrix}, \tag{3.8}$$

which, by the definition of $\mathcal{J}$, is a BMC function. $\square$

Lemma 3.2 shows that (3.6) can be used to construct a low rank approximation to $\tilde{f}$ that preserves BMC symmetry. However, this relies on M being invertible, which may not always be true. For example, M is singular whenever $\tilde{f}$ is π-periodic in λ. For this reason, we must replace M in (3.6) with $M^{\dagger_\epsilon}$, the ϵ-pseudoinverse of M [30, Sec. 5.5.2], defined below.

Definition 3.4 (ϵ-pseudoinverse)**.** *Let A be an $n \times n$ matrix and $\epsilon \geq 0$. If $A = U\Sigma V^*$ is the singular value decomposition of A with $\Sigma = \operatorname{diag}(\sigma_1, \ldots, \sigma_n)$ and $\sigma_{k+1} \leq \epsilon < \sigma_k$, then the ϵ-pseudoinverse of A is given by*

$$A^{\dagger_\epsilon} = V\Sigma^{\dagger_\epsilon}U^*, \qquad \Sigma^{\dagger_\epsilon} = \operatorname{diag}\left(\sigma_1^{-1}, \ldots, \sigma_k^{-1}, 0, \ldots, 0\right).$$

The matrix $M^{\dagger_\epsilon}$ depends on the singular values of M and a tolerance factor, ϵ. The choice of ϵ is determined by a *coupling* parameter, α, discussed in Section 3.3. The singular values of M are simply

$$\sigma_1(M) = \max\{|p^* + q^*|, |p^* - q^*|\} \text{ and } \sigma_2(M) = \min\{|p^* + q^*|, |p^* - q^*|\}, \quad (3.9)$$

where p^* and q^* are the entries of M as defined in (3.5).

Replacing M^{-1} by $M^{\dagger_\epsilon}$ in (3.6) results in the GE step

$$\tilde{f}(\lambda,\theta) \quad \longleftarrow \quad \tilde{f}(\lambda,\theta) - \begin{bmatrix} \tilde{f}(\lambda^* - \pi, \theta) & \tilde{f}(\lambda^*, \theta) \end{bmatrix} M^{\dagger_\epsilon} \begin{bmatrix} \tilde{f}(\lambda, \theta^*) \\ \tilde{f}(\lambda, -\theta^*) \end{bmatrix}. \quad (3.10)$$

Lemma 3.2 holds for (3.10) because, like M^{-1}, $M^{\dagger_\epsilon}$ is centrosymmetric. If $\sigma_2(M) > \epsilon$, then M is considered well-conditioned and $M^{\dagger_\epsilon} = M^{-1}$. In this case, (3.10) is equivalent to (3.6) and a rank 2 update is achieved by the GE step. However, if M is singular or near-singular, then $M^{\dagger_\epsilon}$ replaces M^{-1} and (3.10) produces a rank 1 update. This is discussed further in Section 3.3, where we view GE defined by (3.10) as a coupled process involving standard GE on two functions related to $\tilde{f}$.

3.2.1.1 Pivot Selection

The strategy used to select each pivot matrix M is important, as it relates to the efficiency and convergence of the GE procedure. The 2×2 analogue to complete pivoting proceeds by choosing $(\lambda^*, \theta^*) \in [0, \pi] \times [0, \pi]$ such that $\sigma_1(M)$ is maximized over all possible choices of M. In practice, however, it is much more efficient to choose (λ^*, θ^*) over a coarse, discrete grid on $[-\pi, \pi] \times [0, \pi]$. This results in a large, but not necessarily maximal, value of $\sigma_1(M)$, and GE is robust to these kinds of compromises [65].

Using (3.10) and accumulating the rank 2 or rank 1 functions used in the GE procedure[2], a low rank approximation to $\tilde{f}$ can be constructed that is of the form given in (2.12).

3.2.2 Preserving Structure for BMC-I Functions (the Sphere)

The above GE procedure preserves general BMC symmetry, but it does not preserve BMC-I structure (see Def. 3.2). Nothing in (3.10) enforces that each rank 1 function constructed by (3.10) is constant along the lines $\tilde{f}(\lambda, 0)$ and $\tilde{f}(\lambda, \pm\pi)$. However, in the case where $\tilde{f}(\lambda, 0) = \tilde{f}(\lambda, \pm\pi) = 0$, BMC-I structure is preserved in each of the rank 1 terms since each GE step preserves the zero row and column slices of the function it is acting on. This suggests a more general strategy. If $\tilde{f}$ is nonzero along either $\theta = 0$ or $\theta = \pm\pi$, then $\tilde{f}$ is constant along the row slices $\tilde{f}(\lambda, 0)$ and $\tilde{f}(\lambda, \pm\pi)$ for $\lambda \in [-\pi, \pi]$. Choosing the first GE step as

$$\tilde{f}(\lambda, \theta) \quad \longleftarrow \quad \tilde{f}(\lambda, \theta) - \tilde{f}(\lambda^*, \theta) \tag{3.11}$$

will zero out any row slices of $\tilde{f}$ that are constant in λ. After this initial step, subsequent updates to $\tilde{f}$ are always zero along $\theta = 0$ and $\theta = \pm\pi$, so BMC-I structure is always preserved.

A continuous idealization of the BMC-preserving GE process is shown in Figure 3.5. In practice, the GE procedure is implemented in two phases, and this process is similar to the method described in [67], except with 2×2 pivots and a pivot selection method based on maximizing the largest singular value of the pivot matrix. The result is a low rank approximation to $\tilde{f}$ that can be formulated as in (2.12) (see Section 3.4), i.e.,

[2]A rank 2 update can be written as the sum of two rank 1 updates (see Section 3.4).

```
Algorithm: Structure-preserving GE on BMC functions
Input: A BMC function $\tilde{f}$ and a coupling parameter $0 \le \alpha \le 1$.
Output: A structure-preserving low rank approximation $\tilde{f}_k$ to $\tilde{f}$.

Set $\tilde{f}_0 = 0$ and $\tilde{e}_0 = \tilde{f}$.
for $k = 1, 2, 3, \ldots,$
    Find $(\lambda_k, \theta_k)$ such that $M = \begin{bmatrix} p^* & q^* \\ q^* & p^* \end{bmatrix}$, where $p^* = \tilde{e}_{k-1}(\lambda_{k-1} - \pi, \theta_{k-1})$ and $q^* = \tilde{e}_{k-1}(\lambda_{k-1}, \theta_{k-1})$, has maximal $\sigma_1(M)$ (see (3.9)).
    Set $\epsilon = \alpha\sigma_1(M)$.
    $\tilde{e}_k = \tilde{e}_{k-1} - \begin{bmatrix} \tilde{e}_{k-1}(\lambda_k - \pi, \theta) & \tilde{e}_{k-1}(\lambda_k, \theta) \end{bmatrix} M^{\dagger_\epsilon} \begin{bmatrix} \tilde{e}_{k-1}(\lambda, \theta_k) \\ \tilde{e}_{k-1}(\lambda, -\theta_k) \end{bmatrix}.$
    $\tilde{f}_k = \tilde{f}_{k-1} - \begin{bmatrix} \tilde{e}_{k-1}(\lambda_k - \pi, \theta) & \tilde{e}_{k-1}(\lambda_k, \theta) \end{bmatrix} M^{\dagger_\epsilon} \begin{bmatrix} \tilde{e}_{k-1}(\lambda, \theta_k) \\ \tilde{e}_{k-1}(\lambda, -\theta_k) \end{bmatrix}.$
end
```

Figure 3.5: A continuous idealization of our structure-preserving GE procedure on BMC functions. In practice, we use a discretization of this procedure and terminate it after a finite number of steps.

$$\tilde{f}(\lambda, \theta) \approx \sum_{j=1}^{K} d_j c_j(\theta) r_j(\lambda), \qquad (\lambda, \theta) \in [-\pi, \pi]^2. \tag{3.12}$$

We represent each of the $c_j(\theta)$ and $r_j(\lambda)$ functions in (3.12) using a Fourier expansion. This process, including the retrieval of the Fourier coefficients for each $c_j(\theta)$ and $r_j(\lambda)$, is achieved in $\mathcal{O}(K^3 + K^2(m+n) + K(m \log m + n \log n))$ operations (see [67]), where K is the numerical rank of the function, and m and n are the maximum Fourier modes required to resolve the functions $c_j(\theta)$ and $r_j(\lambda)$, respectively, to approximately machine precision.

The example in Figure 3.6 illustrates the form of approximation given by (3.12). Each $c_j(\theta)$ and $r_j(\lambda)$ define a longitudinal and latitudinal "slice" of $\tilde{f}$, respectively. Together, these slices form a sparse collection of samples from the full tensor product grid that have been adaptively selected to approximate $\tilde{f}$ to machine epsilon. This collection is referred to as the "skeleton" of $\tilde{f}$. As illustrated in Figure 3.6, this method does not oversample the function near the poles.

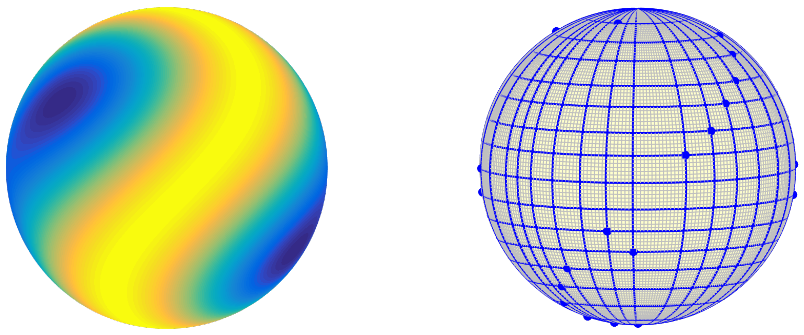


Figure 3.6: Left: The function $f(x, y, z) = \cos(xz - \sin y)$ on the unit sphere. Right: The "skeleton" used to approximate f that is found via the BMC structure-preserving GE procedure. The blue dots are the entries of the 2×2 pivot matrices used by GE. The GE procedure only samples f along the blue lines. The underlying tensor grid (in gray) shows the sampling grid required without low rank techniques, which cluster near the poles.

3.2.3 Preserving Structure for BMC-II Functions (the Disk)

A function g on the unit disk can be associated with the BMC-II function $\tilde{g}(\theta, \rho)$ via (3.4). Since the GE step (3.10) preserves *general* BMC symmetry, it is immediately applicable to $\tilde{g}$. However, BMC-II symmetry requires that $\tilde{g}$ is constant along $\rho = 0$, and this feature of $\tilde{g}$ is not preserved by (3.10). BMC-II symmetry is preserved whenever $\tilde{g}(\theta, 0) = 0$, so if $\tilde{g}(\theta, 0) \neq 0$, then we choose the first GE step as

$$\tilde{g}(\theta, \rho) \quad \longleftarrow \quad \tilde{g}(\theta, \rho) - \tilde{g}(\theta^*, \rho), \tag{3.13}$$

where $\theta^* \in [-\pi, \pi]$. After this initial step is applied, the row slice $\tilde{g}(\theta, 0)$ is zeroed out and subsequently, every update function in (3.10) has BMC-II symmetry. Note that this technique is identical to the method applied for preserving BMC-I structure.

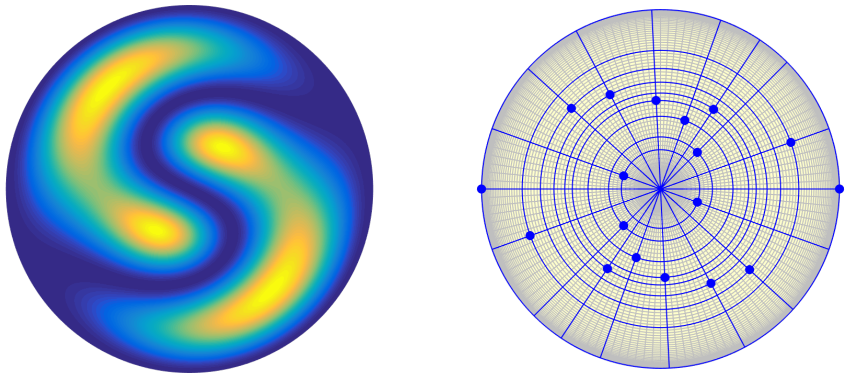


Figure 3.7: Left: The function $g(\theta, \rho) = -\cos((\sin(\pi\rho)\cos(\theta) + \sin(2\pi\rho)\sin(\theta))/4)$ on the unit disk. Right: The adaptively selected skeleton for $\tilde{g}$. The blue dots are the pivot locations selected via GE. The GE procedure only samples g along the blue lines. The underlying tensor product grid (in gray) shows the sample points required to approximate g to approximately machine precision without the GE procedure applied to the DFS method. The oversampling of the tensor grid, in contrast to the low rank skeleton, can be seen.

The same GE procedure described for BMC-I functions is therefore applicable to BMC-II functions, but the implementational details and the subsequent algorithms used for computations involving the approximants are different. This is because BMC-II functions associated with the disk are not periodic in ρ. Selecting the adaptive skeleton for $\tilde{g}$ via the GE procedure, a low rank approximation to $\tilde{g}$ of the form in (2.12) is given by

$$\tilde{g}(\theta, \rho) \approx \sum_{j=1}^{K} d_j c_j(\rho) r_j(\theta), \qquad (\theta, \rho) \in [-\pi, \pi] \times [-1, 1]. \tag{3.14}$$

Here, a Fourier expansion is used to represent each of the $r_j(\theta)$ functions in (3.14), and a Chebyshev expansion is used for each $c_j(\rho)$ function. Figure 3.7 displays the skeleton constructed for a function on the disk. Each $c_j(\rho)$ function forms a radial slice, and each $r_j(\theta)$ forms a circular slice. One can see in Figure 3.7 that unlike a tensor product representation, the approximant constructed via GE does not oversample the function near the origin.

3.3 Structure-preserving Gaussian Elimination as a Coupled Procedure

An alternative interpretation of structure-preserving GE views the process as a coupled procedure involving two standard GE algorithms. This interpretation can be used to state the convergence properties of the GE procedure (see Section 3.5), and it also reveals that the approximants constructed via BMC structure-preserving GE possess important parity properties inherent to functions in spherical and polar geometries (see Section 3.4). For the sake of exposition, we consider the BMC-I function $\tilde{f}$ defined in (3.1) and associated with the sphere throughout this section. However, these results have equivalent interpretations for BMC-II functions associated with the disk.

Observe that $\tilde{f}$ in (3.1) can be decomposed into a sum of two BMC functions: let $f^+ = p + q$ and $f^- = p - q$, where p and q are defined in (3.1). Then,

$$\tilde{f} = \underbrace{\frac{1}{2}\begin{bmatrix} f^+ & f^+ \\ \texttt{flip}(f^+) & \texttt{flip}(f^+) \end{bmatrix}}_{=\tilde{f}^+} + \underbrace{\frac{1}{2}\begin{bmatrix} f^- & -f^- \\ -\texttt{flip}(f^-) & \texttt{flip}(f^-) \end{bmatrix}}_{=\tilde{f}^-}. \tag{3.15}$$

Consider the action of the GE step given in (3.10). Let M be the first 2×2 pivot matrix defined in (3.5). Then, $M^{\dagger_\epsilon}$ can be expressed as

$$M^{\dagger_\epsilon} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} m^+ & \\ & m^- \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}, \tag{3.16}$$

where the possible values of m^+ and m^- are given by

$$(m^+, m^-) = \begin{cases} (1/(p^* + q^*), 0), & \text{if } |p^* - q^*| < \alpha|p^* + q^*|, \\ (0, 1/(p^* - q^*)), & \text{if } |p^* + q^*| < \alpha|p^* - q^*|, \\ (1/(p^* + q^*), 1/(p^* - q^*)), & \text{otherwise.} \end{cases} \tag{3.17}$$

Here, $\alpha = \epsilon/\sigma_1(M) = \epsilon/\max\{|p^* + q^*|, |p^* - q^*|\}$. We can use (3.16) to re-write the GE step in (3.10) as

$$\begin{aligned} \tilde{f}(\lambda, \theta) \longleftarrow \tilde{f}(\lambda, \theta) &- \frac{m^+}{2}\left(\tilde{f}(\lambda^* - \pi, \theta) + \tilde{f}(\lambda^*, \theta)\right)\left(\tilde{f}(\lambda, \theta^*) + \tilde{f}(\lambda, -\theta^*)\right) \\ &- \frac{m^-}{2}\left(\tilde{f}(\lambda^* - \pi, \theta) - \tilde{f}(\lambda^*, \theta)\right)\left(\tilde{f}(\lambda, \theta^*) - \tilde{f}(\lambda, -\theta^*)\right). \end{aligned} \tag{3.18}$$

Using the definitions of $\tilde{f}^+$ and $\tilde{f}^-$, (3.18) can then be written as

$$\begin{aligned} \tilde{f}(\lambda, \theta) \quad \longleftarrow \quad & \frac{1}{2}(\tilde{f}^+(\lambda, \theta) - m^+\tilde{f}^+(\lambda^*, \theta)\tilde{f}^+(\lambda, \theta^*)) \ + \\ & \frac{1}{2}(\tilde{f}^-(\lambda, \theta) - m^-\tilde{f}^-(\lambda^*, \theta)\tilde{f}^-(\lambda, \theta^*)), \end{aligned} \tag{3.19}$$

which suggests that the step is equivalent to two coupled GE steps on the functions $\tilde{f}^+$ and $\tilde{f}^-$. This is verified by noting that $p^* + q^* = \tilde{f}^+(\lambda^*, \theta^*)$ and $p^* - q^* = \tilde{f}^-(\lambda^*, \theta^*)$ in the definition of m^+ and m^- in the third case of (3.17).

The coupling of the two GE steps is through the parameter α in (3.17); for this reason, we call α the *coupling parameter*. If either of the first two cases of (3.17) is chosen, then GE with complete pivoting is performed on only one term in (3.19), resulting in a rank 1 update to $\tilde{f}$. In the third case of (3.17), $M^{\dagger_\epsilon} = M^{-1}$, and a rank 2 update is achieved. Performing as many rank 2 updates as possible reduces the overall number of pivot searches required by the GE procedure. For this reason, the choice of α is important. Too small a value of α may allow the use of M^{-1} when M is ill–conditioned, but choosing α too close to 1 hampers the efficiency of the procedure. In practice, we choose $\alpha = 1/100$.

One may wonder if it makes sense to choose $\alpha = 1$, so the GE steps in (3.19) on $\tilde{f}^+$ and $\tilde{f}^-$ are fully decoupled. Then, the structure-preserving GE procedure is equivalent to applying GE with complete pivoting to $\tilde{f}^{+}$ and $\tilde{f}^{-}$ independently. This is problematic because the rank 1 terms attained from applying GE independently to $\tilde{f}^{+}$ and $\tilde{f}^{-}$ cannot then be properly ordered when constructing a low rank approximant of $\tilde{f}$. By selecting $0 < \alpha < 1$, the GE steps are coupled, the rank 1 terms are selected in an ordered way, and a single GE step can achieve a rank 2 update, which improves efficiency by reducing the number of required pivot searches.

The decomposition of $\tilde{f}$ in (3.19) is useful in a variety of contexts. In the next section, we show how it can be used to explicitly express parity properties possessed by continuous functions on the sphere and disk.

3.4 Parity Properties for BMC Approximants

Functions defined in polar and spherical geometries possess parity properties that can be used in approximation schemes to satisfy pole conditions [10, 57, 82]. Typically, these properties are used to enforce symmetry on the 2D Fourier or 2D Fourier–Chebyshev coefficients of functions $\tilde{f}$ in (3.1) or $\tilde{g}$ in (3.4), respectively. Using the more general concept of BMC symmetry, we can enforce these properties directly on the *values* of $\tilde{f}$ and $\tilde{g}$ instead of the coefficients, and this is the premise BMC structure-preserving GE operates on. In this section, we show that low rank approximations to $\tilde{f}$ and $\tilde{g}$ constructed via BMC structure-preserving GE (see Figure 3.5) satisfy the parity properties associated with their Fourier expansions. These results can be used to simplify algorithmic procedures, such as integration (see Sections 4.2.3 and 4.3.3), and the resulting expressions also clarify why our approximants are stable for differentiation (see Sections 4.2.4 and 4.3.4).

Functions on the Sphere

Consider the truncated Fourier expansion of a BMC-I function $\tilde{f}$ derived from the sphere via (3.1), written as

$$\tilde{f}(\lambda,\theta) \approx \sum_{k=-n/2}^{n/2-1} \psi_k(\theta)e^{ik\lambda} \qquad (\lambda,\theta)\in[-\pi,\pi]^2, \tag{3.20}$$

and note that $\psi_k(\theta)$ can also be expanded as Fourier series. In [82] and [49], it is shown that the following properties must hold for any approximation to $\tilde{f}$ to be continuous and differentiable on the sphere:

(i) k is even $\implies \psi_k(\theta)$ is an even function (i.e., $\psi_k(\theta)$ has only cosine modes),

(ii) k is odd $\implies \psi_k(\theta)$ is an odd function (i.e., $\psi_k(\theta)$ has only sine modes),

(iii) $k \neq 0 \implies \psi_k(0) = \psi_k(\pm\pi) = 0$.

We now show that these properties are inherent to approximants constructed via structure-preserving GE. The function $\tilde{f}$ can be written as the sum of two BMC functions via (3.15), i.e. $\tilde{f} = \frac{1}{2}f^{+} + \frac{1}{2}f^{-}$. Using (3.19), the low rank approximant to $\tilde{f}$ constructed by the GE procedure can be written as

$$\tilde{f}(\lambda,\theta) \approx \sum_{j=1}^{K} d_j c_j(\theta) r_j(\lambda) = \sum_{j=1}^{K^{+}} d_j^{+} c_j^{+}(\theta) r_j^{+}(\lambda) + \sum_{j=1}^{K^{-}} d_j^{-} c_j^{-}(\theta) r_j^{-}(\lambda), \tag{3.21}$$

where $K^{+}+K^{-}=K$, and $\{d_j\}_{j=1}^{K}$ are determined by the eigenvalues of the pseudoinverse of the pivot matrices. It is evident in (3.15) that $\tilde{f}^{+}$ is an even function in θ and is π-periodic in λ, and that $\tilde{f}^{-}$ is an odd function in θ and π-antiperiodic in λ[3]. It follows that the functions $c_j^{+}(\theta)$ and $r_j^{+}(\lambda)$ for $1 \le j \le K^{+}$ are even and π-periodic, respectively, while $c_j^{-}(\theta)$ and $r_j^{-}(\lambda)$ for $1 \le j \le K^{-}$ are odd and π-antiperiodic,

[3]A function f is said to be π-antiperiodic if $f(\xi+\pi) = -f(\xi)$.

respectively. This implies that the Fourier expansions of each of these terms will naturally satisfy parity properties (i) and (ii), since π-periodic functions have only even Fourier modes, and π-antiperiodic functions have only odd Fourier modes.

If $\tilde{f}$ is non-zero at the poles and (3.11) is applied in the first step of the GE procedure, then $c_1^+(\theta) = \tilde{f}(\lambda^*, \theta)$, $r_1^+(\lambda) = 1$, and $d_1^+ = 1$. This ensures that for $2 \leq j \leq K^+$, $c_j^+(0) = c^+(\pm\pi) = 0$, enforcing that parity property (iii) holds for the approximant.

Functions on the Disk

Functions on the disk possess similar parity properties. For $\tilde{g}$ as in (3.4), the Fourier expansion of $\tilde{g}$ given in (2.16) must satisfy the following properties:

i k is even $\implies$ $\phi_k(\rho)$ is an even function,

ii k is odd $\implies$ $\phi_k(\rho)$ is odd function,

iii $k \neq 0 \implies \phi_k(0) = 0$.

Applying the results from Section 3.3, $\tilde{g}$ can be written as the sum of two BMC functions, so that $\tilde{g} = \tilde{g}^+ + \tilde{g}^-$, where $\tilde{g}^+$ is an even function in ρ and π-periodic in θ. Likewise, $\tilde{g}^-$ is an odd function in ρ and π-antiperiodic in θ, implying that parity properties (i) and (ii) are inherent to (3.15) applied to $\tilde{g}$. Furthermore, the low rank approximation to $\tilde{g}$ can be re-written as

$$\tilde{g}(\theta, \rho) \approx \sum_{j=1}^{K} d_j c_j(\rho) r_j(\theta) = \sum_{j=1}^{K^+} d_j^+ c_j^+(\rho) r_j^+(\theta) + \sum_{j=1}^{K^-} d_j^- c_j^-(\rho) r_j^-(\theta). \tag{3.22}$$

If $\tilde{g}$ is non-zero along $\tilde{g}(\theta, 0)$, the first step of the GE procedure is given by (3.11). This chooses $c_1^+(\rho) = \tilde{g}(\theta^*, \rho)$, $r_1^+(\theta) = 1$, and $d_1^+ = 1$, so that for $2 \leq j \leq K^+$,

$c_j(0) = 0$. In this way, parity property (iii) is preserved in the decomposition given by (3.22).

Note that in a more generalized setting, a BMC function $\tilde{f}(\xi, \eta)$ that is not necessarily associated with the sphere or disk will have the property that $\tilde{f}^+$ is even in η and symmetric in ξ, and $\tilde{f}^-$ is odd in η and anti-symmetric in ξ.

3.5 Convergence and Recovery Properties

In this section, we present several theorems describing the convergence behavior of approximants constructed via the structure-preserving GE procedure. Since these results apply to BMC functions associated with either the sphere or the disk, we consider a general BMC function $\tilde{f}(\xi, \eta)$ with $(\xi, \eta) \in [-a, a] \times [-b, b]$. Our first result is for $\tilde{f}$ that are of finite rank K.

Theorem 3.1 (Exact recovery for functions of finite rank)**.** *If $\tilde{f} : [-a, a] \times [-b, b] \to \mathbb{C}$ is a rank K BMC function, then the structure-preserving GE procedure exactly recovers $\tilde{f}$ in K or less steps.*

Proof. Let (ξ^*, η^*) be the pivot locations selected for the first GE step, and let $M^{\dagger_\epsilon}$ be the corresponding 2×2 pivot matrix of rank k ($k = 1$ or $k = 2$). By the generalized Guttman rank additivity formula [15],

$$\operatorname{rank}(\tilde{f}) = \operatorname{rank}(M^{\dagger_\epsilon}) + \operatorname{rank}\left(\tilde{f} - \begin{bmatrix} \tilde{f}(\xi^* - a, \cdot) & \tilde{f}(\xi^*, \cdot) \end{bmatrix} M^{\dagger_\epsilon} \begin{bmatrix} \tilde{f}(\cdot, \eta^*) \\ \tilde{f}(\cdot, -\eta^*) \end{bmatrix}\right),$$

where rank$(\cdot)$ denotes the rank of a function or matrix. Thus, the rank of the residual produced by each GE step is rank$(\tilde{f})$-k. After at most K steps, the residual is of rank 0, and this can only be true if the residual is the zero function. The GE procedure terminates at this point, so a rank K function has been constructed, and it is exactly equal to $\tilde{f}$. $\square$

On the sphere, a band-limited function is a function that can be expressed as a finite series of spherical harmonics. Each spherical harmonic is a rank 1 function [47, Sec. 14.30], so Theorem 3.1 implies that the GE procedure exactly recovers band-limited functions on the sphere in finitely many steps.

A continuous, band-limited function on the disk is any function such that its Fourier transform is compactly supported. Since its Fourier expansion is finite, the function is of finite rank. It follows that after a finite number of steps, the GE procedure exactly recovers continuous, band-limited functions on the disk.

What can we say about convergence if $\tilde{f}$ is of infinite rank? If $\tilde{f}$ is sufficiently analytic, then it can be shown that the approximants to $\tilde{f}$ constructed via structure-preserving GE converge to $\tilde{f}$ at a geometric rate. Proving this requires an analysis of the *intermediate growth factor* associated with the GE procedure:

Definition 3.5 (Intermediate growth factor)**.** *For any given GE strategy performed on a bivariate function $\tilde{f}(\xi,\eta)$, the intermediate growth factor $\gamma_k(\tilde{f})$ is defined as the ratio between the absolute maximum of $\tilde{f}$ and the absolute maximum of the residual obtained after a rank k update. Formally,*

$$\gamma_k(\tilde{f}) = \frac{\max_{\xi,\eta} |\tilde{e}_k(\xi,\eta)|}{\max |\tilde{f}(\xi,\eta)|},$$

where $\tilde{e}_k$ is the residual after k steps of the GE procedure.

In the following lemma, an upper bound is given for the intermediate growth factor after a single step of the structure–preserving GE procedure.

Lemma 3.3. *Let $\tilde{f}(\xi,\eta)$, $(\xi,\eta) \in [-a,a]\times[-b,b]$, be a BMC function. After a single step, the intermediate growth factor for the structure–preserving GE procedure satisfies $\gamma_1(\tilde{f}) \leq \max\{3, \sqrt{1+4/\alpha}\}$, where α is the coupling parameter in* (3.17).

Proof. Consider the first GE step. Let M be the selected 2×2 pivot matrix as in (3.5) that maximizes the choice of $\sigma_1(M)$ in (3.9). Observe that $\sigma_1(M) \geq ||\tilde{f}||_\infty$, where $||\tilde{f}||_\infty$ is the absolute maximum of $\tilde{f}$ on $[-a,a]\times[-b,b]$. There are two cases to consider:

Case 1: $\sigma_2(M) < \alpha\sigma_1(M)$. Here $M^{\dagger_\epsilon}$ in (3.10) with $\epsilon = \alpha\sigma_1(M)$ is of rank 1. Using the spectral decomposition of $M^{\dagger_\epsilon}$ in (3.16) and the fact that $\sigma_1(M) \geq \|\tilde{f}\|_\infty$, it follows that

$$\left\| \tilde{f} - \begin{bmatrix} \tilde{f}(\xi^*-a,\cdot) & \tilde{f}(\xi^*,\cdot) \end{bmatrix} M^{\dagger_\epsilon} \begin{bmatrix} \tilde{f}(\cdot,\eta^*) \\ \tilde{f}(\cdot,-\eta^*) \end{bmatrix} \right\|_\infty \leq \|\tilde{f}\|_\infty + \frac{2\|\tilde{f}\|_\infty^2}{\sigma_1(M)} \leq 3\|\tilde{f}\|_\infty.$$

Thus, the growth factor in this case is at most 3.

Case 2: $\sigma_2(M) \geq \alpha\sigma_1(M)$. Here, $M^{\dagger_\epsilon} = M^{-1}$. Let $\|M^{-1}\|_{\max}$ denote the maximum absolute entry of M^{-1}. Then,

$$\|M^{-1}\|_{\max} \leq \frac{\|\tilde{f}\|_\infty}{\det(M)} = \frac{\|\tilde{f}\|_\infty}{\sigma_1(M)\sigma_2(M)} \leq \frac{\|\tilde{f}\|_\infty}{\alpha\sigma_1(M)^2} \leq \frac{1}{\alpha\|\tilde{f}\|_\infty}.$$

It follows that

$$\left\| \tilde{f} - \begin{bmatrix} \tilde{f}(\xi^*-a,\cdot) & \tilde{f}(\xi^*,\cdot) \end{bmatrix} M^{\dagger_\epsilon} \begin{bmatrix} \tilde{f}(\cdot,\eta^*) \\ \tilde{f}(\cdot,-\eta^*) \end{bmatrix} \right\|_\infty \leq \|\tilde{f}\|_\infty + \frac{4\|\tilde{f}\|_\infty}{\alpha} \leq \left(1+\frac{4}{\alpha}\right)\|\tilde{f}\|_\infty.$$

In this case, the growth factor is $\leq \sqrt{1+4/\alpha}$ because the GE update is of rank 2. □

Since the intermediate growth factor is bounded for each step of GE, convergence of the GE procedure can be proven for functions $\tilde{f}$ that satisfy the following property: For any $\eta^* \in [-b,b]$, $\tilde{f}(\cdot,\eta^*)$ is an analytic function in a sufficiently large region of the complex plane containing $[-a, a]$. We formalize this idea using the concept of a *stadium.*

Definition 3.6 (Stadium)**.** *The stadium S_β with radius $\beta > 0$ is the region in the complex plane consisting of all numbers lying at a distance $\leq \beta$ from an interval $[c, d]$, i.e.,*

$$S_\beta = \left\{ z \in \mathbb{C} : \inf_{x \in [c,d]} |x - z| \leq \beta \right\}.$$

We now prove that for sufficiently analytic functions, the GE procedure converges at a geometric rate. In the following theorem, the roles of ξ and η can be exchanged.

Theorem 3.2. *Let $\tilde{f} : [-a, a] \times [-b, b] \to \mathbb{C}$ be a BMC function such that $\tilde{f}(\xi, \cdot)$ is continuous for any $\xi \in [-a, a]$ and $\tilde{f}(\cdot, \eta)$ is analytic and uniformly bounded in a stadium S_β of radius $\beta = \max(3, 1 + \alpha^{-1})\kappa a$, $\kappa > 1$, for any $\eta \in [-b, b]$, where α is the coupling parameter described in* (3.17)*. Then, there exists a constant $c > 0$ such that*

$$||\tilde{f} - \tilde{f}_k||_\infty = ||\tilde{e}_k||_\infty \leq c\mu^{-k},$$

where $\mu = \min\{\kappa, \alpha^{-1}\}$ and $\tilde{f}_k$ is the approximant constructed after k steps of the structure-preserving GE procedure. In other words, the GE procedure constructs rank k approximants that converge to $\tilde{f}$ at a geometric rate.

Proof. Since $\tilde{e}_k$ is a BMC function for $k \geq 0$, $\tilde{e}_k$ can be decomposed into the sum of an even-symmetric and odd-antisymmetric function, i.e., $\tilde{e}_k = \tilde{e}_k^+ + \tilde{e}_k^-$, as discussed in Section 3.4. From Section 3.3, we know that the structure-preserving GE procedure in Figure 3.5 can be regarded as two coupled GE procedures on the even-symmetric and odd-antisymmetric parts of $\tilde{e}_k$.

Let $\mu = \min\{\kappa, \alpha^{-1}\}$, where $0 < \alpha < 1$ is the coupling parameter described in (3.17), and $\kappa > 1$. Choose c so that $||\tilde{e}_0^+||_\infty \leq c/2$ and $||\tilde{e}_0^-||_\infty \leq c/2$. We will show that for all k, $||e_k||_\infty \leq c\mu^{-k}$.

We proceed by induction. For $k = 0$, observe that $\max\{||\tilde{e}_0^+||_\infty, ||\tilde{e}_0^-||_\infty\} \leq (c/2)\mu^0$. Now suppose that at step k, the following induction hypothesis holds:

$$\max\{||\tilde{e}_k^+||_\infty, ||\tilde{e}_k^-||_\infty\} \leq (c/2)\mu^{-k}.$$

Consider the next structure-preserving GE step. This step occurs in one of three cases described in (3.17). In each case, we will show that the induction hypothesis continues to hold at the $k+1$ step.

Case 1: In this case, $||\tilde{e}_k^-||_\infty < \alpha||\tilde{e}_k^+||_\infty$, and only $\tilde{e}_k^+$ is updated in the $k+1$ step of structure-preserving GE (see Section 3.3). This step is therefore equivalent to performing a standard GE step as in (2.11) on $\tilde{e}_k^+$, and letting $\tilde{e}_{k+1}^- = \tilde{e}_k^-$. In the proof of Theorem 8.1 in [69], it is shown that after applying (2.11) to $\tilde{e}_k^+$, which is a function that in at least one variable is analytically continuable to the region S_β, it must be true that

$$||\tilde{e}_{k+1}^+||_\infty \leq \kappa^{-1}||\tilde{e}_k^+||_\infty.$$

It is also true that

$$||\tilde{e}_{k+1}^-||_\infty = ||\tilde{e}_k^-||_\infty < \alpha||\tilde{e}_k^+||_\infty.$$

Using the definition of μ, it follows that

$$\max\{||\tilde{e}_{k+1}^+||_\infty, ||\tilde{e}_{k+1}^-||_\infty\} \leq \mu^{-1}\max\{||\tilde{e}_k^+||_\infty, ||\tilde{e}_k^-||_\infty\}.$$

Applying the induction hypothesis, we find that

$$\max\{||\tilde{e}_{k+1}^+||_\infty, ||\tilde{e}_{k+1}^-||_\infty\} \leq (c/2)\mu^{-(k+1)}.$$

<u>Case 2</u>: In this case, $||\tilde{e}_k^+||_\infty < \alpha||\tilde{e}_k^-||_\infty$, and only $\tilde{e}_k^-$ is updated by the $k+1$ structure-preserving GE step. Applying the same reasoning used in Case 1, we conclude that

$$\max\{||\tilde{e}_{k+1}^+||_\infty, ||\tilde{e}_{k+1}^-||_\infty\} \leq (c/2)\mu^{-(k+1)}.$$

<u>Case 3</u>: In this case, $M^{\dagger_\epsilon} = M^{-1}$, so the $k+1$ BMC structure-preserving GE step is of the form given in (3.6). As shown in Section 3.3, this GE step is equivalent to performing coupled GE steps with scalar pivots $1/m^+$ and $1/m^-$ from (3.17) on $\tilde{e}^+$ and $\tilde{e}^-$, respectively. Without loss of generality, suppose that $|m^-| > |m^+|$. Then, a standard GE step as in (2.11) is applied to $\tilde{e}_k^+$, and a GE step with a nonstandard pivoting strategy is performed on $\tilde{e}_k^-$. The only difference between these GE steps is that in the latter case, the selected pivot may not be the maximum value of $\tilde{e}_k^-$.

For $\tilde{e}_k^+$, we apply Theorem 8.1 in [69], finding that

$$||\tilde{e}_{k+1}^+||_\infty \leq \kappa^{-1}||\tilde{e}_k^+||_\infty.$$

Theorem 8.1 in [69] describes convergence for GE with *complete* pivoting, but it is not difficult to see that an analogous result holds for any GE scheme with an intermediate growth factor that is appropriately bounded at each step. We observe that Case 2 of Lemma 3.3 gives a bound of $\sqrt{1+4/\alpha}$ for the intermediate growth factor associated with the coupled, structure-preserving GE step, but we require a bound on the intermediate growth factor for the individual, nonstandard GE step being applied to $\tilde{e}_k^-$ in (3.19) .

In this instance, the pivot is $1/m^-$. Using the fact that $1/|m^-| = \sigma_2(M)$, and $\sigma_2(M) \geq \alpha\sigma_1(M) > \alpha||\tilde{e}_k^-||_\infty$, it is straightforward to show that the intermediate growth factor for this nonstandard GE step on $\tilde{e}_k^-$ is bounded by $(1+\alpha^{-1})$.

We use this bound in the same argument as is applied in the proof of Theorem 8.1 in [69] to conclude that

$$||\tilde{e}^{-}_{k+1}||_\infty \leq \kappa^{-1}||\tilde{e}^{-}_{k}||_\infty.$$

It follows from the definition of μ and the induction hypothesis that

$$\max\{||\tilde{e}^{+}_{k+1}||_\infty, ||\tilde{e}^{-}_{k+1}||_\infty\} \leq (c/2)\mu^{-(k+1)}.$$

We have shown that the induction hypothesis continues to hold in each case for the $k+1$ step. By induction, for all $k \geq 0$,

$$\max\{||\tilde{e}^{+}_{k}||_\infty, ||\tilde{e}^{-}_{k}||_\infty\} \leq (c/2)\mu^{-k}.$$

Applying the triangle inequality, i.e. $||\tilde{e}_k||_\infty \leq ||\tilde{e}^{+}_{k}||_\infty + ||\tilde{e}^{-}_{k}||_\infty$, it follows that for all $k \geq 0$, $||\tilde{e}_k||_\infty \leq c\mu^{-k}$. $\square$

While convergence can be proven for highly analytic functions, we observe in practice that convergence occurs even for functions that are only a few times differentiable, and it occurs at rates that are near-optimal. We discuss the meaning of this statement in the next section.

3.6 Near-optimality

If a function $\tilde{f}(\xi, \eta)$ is Lipschitz continuous with respect to both variables for $(\xi, \eta) \in [-a, a] \times [-b, b]$, then the best rank K approximation to $\tilde{f}$ in the standard L_2 norm is given by the Karhunen–Loève expansion, also known as the singular value decomposition (SVD) of $\tilde{f}$ (see Section 2.2.1).

In Appendix A.2, it is shown that the SVD preserves BMC structure[4]. Unfortunately, the high cost of computing the SVD makes this an untenable approach for constructing low rank approximants to BMC functions. Nonetheless, the SVD gives

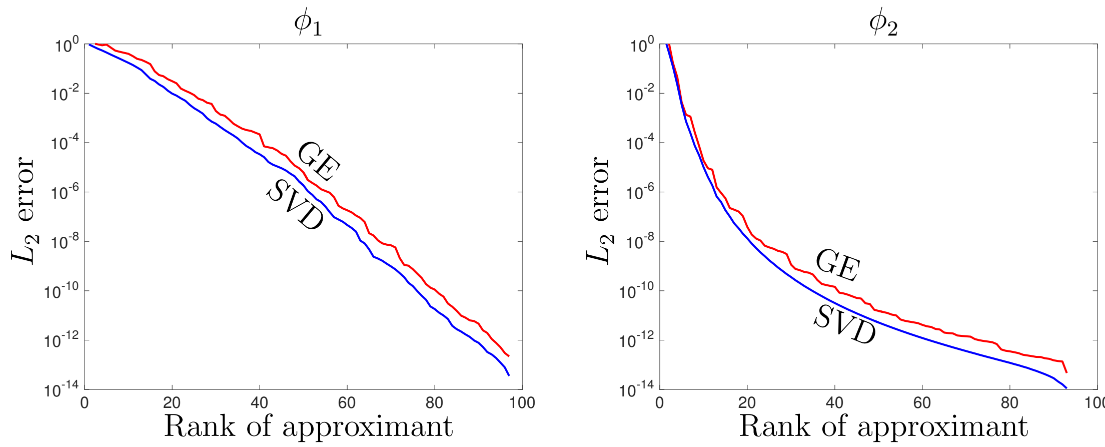


Figure 3.8: A comparison of low rank approximations to the functions in (3.23) computed using the SVD and the iterative GE procedure. The L_2 error is plotted against the rank of the approximants to ϕ_1 and ϕ_2. The L_2 error given by the SVD approximants are optimal and we observe that that the low rank approximants constructed by the GE procedure are near-optimal.

optimal low rank approximants and therefore provides a way to check the quality of the low rank approximants constructed by our GE procedure.

As an example, we examine the following two BMC-II functions defined on the disk:

$$\begin{aligned} \phi_1(\theta,\rho) &= e^{-(\cos(11\rho\sin\theta)+\sin(\rho\cos\theta))^2}, \\ \phi_2(\theta,\rho) &= (1-\omega(\theta,\rho))_+^6\left(35(\omega(\theta,\rho))^2+18\omega(\theta,\rho)+3\right), \end{aligned} \tag{3.23}$$

where $\omega(\theta,\rho)=\sqrt{(\rho\cos\theta-1/5)^2+(\rho\sin\theta-1/5)^2}$ and $\zeta_+=\max\{\zeta,0\}$. Figure 3.8 displays the L_2 error over $[-\pi,\pi]\times[-1,1]$ for rank K approximations constructed via the SVD and the GE procedure. The error given by the SVD behaves in accordance

[4]The SVD preserves *general* BMC structure, but it does not necessarily preserve BMC-I or BMC-II structure.

with known theoretical results, decaying geometrically for the function ϕ_1 and at an algebraic rate for ϕ_2 [64]. In practice, it is observed that the GE procedure developed in this chapter efficiently and reliably constructs near-best low rank approximants to BMC functions that have some degree of smoothness.

We have shown that the BMC structure-preserving GE procedure is an efficient, near-optimal method for constructing approximants to functions in polar and spherical geometries. In the next chapter, we develop algorithms for computing with these approximants.

CHAPTER 4

NUMERICAL COMPUTATION IN SPHERICAL AND POLAR GEOMETRIES

4.1 Software

Chapter 3 describes an effective GE procedure for constructing approximants to functions on the sphere and disk. We have implemented this procedure in the Spherefun and Diskfun software programs[1], which are integrated within the Chebfun software program [21]. The underlying algorithms in these programs have been designed to exploit the low rank form of function approximants. In this way, many operations, such as integration and differentiation, can be performed as essentially 1D procedures involving Fourier or Chebyshev expansions. This is especially convenient within Chebfun, where 1D algorithms using these expansions are highly optimized. Our software is adaptive, efficient, and also intuitive to use. Investigators can easily compute with and visualize functions on the sphere and disk using familiar MATLAB commands, without concern for the underlying numerical procedures.

In Chebfun, functions in spherical and polar geometries are accessed through the creation of spherefun and diskfun objects, respectively. Below, we display the MATLAB code used to represent the function $f(\theta, \rho) = \sin\big(2\rho\sin\theta - .4\big)$ as a diskfun

[1]After our software was developed and posted publicly, another software system named "diskfun" was released in the Approxfun software system written in Julia. It is not related to this work.

object:

```
   g = diskfun(@(t,r) sin(2*r.*sin(t)-.4), 'polar')
   g =
     diskfun object:
       domain        rank    vertical scale
      unit disk       8            1.4
```

The printout provides the numerical *rank* of the function, discussed in Section 2.2, and it also displays the vertical scale, an approximation of the absolute maximum value of g.

The default setting of Diskfun assumes that functions are supplied in Cartesian coordinates. However, diskfun objects can be constructed from function handles in polar coordinates by adding the flag `'polar'` to the construction command. Similarly, spherefun objects can be constructed from function handles in either polar or Cartesian coordinates[2] (see examples in Section 4.2).

Once a diskfun or spherefun object is created, users have access to a large number of algorithms tailored to functions defined on the disk or sphere via overloaded MATLAB commands. For example, integration is performed by the `sum2` command, and differentiation is performed by `diff`.

In the following sections, we describe several of the algorithms used for computing with functions on the sphere and disk, and also provide examples showing how these procedures are accessed in Chebfun. More examples can be found online at `www.chebfun.org/examples`.

[2]In the case of the sphere, there is no need to add a flag since the number of variables in the function handle implies the coordinate system.

4.2 Numerical Computations with Functions on the Sphere

Here, we discuss several operations for computing with functions on the sphere, which are all available as part of Spherefun. In the discussion, we assume that we are working with a smooth function on the sphere, f, that has been extended to a BMC function, $\tilde{f}$, using the DFS method (see (3.1)). We assume $\tilde{f}$ is represented to approximately machine precision by an approximant of the form (3.12) constructed via the GE procedure in Figure 3.5. The functions $c_j(\theta)$ and $r_j(\theta)$ in (3.12) are 2π-periodic, and we represent them in Spherefun with Fourier expansions, i.e.,

$$c_j(\theta) = \sum_{k=-m/2}^{m/2-1} a_k^j e^{ik\theta}, \qquad r_j(\lambda) = \sum_{k=-n/2}^{n/2-1} b_k^j e^{ik\lambda}, \tag{4.1}$$

where m and n are even integers. We could go further and split the functions c_j and r_j into the functions c_j^+, r_j^+, c_j^-, and r_j^- in (3.21). In Spherefun, this can be done with the `partition` command. Then, the Fourier coefficients of these functions would satisfy the parity properties discussed in Section 3.4.

In principle, the number of Fourier modes in the expansions for $c_j(\theta)$ and $r_j(\lambda)$ in (4.1) could depend on j. Here, we use the same number of modes, m, for each $c_j(\theta)$, and n for each $r_j(\lambda)$. This allows operations on spherefun objects to be more computationally efficient as the underlying code can be vectorized.

4.2.1 Pointwise Evaluation

The evaluation of $f(x, y, z)$ on the surface of the sphere, i.e., when $x^2 + y^2 + z^2 = 1$, is computationally very efficient. This immediately follows from the low rank representation for $\tilde{f}$ since

$$f(x,y,z) = \tilde{f}(\lambda,\theta) \approx \sum_{j=1}^{K} d_j c_j(\theta) r_j(\lambda),$$

where $\lambda = \tan^{-1}(y/x)$ and $\theta = \cos^{-1}(z/(x^2+y^2)^{1/2})$. Thus, $f(x,y,z)$ can be calculated by evaluating $2K$ 1D Fourier expansions (4.1) using Horner's algorithm, which requires a total of $\mathcal{O}(K(n+m))$ operations [81].

The Spherefun software allows users to evaluate using either Cartesian or spherical coordinates. In the former case, points that do not exactly satisfy $x^2+y^2+z^2=1$ are projected to the unit sphere in the radial direction.

4.2.2 Computation of Fourier Coefficients

The DFS method and our low rank approximant for $\tilde{f}$ means that the FFT is applicable when computing with $\tilde{f}$. Here, we assume that the Fourier coefficients for c_j and r_j in (4.1) are unknown. In full tensor-product form, the bi-periodic BMC-I function can be approximated using a 2D Fourier expansion. That is,

$$\tilde{f}(\lambda,\theta) \approx \sum_{j=-m/2}^{m/2-1} \sum_{k=-n/2}^{n/2-1} X_{jk} e^{ij\theta} e^{ik\lambda}. \tag{4.2}$$

The $m \times n$ matrix X of Fourier coefficients can be directly computed by sampling $\tilde{f}$ on a 2D uniform tensor-product grid and then applying the 2D FFT, costing $\mathcal{O}(mn\log(mn))$ operations. However, using the low rank structure of $\tilde{f}$, we can compute a low rank approximation of X in $\mathcal{O}(K(m\log m + n\log n))$ operations.

After sampling $\tilde{f}$ along the adaptively selected skeleton from Section 3.2, the coefficients for c_j and r_j in (4.1) are computed in $\mathcal{O}(K^3 + K^2(m+n) + K(m\log m + n\log n))$ operations. This is achieved by performing GE on the skeleton [68] to obtain the values of c_j and r_j on a uniform grid, and then using the FFT. The matrix X will be calculated in low rank form:

$$X = ADB^T, \tag{4.3}$$

where A is an $m \times K$ matrix and B is an $n \times K$ matrix, so that the jth column of A and B is the vector of Fourier coefficients for c_j and r_j, respectively. The matrix D is a $K \times K$ diagonal matrix containing d_j. From the low rank format in (4.3), one can calculate the entries of X by matrix multiplication in $\mathcal{O}(Kmn)$ operations. However, many operations, such as integration and differentiation, only require the low rank form of X.

The inverse operation is to sample $\tilde{f}$ on an $m \times n$ uniform grid over $[-\pi, \pi] \times [-\pi, \pi]$ given its Fourier coefficient matrix. If X is given in low rank form, then this can be achieved in $\mathcal{O}(K(m \log m + n \log n))$ operations via the inverse FFT.

These two procedures are critical to Spherefun. As exemplified in this section, we rely on a coefficient-based representation of $\tilde{f}$ for several algorithms and in fact store $\tilde{f}$ in this way, and yet we also regularly require values of $\tilde{f}$. The Fourier coefficients of a spherefun object are computed by the `coeffs2` command, and the values of the function at a uniform grid $[\pi, \pi] \times [0, \pi]$ are computed by the command `sample`.

4.2.3 Integration

The definite integral of a function $f(x, y, z)$ over the sphere can be efficiently computed in Spherefun as follows:

$$\begin{aligned}\int_{\mathbb{S}^2} f(x, y, z) dx\, dy\, dz &= \int_0^\pi \int_{-\pi}^\pi \tilde{f}(\lambda, \theta) \sin\theta \, d\lambda \, d\theta \\ &\approx \sum_{j=1}^K d_j \int_0^\pi c_j(\theta) \sin\theta \, d\theta \int_{-\pi}^\pi r_j(\lambda) \, d\lambda.\end{aligned}$$

Hence, the approximation of the integral of f over the sphere reduces to $2K$ one-dimensional integrals involving 2π-periodic functions.

Due to the orthogonality of the Fourier basis, the integrals of $r_j(\lambda)$ are given as

$$\int_{-\pi}^{\pi} r_j(\lambda)\, d\lambda = 2b_0^j, \qquad 1 \leq j \leq K,$$

where b_0^j is the zeroth Fourier coefficient of r_j in (4.1). The integrals of $c_j(\theta)$ are over half the period, so the expressions are a bit more complicated. Using symmetry and orthogonality,

$$\int_{0}^{\pi} c_j(\theta) \sin\theta\, d\theta = \sum_{k=-m/2}^{m/2-1} w_k a_k^j, \qquad 1 \leq j \leq K, \tag{4.4}$$

where $w_{\pm 1} = 0$ and $w_k = (1 + e^{i\pi k})/(1 - k^2)$ for $-m/2 \leq k \leq m/2 - 1$ and $k \neq \pm 1$. Here, a_k^j are the Fourier coefficients for c_j in (4.1).

Therefore, we can compute the surface integral of $f(x, y, z)$ over the sphere in $\mathcal{O}(Km)$ operations. This algorithm is used in the `sum2` command of spherefun. For example, the function $f(x, y, z) = 1 + x + y^2 + x^2y + x^4 + y^5 + (xyz)^2$ has a surface integral of $216\pi/35$ and can be calculated in spherefun as follows:

```
f = spherefun(@(x,y,z) 1+x+y.^2+x.^2.*y+x.^4+y.^5+...
(x.*y.*z).^2);
sum2(f)
ans =
     19.388114662154155
```

The error is computed as `abs(sum2(f)-216*pi/35)` and is given by 3.553×10^{-15}. We can further reduce the cost of this computation by applying results associated with the parity properties in Section (3.4). The odd-antiperiodic functions in the decomposition (3.21) will not contribute to the integral, and can therefore be excluded from the computation.

4.2.4 Differentiation

Differentiation of a function on the sphere with respect to spherical coordinates (λ, θ) can lead to singularities at the poles, even for smooth functions [61]. For example, consider the simple function $f(\lambda, \theta) = \cos\theta$. The θ-derivative of this function is $\sin\theta$, which is continuous on the sphere but not smooth at the poles, as it does not satisfy the parity properties in Section 3.4. Fortunately, one is typically interested in the derivatives that arise in applications such as vector calculus operations involving the gradient, divergence, curl, or Laplacian. All of these operators can be expressed in terms of the components of the surface gradient with respect to the Cartesian coordinate system [24].

Let $\mathbf{e}^x$, $\mathbf{e}^y$, and $\mathbf{e}^z$ denote the unit vectors in the x, y, and z directions, respectively, and $\nabla_{\mathcal{S}}$ denote the surface gradient on the sphere in Cartesian coordinates. From the chain rule, we can derive the Cartesian components of $\nabla_{\mathcal{S}}$ as

$$\mathbf{e}^x \cdot \nabla_{\mathcal{S}} := \frac{\partial^{\mathrm{t}}}{\partial x} = -\frac{\sin\lambda}{\sin\theta}\frac{\partial}{\partial\lambda} + \cos\lambda\cos\theta\frac{\partial}{\partial\theta}, \tag{4.5}$$

$$\mathbf{e}^y \cdot \nabla_{\mathcal{S}} := \frac{\partial^{\mathrm{t}}}{\partial y} = \frac{\cos\lambda}{\sin\theta}\frac{\partial}{\partial\lambda} + \sin\lambda\cos\theta\frac{\partial}{\partial\theta}, \tag{4.6}$$

$$\mathbf{e}^z \cdot \nabla_{\mathcal{S}} := \frac{\partial^{\mathrm{t}}}{\partial z} = \sin\theta\frac{\partial}{\partial\theta}. \tag{4.7}$$

Here, the superscript ‘t’ indicates that these operators are tangential gradient operations. The result of applying any of these operators to a smooth function on the sphere is a smooth function on the sphere [61]. For example, applying $\partial^t/\partial x$ to $\cos\theta$ gives $-\cos\lambda\,\sin\theta\,\cos\theta$, which in Cartesian coordinates is $-xz$ restricted to the sphere.

As with integration, our low rank approximation for $\tilde{f}$ can be exploited to compute (4.5)–(4.7) efficiently. For example, using (4.1), we have

$$\begin{aligned}\frac{\partial^{\mathrm{t}}\tilde{f}}{\partial x} &= -\frac{\sin\lambda}{\sin\theta}\frac{\partial\tilde{f}}{\partial\lambda} + \cos\lambda\cos\theta\frac{\partial\tilde{f}}{\partial\theta} \\ &\approx -\sum_{j=1}^{K}\left(\frac{c_j(\theta)}{\sin\theta}\right)\left(\sin\lambda\,\frac{\partial r_j(\lambda)}{\partial\lambda}\right) + \sum_{j=1}^{K}\left(\cos\theta\,\frac{\partial c_j(\theta)}{\partial\theta}\right)\big(\cos\lambda\, r_j(\lambda)\big)\,.\end{aligned} \tag{4.8}$$

Although it appears that an artificial singularity has been introduced in (4.8) with the division by $\sin\theta$, this is not the case. Suppose that $\tilde{f}$ is nonzero at $\theta = 0$ or $\theta = \pm\pi$. Then, writing (4.8) as a sum of even-π-periodic and odd π-antiperiodic functions as in (3.21), note that $r_1^+(\lambda)$ is constant because of (3.11), and any terms involving $\partial r_1^+(\lambda)/\partial\lambda$ vanish. For $2 \le j \le K^+$, each $c_j^+(\theta)$ is an even function, and can therefore be written as a cosine series, i.e., $c_j^+(\theta) = \sum_{\ell=0}^{q}\alpha_\ell^j\cos(\ell\theta)$, for some coefficients $\{\alpha_\ell^j\}_{\ell=0}^{q}$. Furthermore, by (3.11), $c_j^+(0) = c_j^+(\pm\pi) = 0$. Letting
$\theta = \cos^{-1}(t)$, we can transform c_j^+ into a Chebyshev series. This series can be expressed as $c_j^+(t) = \sum_{\ell=0}^{q}\alpha_\ell^j(t-1)(t+1)T_\ell(t)$, where we have used the fact that $c_j(t)$ must have roots at $t = \pm1$ to satisfy that $c_j(\theta) = 0$ when $\theta = 0, \pm\pi$. Transforming back to θ, this gives $c_j^+(\theta) = \sin^2\theta\sum_{\ell=0}^{q}\alpha_\ell^j\cos(\ell\theta)$, and it is clear that for $j > 2$, each $c_j^+(\theta)$ function is divisible by $\sin\theta$. Every $c_j^-(\theta)$ is an odd function that can be expanded as a sine series, and is therefore also divisible by $\sin\theta$. This means that approximants constructed via the GE procedure are differentiable over the poles of the sphere.

Using (4.8), it follows that $\partial^{\mathrm{t}}\tilde{f}/\partial x$ can be calculated by essentially 1D algorithms involving differentiating Fourier expansions, as well as multiplying and dividing them by cosine and sine. In the above expression, for example, we have

$$(\sin\lambda)\frac{\partial r_j(\lambda)}{\partial\lambda} = \sum_{k=-n/2-1}^{n/2}\frac{-(k+1)b_{k+1}^j + (k-1)b_{k-1}^j}{2}e^{ik\lambda}$$

and

$$(\cos\lambda) r_j(\lambda) = \sum_{k=-n/2-1}^{n/2} \frac{b^j_{k+1} + b^j_{k-1}}{2} e^{ik\lambda},$$

where $b^j_{-n/2-2} = b^j_{-n/2-1} = 0$ and $b^j_{n/2} = b^j_{n/2+1} = 0$. Note that the number of coefficients in the Fourier representations of these derivatives has increased by two modes to account for multiplication by $\sin\lambda$ and $\cos\lambda$. Similarly, we also have

$$(\cos\theta)\frac{\partial c_j(\theta)}{\partial\theta} = \sum_{k=-m/2-1}^{m/2+1} \frac{(k+1)ia^j_{k+1} + (k-1)ia^j_{k-1}}{2} e^{ik\theta},$$

where $a^j_{-m/2-2} = a^j_{-m/2-1} = 0$ and $a^j_{m/2} = a^j_{m/2+1} = 0$. Lastly, for (4.8), we must compute $c_j(\theta)/\sin\theta$. This can be done as follows:

$$\frac{c_j(\theta)}{\sin\theta} = \sum_{k=-m/2}^{m/2-1} (M_{\sin}^{-1}\underline{a}^j)_k e^{ik\lambda}, \qquad M_{\sin} = \begin{bmatrix} 0 & \frac{i}{2} & & & & \\ -\frac{i}{2} & 0 & \frac{i}{2} & & & \\ & -\frac{i}{2} & \ddots & \ddots & & \\ & & \ddots & \ddots & \frac{i}{2} & \\ & & & -\frac{i}{2} & 0 & \frac{i}{2} \\ & & & & -\frac{i}{2} & 0 \end{bmatrix}, \qquad (4.9)$$

where $\underline{a}^j = (a^j_{-m/2}, \ldots, a^j_{m/2-1})^T$. Here, $M_{\sin}^{-1}$ exists because it is of size $m \times m$ and m is an even integer.[3]

Our treatment of the artificial pole singularity by operating on the coefficients directly appears to be new. The standard technique when using spherical coordinates on a latitude-longitude grid is to shift the grid in the latitude direction so that the poles are not sampled [17, 25, 83]. In (4.9), there is no need to explicitly avoid the pole, it is easy to implement, and is possibly more accurate numerically than shifting the grid. The total cost of this algorithm is $\mathcal{O}(K(m+n))$ operations.

We use similar ideas to compute (4.6) and (4.7), which require a similar number of operations. They are implemented in the `diff` command of spherefun.

[3]This follows from the observation that the eigenvalues of $M_{\sin}$ are $\cos(\pi\ell/(m+1))$, $\ell = 1, \ldots, m$, so that when m is even all the eigenvalues are nonzero.

4.2.5 The L_2 Norm

In Spherefun, we employ the L_2 norm, which is the continuous analogue of the matrix Frobenius norm [69]. This is one of the few instances where it makes more sense to work with f directly, rather than $\tilde{f}$. For f expressed in spherical coordinates, the Frobenius or L_2 norm of f is given by

$$||f||_2^2 = \int_{-\pi}^{\pi} \int_{0}^{\pi} f^2(\lambda, \theta) \sin\theta \, d\theta \, d\lambda. \tag{4.10}$$

Computing $||f||_2$ using (4.10) directly is numerically unstable, especially when f is near zero. A more stable formulation is given in [53]: If f is L_2 integrable, then

$$||f||_2^2 = \sum_{j=1}^{\infty} \sigma_j^2, \tag{4.11}$$

where $\sigma_1 \geq \sigma_2 \geq \cdots \sigma_j \geq \cdots$ are real and nonnegative numbers referred to as the *weighted singular values* of f. For this reason, we are interested in the weighted SVD of f, which is given by

$$f(\lambda, \theta) = \sum_{j=1}^{\infty} \sigma_j u_j(\theta) v_j(\lambda), \qquad (\lambda, \theta) \in [-\pi, \pi] \times [0, \pi]. \tag{4.12}$$

The singular functions $\{u_j(\theta)\}$, $\theta \in [0, \pi]$, and $\{v_j(\lambda)\}$, $\lambda \in [-\pi, \pi]$, are orthonormal, respectively, under the following inner products:

$$< u, s >_\theta = \int_{0}^{\pi} u(\theta) s(\theta) \sin\theta \, d\theta, \qquad < v, w > = \int_{-\pi}^{\pi} v(\lambda) w(\lambda) \, d\lambda. \tag{4.13}$$

The weighted SVD for a function on the sphere is determined by applying a generalization of QR factorization to quasimatrices. Restricting the low rank approximation to $\tilde{f}$ given by (3.12) to $(\lambda, \theta) \in [-\pi, \pi] \times [0, \pi]$, we form a $[0, \pi] \times K$ quasimatrix C such that the j^{th} column of C is $c_j(\theta)$ in (3.12) restricted to $\theta \in [0, \pi]$. Similarly, we form the $[-\pi, \pi] \times K$ quasimatrix R such that the j^{th} column of R is $r_j(\lambda)$, $\lambda \in [-\pi, \pi]$.

If D is a diagonal matrix containing the values $\{d_j\}$ from (3.12), then $f \approx CDR^T$. A QR quasimatrix factorization with respect to the standard L_2 inner product is given in [73] and selects the Legendre polynomials to orthogonalize against. We apply this procedure to R, but orthogonalize against the Fourier basis. In consideration of (4.13), C is orthogonalized against the normalized Legendre polynomials $\tilde{P}_m(\cos\theta)$, which are orthonormal with respect to the inner product $< \cdot, \cdot >_\theta$ in (4.13). This finds $\{u_j(\theta)\}$ in (4.12). Once the QR factorizations for C and R are known, the SVD is determined through standard techniques, as discussed in [69].

In addition to providing a mathematically stable way to compute (4.10), the weighted SVD gives the best rank–K approximation to f with respect to the weighted L_2 inner product on the disk. Unfortunately, use of the weighted SVD is limited because the rank 1 terms in (4.12) may be discontinuous at the poles, and consequently, the truncation of (4.12) may not be smooth. The SVD is accessed in Spherefun through the command `svd(f)`, and it is used internally within `norm(f)` to compute $||f||_2$.

4.2.6 Vector-valued Functions and Vector Calculus on the Sphere

Expressing vector-valued functions that are tangent to the sphere in spherical coordinates is very inconvenient since the unit vectors in this coordinate system are singular at the poles [61]. It is therefore common practice to express vector-valued functions in Cartesian coordinates. In Cartesian coordinates, the components of the vector field are smooth and can be approximated using the low rank techniques developed in Section 3.2.

All the standard vector-calculus operations can be expressed in terms of the tangential derivative operators in (4.5)–(4.7). For example, the surface gradient,

$\nabla_{\mathcal{S}}$, of a scalar-valued function f on the sphere is given by the vector

$$\nabla_{\mathcal{S}} f = \begin{bmatrix} \dfrac{\partial^{\mathrm{t}} f}{\partial x}, & \dfrac{\partial^{\mathrm{t}} f}{\partial y}, & \dfrac{\partial^{\mathrm{t}} f}{\partial z} \end{bmatrix}^T,$$

where the partial derivatives are defined in (4.5)–(4.7). The surface divergence and curl of a vector field $\mathbf{f} = \begin{bmatrix} f_1, & f_2, & f_3 \end{bmatrix}^T$ that is tangent to the sphere can also be computed using (4.5)–(4.7) as

$$\nabla_{\mathcal{S}} \cdot \mathbf{f} = \frac{\partial^{\mathrm{t}} f_1}{\partial x} + \frac{\partial^{\mathrm{t}} f_2}{\partial y} + \frac{\partial^{\mathrm{t}} f_3}{\partial z} \quad \text{and} \quad \nabla_{\mathcal{S}} \times \mathbf{f} = \begin{bmatrix} \dfrac{\partial^{\mathrm{t}} f_3}{\partial y} - \dfrac{\partial^{\mathrm{t}} f_2}{\partial z} \\ \dfrac{\partial^{\mathrm{t}} f_1}{\partial z} - \dfrac{\partial^{\mathrm{t}} f_3}{\partial x} \\ \dfrac{\partial^{\mathrm{t}} f_2}{\partial x} - \dfrac{\partial^{\mathrm{t}} f_1}{\partial y} \end{bmatrix}.$$

The result of the surface curl $\nabla_{\mathcal{S}} \times \mathbf{f}$ is a vector that is tangent to the sphere.

In 2D, one can define the "curl of a scalar-valued function" as the cross product of the unit normal vector to the surface and the gradient of the function. For a scalar-valued function on the sphere, the curl in Cartesian coordinates is given by

$$\mathbf{n} \times \nabla_{\mathcal{S}} f = \begin{bmatrix} y\dfrac{\partial^{\mathrm{t}} f}{\partial z} - z\dfrac{\partial^{\mathrm{t}} f}{\partial y} \\ z\dfrac{\partial^{\mathrm{t}} f}{\partial x} - x\dfrac{\partial^{\mathrm{t}} f}{\partial z} \\ x\dfrac{\partial^{\mathrm{t}} f}{\partial y} - y\dfrac{\partial^{\mathrm{t}} f}{\partial x} \end{bmatrix}, \tag{4.14}$$

where x, y, and z are points on the unit sphere given by (2.13). This follows from the fact that the unit normal vector at (x, y, z) on the unit sphere is just $\mathbf{n} = (x, y, z)^T$.

The final vector calculus operation we consider is the vorticity of a vector field, which for a two-dimensional surface is a scalar-valued function defined as
$\zeta = (\nabla_{\mathcal{S}} \times \mathbf{f}) \cdot \mathbf{n}$, and can be computed based on the operators described above.

Vector-valued functions are represented in Chebfun by spherefunv objects. These objects contain three spherefun objects, one for each component of the vector-valued function, and can be used for computations in the same way as spherefun objects. Low rank techniques described in Section 3.2 are employed on each component separately. The operations listed above can be computed using the `grad`, `div`, `curl`, and `vort` functions; see Figure 4.1 for an example.

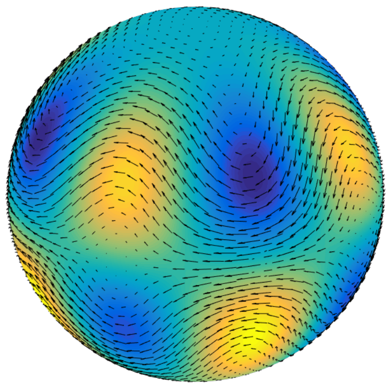


Figure 4.1: Arrows indicate the tangent vector field generated from $\mathbf{u} = \nabla_{\mathcal{S}} \times \psi$, where $\psi(\lambda, \theta) = \cos\theta + (\sin\theta)^4 \cos\theta \cos(4\lambda)$, which is the stream function for the Rossby–Haurwitz benchmark problem for the shallow water wave equations [79]. After constructing ψ in Spherefun, the tangent vector field was computed using `u = curl(psi)`, and plotted using `quiver(u)`. The superimposed false color plot represents the vorticity of $\mathbf{u}$ and is computed using `vort(u)`.

4.2.7 Miscellaneous Operations

The spherefun class is written as part of Chebfun, which means that spherefun objects have immediate access to all the operations available in Chebfun. For operations that do not require a strict adherence to the symmetry of the sphere, we can use Chebfun2 with spherical coordinates [67]. Important examples include two-dimensional optimization such as `min2`, `max2`, and `roots` as well as continuous linear algebra operators such as `lu` and `flipud`. The operations that use the Chebfun2 technology are performed seamlessly without user intervention.

4.3 Numerical Computations with Functions on the Disk

In this section, we describe the underlying mathematics for several of the algorithms used in Diskfun. These methods rely on the fact that every smooth function g on the disk is associated with a BMC-II function $\tilde{g}$ that is periodic in θ. We compute with a low rank approximation to $\tilde{g}$ as in (3.14), which is constructed by the GE procedure in Figure 3.5. In (3.14), each $c_j(\rho)$ and $r_j(\theta)$ can be expressed by a Chebyshev and Fourier series, respectively, so that for $1 \leq j \leq K$,

$$c_j(\rho) = \sum_{\ell=0}^{n-1} a_\ell^j \, T_\ell(\rho), \qquad r_j(\theta) = \sum_{k=-m/2}^{m/2-1} b_k^j \, e^{i\theta k}, \tag{4.15}$$

where $T_\ell(\rho)$ is the Chebyshev polynomial of degree ℓ, and m is an even integer.

The algorithms for computing with functions represented in Fourier and Chebyshev bases differ considerably from one another. However, implementation within Chebfun environment is significantly simplified due to Chebfun's underlying object-oriented class structure. For example, Chebfun overloads commands such as `sum(h)` (integration) or `diff(h)` (differentiation), so that the same syntax executes different underlying algorithms based on whether the object `h` represents a periodic function or not [81].

4.3.1 Pointwise Evaluation

To efficiently evaluate $\tilde{g}$ at a fixed point (θ_*, ρ_*), we use (3.14), observing that

$$\tilde{g}(\theta_*, \rho_*) \approx \sum_{j=1}^{K} d_j c_j(\rho_*) r_j(\theta_*). \tag{4.16}$$

Evaluation of $\tilde{g}$ proceeds as $2K$ 1D function evaluations. Functions $c_j(\rho)$, $1 \leq j \leq K$, are evaluated using Clenshaw's algorithm [74, Ch. 19], and functions $r_j(\theta)$, $1 \leq j \leq K$, are evaluated using Horner's scheme [81]. Altogether, this requires a total $\mathcal{O}(K(n+m))$ operations, and this algorithm is implemented in the `feval` command.

4.3.2 Computation of Fourier–Chebyshev Coefficients

As with functions on the sphere, the low rank form of $\tilde{g}$ facilitates the use of fast transform methods based on the FFT. We can write the truncated Fourier–Chebyshev expansion of $\tilde{g}$ as follows:

$$\tilde{g}(\theta, \rho) \approx \sum_{k=-m/2}^{m/2-1} \sum_{j=0}^{n-1} X_{jk} T_j(\rho) e^{ik\theta}, \tag{4.17}$$

where X is a matrix whose entries are the 2D Fourier–Chebyshev coefficients of $\tilde{g}$. Using the low rank form of $\tilde{g}$ given by (3.14), the matrix X can also be expressed in low rank form as

$$X = ADB^T.$$

Here, A is an $m \times K$ matrix whose j^{th} column contains the coefficients $\{a_k^j\}$ from (4.15), B is an $n \times K$ matrix whose j^{th} column contains the coefficients $\{b_\ell^j\}$ from (4.15), and D is a diagonal matrix consisting of the pivot values $\{d_j\}$. Given a sample of $\tilde{g}$ on an $n \times m$ Fourier–Chebyshev grid, the direct computation of the Fourier–Chebyshev coefficients of $\tilde{g}$ costs $\mathcal{O}(mn \log(mn))$ operations. However, using the GE procedure in Section 3.2, the low rank form of X can be found in only $\mathcal{O}(K^3 + K^2(m+n) + K(m \log m + n \log n))$ operations. This is because once the GE process adaptively selects the skeleton representing $\tilde{g}$ at a cost of $\mathcal{O}(K^3)$, the coefficients in (4.15) for every $c_j(\rho)$ and $r_j(\theta)$ in (3.14) can be found in $\mathcal{O}(K^2(m+n) + K(m \log m + n \log n))$ operations by sampling $\tilde{g}$ along the appropriate row and column slices, and then using FFT-based algorithms.

Several procedures, such as integration and differentiation, can be executed using the ADB^T factorization of X. Using the command `coeffs2` in Diskfun, X can be explicitly computed with an additional $\mathcal{O}(Kmn)$ operations.

The above operation retrieves coefficients when supplied with a sample of $\tilde{g}$, and the inverse of this operation provides an efficient way to sample $\tilde{g}$ on a $p \times q$ Fourier–Chebyshev grid once the coefficient matrix X is known. Given X in low rank form, this proceeds in $\mathcal{O}(K(p\log p + q\log q))$ operations, and is accessed in diskfun through the `sample` command.

These algorithms are regularly employed in Diskfun, as we store a coefficient-based representation of $\tilde{g}$, but frequently require the values of $\tilde{g}$.

4.3.3 Integration

To integrate $\tilde{g}(\theta, \rho)$ over the unit disk, we again take advantage of the low rank form of (2.12), transforming the double integral into sums of 1D integrals:

$$\int_{-\pi}^{\pi}\int_{0}^{1} \tilde{g}(\theta,\rho)\rho\, d\rho\, d\theta \approx \sum_{j=1}^{K} d_j \int_{-\pi}^{\pi} r_j(\theta) d\theta \int_0^1 c_j(\rho)\ \rho\, d\rho. \tag{4.18}$$

For integration of the periodic $r_j(\theta)$ functions, the trapezoidal rule is used. To evaluate $\int_0^1 c_j(\rho)\rho d\rho$, the coefficients for $\rho c_j(\rho)$ are computed, and then Clenshaw–Curtis quadrature is applied [74, Ch. 19]. These $2K$ 1D integrals can be computed in a total of $\mathcal{O}(Kn)$ operations. This can be further reduced using (3.22): the odd, π-antiperiodic terms will not contribute to the integral and can be discarded from the computation.

This algorithm is implemented in the `sum2` command. For example, the integral of $g(x,y) = -x^2 - 3xy - (y-1)^2$ over the unit disk is $-3\pi/2$, and this is computed in Diskfun using two lines of code:

```
g= diskfun(@(x,y) -x.^2-3*x.*y -(y-1).^2)
sum2(g)
ans =


   -4.712388980384692
```

The error for this calculation is determined with `abs(sum2(f)+3*pi/2)`, which gives 1.7764×10^{-15}.

4.3.4 Differentiation

When considering derivatives on the disk, note that partial differentiation with respect to ρ can lead to artificial singularities at $\rho = 0$. For example, if $g(\theta, \rho) = \rho^2$, then $\partial g/\partial \rho = 2\rho$, which is not smooth on the disk. In contrast, for a smooth function $\tilde{g}$, partial derivatives with respect to x and y will always be well-defined.[4] For this reason, and because of the usefulness of these operators in vector calculus (see Section 4.3.6), we consider efficient and stable ways to calculate $\partial\tilde{g}/\partial x$ and $\partial\tilde{g}/\partial y$.

By (2.15), $\rho = \sqrt{x^2 + y^2}$, and $\theta = \tan^{-1}(y/x)$, so the chain rule can be applied to obtain

$$\frac{\partial \tilde{g}}{\partial x} = \cos\theta \frac{\partial \tilde{g}}{\partial \rho} - \frac{1}{\rho}\sin\theta \frac{\partial \tilde{g}}{\partial \theta}, \tag{4.19}$$

$$\frac{\partial \tilde{g}}{\partial y} = \sin\theta \frac{\partial \tilde{g}}{\partial \rho} + \frac{1}{\rho}\cos\theta \frac{\partial \tilde{g}}{\partial \theta}. \tag{4.20}$$

Exploiting the low rank form given in (3.14), equation (4.19) can be written as

$$\frac{\partial \tilde{g}}{\partial x} \approx \sum_{j=1}^{K} d_j \left(\frac{\partial c_j(\rho)}{\partial \rho} \right) \Big(\cos\theta r_j(\theta) \Big) - \sum_{j=1}^{K} d_j \left(\frac{c_j(\rho)}{\rho} \right) \left(\sin\theta \frac{\partial r_j(\theta)}{\partial \theta} \right), \tag{4.21}$$

[4]Note that unlike the partial derivatives computed for the sphere (see Section 4.2), which are surface gradients, the partial derivatives on the disk are defined in the usual sense.

and (4.20) can be similarly described.

Here we make an important observation. The above result establishes that approximants are differentiable at $\rho = 0$ only if $\sum_{j=1}^{K} c_j(\rho)$ has a root at ρ. Suppose $\tilde{g}$ is nonzero at $\rho = 0$ and write the approximant in the form given by (3.22). Then, because of (3.13), for $j \geq 2$, each term $d_j^+ c_j^+(\rho) r_j^+(\theta)$ is zero at $\rho = 0$. Since $c_j^+(\rho)$ is an even Chebyshev polynomial, it must be of the form $A_1\rho^2 + A_2\rho^4 + \cdots + A_q\rho^{2q}$, where $q \leq \lfloor (n-1)/2 \rfloor$. This implies that these functions are all divisible by ρ. For $j = 1$, $r_1^+(\theta)$ is constant by (3.13), and so all terms in (4.21) involving derivatives of $r_1^+(\theta)$ with respect to θ vanish. Since every $c_j^-(\rho)$ function for $1 \leq j \leq K^-$ is an odd function, these are also always divisible by ρ. This means that the approximants constructed by the BMC-II structure preserving GE procedure have inherited properties that ensure differentiability over $\rho = 0$.

Using (4.15), we have

$$\sin\theta \, \frac{\partial r_j(\theta)}{\partial \theta} = \sum_{k=-m/2}^{m/2-1} \frac{-(k+1)b_{k+1}^j + (k-1)b_{k-1}^j}{2} e^{ik\theta}, \tag{4.22}$$

$$\cos\theta \, r_j(\theta) = \sum_{k=-m/2}^{m/2-1} \frac{b_{k+1}^j + b_{k-1}^j}{2} e^{ik\theta}, \tag{4.23}$$

where $b_{-m/2-1}$ and $b_{m/2}$ are set to zero. Expanding each $c_j(\rho)$ as in (4.15), the recursion formula in [43, p. 34] gives the coefficients for $\partial c_j/\partial\rho$ in $\mathcal{O}(n)$ operations. To determine $c_j(\rho)/\rho$, we follow a procedure analogous to the one introduced in Section 4.2.4. We construct the operator B_ρ, which represents multiplication by the function $g(\rho) = \rho$ in the Chebyshev basis. Then, we have

$$\frac{c_j(\rho)}{\rho} = \sum_{\ell=0}^{n-1} (B_\rho^{-1} \underline{a}^j)_\ell T_\ell(\rho), \qquad B_\rho = \begin{bmatrix} 0 & \frac{1}{2} & & & & \\ 1 & 0 & \frac{1}{2} & & & \\ & \frac{1}{2} & \ddots & \ddots & & \\ & & \ddots & \ddots & \frac{1}{2} & \\ & & & \frac{1}{2} & 0 & \frac{1}{2} \\ & & & & \frac{1}{2} & 0 \end{bmatrix}, \tag{4.24}$$

where $\underline{a}^j = (a_0^j, \ldots, a_{n-1}^j)^T$. Here, B_ρ^{-1} exists because we choose B to be of size $n \times n$, where n is an even integer. Working directly with the coefficients via (4.24) allows one to bypass the artificial singularity introduced in (4.21), without explicitly avoiding computation at $\rho = 0$.

Differentiation is accessed through the `diff` command in Diskfun, and requires $\mathcal{O}(K(m+n))$ operations to perform.

4.3.5 The L_2 Norm

Just as with Spherefun, we frequently require the L_2 norm in Diskfun. This is one of the few instances in Diskfun where it makes more sense to work with g directly, as opposed to $\tilde{g}$. For g expressed in polar coordinates, the Frobenius or L_2 norm of g is given by

$$||g||_2^2 = \int_{-\pi}^{\pi} \int_0^1 g^2(\theta, \rho) \rho \, d\rho \, d\theta. \tag{4.25}$$

Since computing $||g||_2$ using (4.25) directly is numerically unstable, we use a more stable formulation of $||g||_2$ given by [53]: If g is L_2 integrable, then

$$||g||_2^2 = \sum_{j=1}^{\infty} \sigma_j^2, \tag{4.26}$$

where $\sigma_1 \geq \sigma_2 \geq \cdots \sigma_j \geq \cdots$ are real and nonnegative numbers referred to as the *weighted singular values* of f. This requires finding the weighted SVD of g,

$$g(\theta,\rho)=\sum_{j=1}^{\infty}\sigma_j u_j(\rho)v_j(\theta), \qquad (\theta,\rho)\in[-\pi,\pi]\times[0,1]. \tag{4.27}$$

Here, the singular functions $\{u_j(\rho)\}$, $\rho\in[0,1]$, and $\{v_j(\theta)\}$, $\theta\in[-\pi,\pi]$, are orthonormal, respectively, under the following inner products:

$$< u,s >_\rho=\int_0^1 u(\rho)s(\rho)\rho\,d\rho, \qquad < v,w >=\int_{-\pi}^{\pi} v(\theta)w(\theta)\,d\theta. \tag{4.28}$$

Just as in Section 4.2.5, we use a generalization of the QR factorization for quasimatrices to find the weighted SVD for a function on the disk. Restricting the approximation to $\tilde{g}$ in (3.14) to $(\theta,\rho)\in[-\pi,\pi]\times[0,1]$ and forming the quasimatrix R consisting of the periodic $r_j(\theta)$ functions in (3.14), we apply the procedure in [73], but choose the Fourier basis to orthogonalize against. The quasimatrix C consists of the $c_j(\rho)$ functions in (3.14) except that ρ is restricted to $[0,1]$. Here, we choose to orthogonalize against

$$\frac{\sqrt{2}}{J_1(\omega_k)}J_0(\omega_k\rho), \qquad k=1,2,\ldots,$$

where J_a is the Bessel function of order a and ω_k is the k^{th} positive root of $J_0(\rho)$. This finds $\{u_j(\rho)\}$, which are orthonormal with respect to (4.28). Once the QR factorizations for C and R are known, the SVD is determined through standard techniques [69].

The weighted SVD provides a numerically stable way to compute (4.25), and additionally it gives the best rank K approximation to g with respect to the weighted L_2 inner product on the disk. Use of the weighted SVD is limited because the rank 1 terms in (4.27) may not be continuous at $\rho=0$, and consequently, the truncation of (4.27) is also not always smooth. The SVD is accessed in Diskfun through the command `svd(g)`, and applied internally within `norm(g)` to compute $||g||_2$.

4.3.6 Vector-valued Functions and Vector Calculus on the Disk

Vector-valued functions can also be constructed in Diskfun. These functions are represented with respect to the Cartesian coordinate basis vectors $\hat{\mathbf{i}}$ and $\hat{\mathbf{j}}$, since not all smooth vector fields defined over the disk have smooth components when represented with respect to the polar coordinate basis vectors, $\hat{\mathbf{r}}$ and $\hat{\boldsymbol{\theta}}$. For example, the vector field given by $\mathbf{g} = 0\mathbf{i} + \mathbf{j}$ is expressed as $\mathbf{g} = \sin\theta\mathbf{r} + \cos\theta\boldsymbol{\theta}$ in polar coordinates, and both of these functions are discontinous at $\rho = 0$.

Vector-valued functions are accessed in Diskfun through the creation of diskfunv objects. A diskfunv object consists of two diskfun objects, one for each component of the vector-valued function. Algorithms involving diskfunv objects are implemented for algebraic actions, such as addition and multiplication, as well as vector-based operations, such as the dot product, the cross product, and divergence. Commands that map scalar-valued functions to vector-valued functions and vice-versa, such as `grad(g)` and `curl(g)`, are also included. In the latter case, the standard interpretation is used, i.e., $\nabla \times g = [g_y, -g_x]$ when g is a scalar function, and $\nabla \times \mathbf{g} = u_x - v_y$ when $\mathbf{g} = [u, v]$ is a vector function. As an example, consider the potential functions given by

$$\begin{aligned} \psi(x,y) &= 10e^{-10(x+.3)^2-10(y+.5)^2} + 10e^{-10(x+.3)^2-10(y-.5)^2} + 15(1-x^2-y^2), \\ \phi(x,y) &= 10e^{-10(x-.6)^2-40y^2}, \end{aligned} \tag{4.29}$$

and the vector field $\mathbf{u} = (\nabla \times \psi) + \nabla\phi$. This field consists of the sum of a divergence-free term, $\nabla \times \psi$, and a curl-free term, $\nabla\phi$. Once ψ and ϕ are constructed as diskfun objects, $\mathbf{u}$ can be constructed with a single line of code: `u = curl(psi)+grad(phi)`. Figure 4.2 displays a plot of $\mathbf{u}$ together with its curl and divergence.

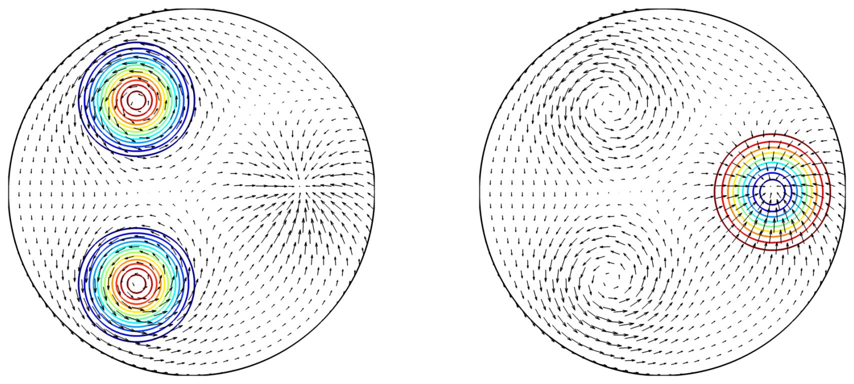


Figure 4.2: The vector function $\mathbf{u} = \nabla \times \psi + \nabla\phi$, with ψ and ϕ defined in (4.29), together with its curl, $\nabla \times \mathbf{u}$ (right), and divergence, $\nabla \cdot \mathbf{u}$ (left). The field was plotted using `quiver(u)`, while the curl and divergence were computed using `curl(u)` and `div(u)` respectively, and plotted using `contour`.

4.3.7 Miscellaneous Operations

Diskfun is included as an object class as part of Chebfun, and so has access to many of the operations in Chebfun. Operations that do not strictly require symmetry properties related to the geometry of the disk are computed using Chebfun2 with functions defined in polar coordinates [67]. This includes optimization routines, such as `min2`, `max2`, and `roots`, as well as matrix-inspired procedures such as `trace` and `lu`, among many others.

CHAPTER 5

AN OPTIMAL POISSON SOLVER ON THE SPHERE AND DISK

The DFS method and its disk analogue can be used in conjunction with Fourier and ultraspherical spectral methods to formulate optimal solvers for Poisson's equation on the sphere and disk. We have implemented these solvers in our software in an integrated way: the output given in response to the command `poisson` is automatically represented by a spherefun or diskfun object, and can therefore immediately be visualized or operated on using standard commands.

A description of the algorithm for the sphere is given in Section 5.1, and in it we apply standard operators derived from the Fourier spectral method [9]. In Section 5.2, we use the ultraspherical spectral method [48] to formulate an optimal complexity algorithm for the disk. A detailed description of the ultraspherical spectral method is provided in Appendix B.

5.1 A Poisson Solver on the Sphere

Combining the DFS method with the Fourier spectral method, an optimal Poisson solver for the sphere can be formulated. Related approaches can be found in [18, 56, 83].

Given a function f on the sphere that satisfies $\int_0^\pi \int_{-\pi}^{\pi} f(\lambda,\theta)\sin\theta d\lambda d\theta = 0$, Poisson's equation in spherical coordinates is given by

$$\sin^2\theta \frac{\partial^2 u}{\partial\theta^2} + \cos\theta\sin\theta\frac{\partial u}{\partial\theta} + \frac{\partial^2 u}{\partial\lambda^2} = \sin^2\theta f, \qquad (\lambda,\theta) \in [-\pi,\pi]\times[0,\pi]. \tag{5.1}$$

Due to the integral condition on f, there are infinitely many solutions for (5.1) that all differ by a constant. To specify a unique solution, we additionally require that

$$\int_0^\pi \int_{-\pi}^{\pi} u(\lambda,\theta)\sin\theta d\lambda d\theta = 0.$$

To ensure that the solution u is 2π-periodic and continuous over the poles, we apply the DFS method (see Section 3.1) and seek the solution $\tilde{u}$ to the related equation

$$\sin^2\theta \tilde{u}_{\theta\theta} + \cos\theta\sin\theta\tilde{u}_\theta + \tilde{u}_{\lambda\lambda} = \sin^2\theta\tilde{f}, \qquad (\lambda,\theta)\in[-\pi,\pi]^2, \tag{5.2}$$

where $\tilde{f}$ is a BMC-I, bi-periodic function. It is straightforward to deduce that $\tilde{u}$ must also be a BMC-I and bi-periodic function, and is therefore differentiable on the sphere. Since $\tilde{u}$ coincides with u on the domain $[-\pi,\pi]\times[0,\pi]$, the same integral constraint is imposed on $\tilde{u}$:

$$\int_0^\pi \int_{-\pi}^{\pi} \tilde{u}(\lambda,\theta)\sin\theta d\lambda d\theta = 0. \tag{5.3}$$

Every function in (5.2) is bi-periodic, so we discretize the equation using the Fourier spectral method [9]. In this case, we assume an approximation to $\tilde{u}$ given by

$$\tilde{u}(\lambda,\theta) \approx \sum_{j=-m/2}^{m/2-1}\sum_{k=-n/2}^{n/2-1} X_{jk}e^{ij\theta}e^{ik\lambda}, \qquad (\lambda,\theta)\in[-\pi,\pi]^2, \tag{5.4}$$

where m and n are even integers, and we seek the coefficient matrix $X \in \mathbb{C}^{m\times n}$. Writing $\tilde{u}$ and $\tilde{f}$ each as a Fourier series in λ yields

$$\tilde{u} \approx \sum_{k=-n/2}^{n/2-1} \phi_k(\theta)e^{ik\lambda}, \qquad \tilde{f} \approx \sum_{k=-n/2}^{n/2-1} \psi_k(\theta)e^{ik\lambda}, \qquad (\theta,\lambda)\in[-\pi,\pi]^2, \tag{5.5}$$

and when we plug this into (5.2), the equation separates into the following n linear ordinary differential equations:

$$\sin^2\theta\phi_k''(\theta) + \cos\theta\sin\theta\phi_k'(\theta) - k^2\phi_k(\theta) = \sin^2\theta\psi_k(\theta), \quad (\lambda,\theta) \in [-\pi,\pi]^2, \tag{5.6}$$

where $k = -n/2, \dots, n/2-1$. For each k, we have

$$\phi_k(\theta) \approx \sum_{j=-m/2}^{m/2-1} X_{jk} e^{ij\theta},$$

so finding the Fourier coefficients of ϕ_k yields the kth column of X, which we denote as X_k.

Discretization

To solve (5.6), we require operators for differentiation and for multiplication by $\sin\theta$ and $\cos\theta$. Applying these operations to (5.5), we find that, for example,

$$\frac{\partial\phi}{\partial\theta} = \sum_{j=-m/2}^{m/2-1} jiX_{jk}e^{ij\theta}, \quad (\cos\theta)\phi(\theta) = \sum_{j=-m/2}^{m/2-1} \frac{X_{j+1,k} + X_{j-1,k}}{2} e^{ij\theta},$$

and we can discretize these operations by the matrices

$$D_m = \begin{bmatrix} -\frac{mi}{2} & & & & & & \\ & \ddots & & & & & \\ & & -i & & & & \\ & & & 0 & & & \\ & & & & i & & \\ & & & & & \ddots & \\ & & & & & & \frac{(m-2)i}{2} \end{bmatrix}, \quad M_{\cos} = \begin{bmatrix} 0 & \frac{1}{2} & & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & & \\ & \frac{1}{2} & \ddots & \ddots & & \\ & & \ddots & \ddots & \frac{1}{2} & \\ & & & \frac{1}{2} & 0 & \frac{1}{2} \\ & & & & \frac{1}{2} & 0 \end{bmatrix}.$$

The operator M_{sin} for multiplication by $\sin\theta$ is formulated in a similar way and given explicitly in (4.9). Using these operators, (5.6) is discretized as

$$\left(M_{\sin}^2 D_m^2 + M_{\cos}M_{\sin}D_m - k^2\right) X_k = \mathbf{r}_k, \tag{5.7}$$

where $\mathbf{r}_k$ is a vector containing the $-m/2 \dots m/2-1$ Fourier coefficients for $\sin^2\theta\psi_k(\theta)$.

For $k \neq 0$, the linear systems in (5.7) are invertible and have a pentadiagonal structure. Each can be solved in $\mathcal{O}(m)$ operations using a sparse LU solver, such as the one employed by calling 'backslash' in MATLAB. Since there are $n-1$ such systems to solve, the overall computational cost is $\mathcal{O}(mn)$ for these systems.

When $k = 0$, the linear system (5.7) is not invertible. This is because without imposing the integral consraint (5.3), there are infinitely many solutions differing by constants, and we must apply the integral constraint to get a unique result. To discretize the constraint, we note that

$$\int_0^{\pi} \int_{-\pi}^{\pi} \tilde{u}(\lambda, \theta) \sin\theta d\lambda d\theta \approx 2\pi \sum_{j=-m/2}^{m/2-1} X_{j0} \frac{1 + e^{i\pi j}}{1 - j^2},$$

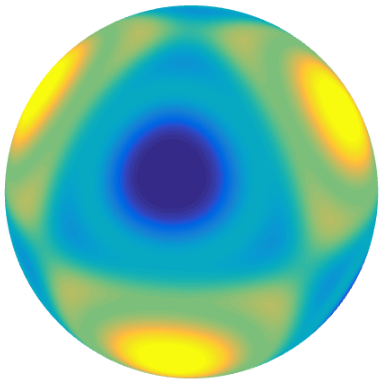
which can be written as $2\pi w^T X_0 = 0$, where the vector w is given in (4.4). We impose $2\pi w^T X_0 = 0$ on X_0 by replacing the zeroth row of the linear system $(M_{\sin}^2 D_m^2 + M_{\cos} M_{\sin} D_m) X_0 = \mathbf{r}_0$ with $2\pi w^T X_0 = 0$. We have selected the zeroth row because it is zero in the linear system. Thus, we solve the following linear system:

$$\begin{bmatrix} w^T \\ P\left(M_{\sin}^2 D_m^2 + M_{\cos} M_{\sin} D_m\right) \end{bmatrix} X_0 = \begin{bmatrix} 0 \\ P(\mathbf{r}_0) \end{bmatrix}. \qquad (5.8)$$

Here, $P \in \mathbb{R}^{(m-1)\times m}$ is a projection matrix that removes the row corresponding to the zeroth mode, i.e.,

$$P\left(v_{-m/2}, \ldots, v_{-1}, v_0, v_1, \ldots, v_{m/2-1}\right)^T = \left(v_{-m/2}, \ldots, v_{-1}, v_1, \ldots, v_{m/2-1}\right)^T.$$

The linear system in (5.8) is banded except for one dense row. It can be solved in $\mathcal{O}(m)$ using the Sherman–Morrison formula as described in Section 5.2. In practice, since solving this single linear system will not dominate the computational cost of the overall procedure, we solve using 'backslash' in MATLAB. This employs a sparse LU solver at a cost of $\mathcal{O}(m)$ operations.



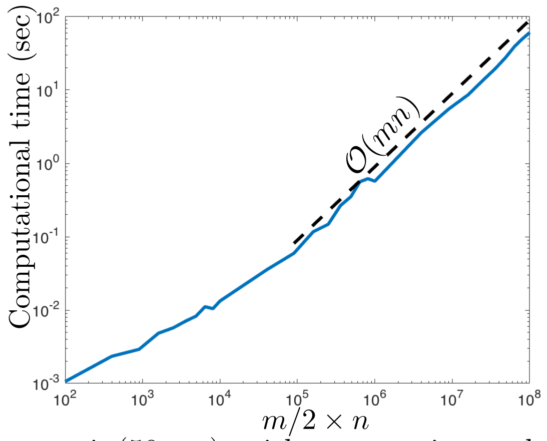


Figure 5.1: Left: Solution to $\nabla^2 u = \sin(50xyz)$ with a zero integral constraint computed by `f = spherefun(@(x,y,z) sin(50*x.*y.*z)); u = spherefun.poisson(f,0,150,150);`, which employs the algorithm above with $m = n = 150$. Right: Execution time of the Poisson solver as a function of the number of unknowns, $nm/2$, when $m = n$.

The resulting Poisson solver is said to be of optimal complexity because it finds the mn Fourier coefficients for $\tilde{u}$ in $\mathcal{O}(mn)$ operations. This does not include the computation of the Fourier coefficients of $\sin^2\theta \tilde{f}$, which, if we exploit the low rank form of $\tilde{f}$ and use the techniques described in Section 4.2.2, costs $\mathcal{O}(Kmn)$. If f is reasonably smooth, then typically $K \ll \min(m, n)$. We can additionally exploit the parity properties possessed by BMC-I functions (see Section 3.4) to further reduce the cost of the solver by a factor of 4.

An Example

Figure 5.1 (left) displays the solution to $\nabla^2 u = \sin(50xyz)$, where $m = n = 150$. Before applying the algorithm, the matrix of 2D coefficients for $\sin(50xyz)$ was found using structure-preserving GE and the ideas in Section 4.2.2. Since $\sin(50xyz)$ has a numerical rank of 12, the cost is $\mathcal{O}(mn)$ operations.

In Figure 5.1 (right), the complexity of the Poisson solver is verified by showing timings for $m = n$. We have denoted the number of degrees of freedom of the final solution as $mn/2$ since this is the number that is employed on the solution u. Without explicit parallelization, we can solve for 10^8 degrees of freedom in the solution in one minute on a standard laptop.[1]

5.2 A Poisson Solver on the Disk

Given a function $g(\theta, \rho)$ on the unit disk, we seek the solution $u(\theta, \rho)$ to Poisson's equation, $\nabla^2 u = g$, with Dirichlet boundary conditions prescribed as $u(\theta, 1) = h(\theta)$, where h is a 2π-periodic function.

To enforce that the numerical solution u is smooth over $u(\theta, 0) = 0$, we apply the disk analogue to the DFS method and consider solving the related equation, $\nabla^2 \tilde{u} = \tilde{g}$, where $\tilde{g}$ is the BMC-II extension of g given by (3.4). An additional boundary condition is given by $u(\theta, -1) = h(\theta)$.

The equation $\nabla^2 \tilde{u} = \tilde{g}$ is expressed in polar coordinates as

$$\rho^2 \frac{\partial^2 \tilde{u}}{\partial \rho^2} + \rho \frac{\partial \tilde{u}}{\partial \rho} + \frac{\partial^2 \tilde{u}}{\partial \theta^2} = \rho^2 \tilde{g}, \qquad (\theta, \rho) \in [-\pi, \pi] \times [-1, 1], \tag{5.9}$$

where the standard formulation is multiplied by by ρ^2 so that the variable coefficients are low degree polynomials in ρ. It is a straightforward exercise to show that $\tilde{u}$ must also possess BMC-II symmetry and therefore corresponds to a continuous function on the disk. Restricting $\tilde{u}$ to $[-\pi, \pi] \times [0, 1]$ gives u.

The solution $\tilde{u}$ is discretized as a Fourier–Chebyshev expansion as in (4.17):

$$\tilde{u}(\theta, \rho) \approx \sum_{k=-m/2}^{m/2-1} \sum_{j=0}^{n-1} X_{jk} T_j(\rho) e^{ik\theta}, \tag{5.10}$$

[1]Timings were done on a MacBook Pro using MATLAB 2015b without explicit parallelization.

and we seek the coefficient matrix $X \in \mathbb{C}^{n\times m}$. To find X, consider $\tilde{u}$ and $\tilde{g}$ each expressed as a Fourier series as in (2.16):

$$\tilde{u} \approx \sum_{k=-m/2}^{m/2-1} \phi_k(\rho)e^{ik\theta}, \quad \tilde{g} \approx \sum_{k=-m/2}^{m/2-1} \psi_k(\rho)e^{ik\theta}, \quad (\theta,\rho) \in [-\pi,\pi]\times[-1,1]. \tag{5.11}$$

Plugging these into (5.9), the set of equations decouples into m linear ordinary differential equations, so that for each $k = -m/2,\dots,m/2-1$,

$$\rho^2\phi_k''(\rho) + \rho\phi_k'(\rho) - k^2\phi_k(\rho) = \rho^2\psi_k(\rho), \tag{5.12}$$

with boundary conditions given by $\phi_k(1) = \gamma_k$ and $\phi_k(-1) = (-1)^k\gamma_k$, where γ_k is the kth Fourier coefficient of $h(\theta)$. The coefficients in the Chebyshev expansion of the unknown function $\phi_k(\rho)$ are the values in the kth column of X, which we will denote by X_k.

Discretization

To discretize (5.12), we use the ultraspherical spectral method (see Appendix B). We require approximations to operators that represent differentiation on $\phi_k(\rho)$ by acting on the coefficients of the Chebyshev expansion of ϕ_k. These are given in (B.7) as

$$D_1 = \begin{bmatrix} 0 & 1 & & & \\ & & 2 & & \\ & & & \ddots & \\ & & & & n-1 \\ & & & & 0 \end{bmatrix} \quad D_2 = \begin{bmatrix} 0 & 0 & 4 & & & \\ & & & 6 & & \\ & & & & \ddots & \\ & & & & & 2(n-2) \\ & & & & & 0 \\ & & & & & 0 \end{bmatrix}.$$

Here, D_1X_k gives the coefficients for the expansion of $\phi'(\rho)$ in the $C^{(1)}$ basis, and D_2X_k gives the coefficients for the expansion of $\phi''(\rho)$ in the $C^{(2)}$ basis, where $C^{(1)}$ and $C^{(2)}$ are families of ultraspherical polynomials (see Appendix B).

To find $\rho\phi'(\rho)$, we require an operator for multiplying by ρ in the $C^{(1)}$ basis. Using (B.14), this operator is explicitly given as

$$M_{1[\rho]} = \begin{bmatrix} 0 & \frac{1}{2} & & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & & \\ & \frac{1}{2} & \ddots & \ddots & & \\ & & \ddots & \ddots & \frac{1}{2} & \\ & & & \frac{1}{2} & 0 & \frac{1}{2} \\ & & & & \frac{1}{2} & 0 \end{bmatrix}. \tag{5.13}$$

Similarly, we use (B.16) to create the operator $M_{2[\rho^2]}$ for multiplying by ρ^2 in the $C^{(2)}$ basis for $\rho^2\phi''(\rho)$. As discussed in [48], the multiplication operators $M_{1[\rho]}$ and $M_{2[\rho^2]}$ are banded, and in this case the bandwidths are 1 and 2, respectively. We then apply the tridiagonal conversion operators S_1 and S_0 given by (B.11) to transform each term of the discretized ODE into the $C^{(2)}$ basis. Using these operators, (5.12) is discretized as follows:

$$\underbrace{\left(M_{2[\rho^2]}D_2 + S_1 M_{1[\rho]} D_1 + S_1 S_0 k^2\right)}_{=L} X_k = \mathbf{r}_k, \tag{5.14}$$

where the $n \times 1$ vector $\mathbf{r}_k$ consists of the first n coefficients in the expansion of $\rho^2\psi_k(\rho)$ in the $C^{(2)}$ basis. This forms a banded, pentadiagonal linear system of equations. As discussed in Appendix B, we impose boundary conditions on the system using the boundary bordering technique [48, 50]. This removes the final 2 rows of L and $\mathbf{r}_k$ in (5.14) and replaces them with the boundary conditions. We then permute the boundary conditions to the top two rows of the $n \times n$ linear system to form the system

$$\begin{bmatrix} 1 & -1 & 1 & -1 & \cdots & (-1)^{n-1} \\ 1 & 1 & 1 & 1 & \cdots & 1 \\ \\ \multicolumn{6}{c}{\underbrace{P\left(M_{2[\rho^2]}D_2 + S_1 M_{1[\rho]} D_1 + S_1 S_0 k^2\right)}_{=B}} \\ \\ \end{bmatrix} X_k = \begin{bmatrix} \gamma_k \\ \\ (-1)^k \gamma_k \\ \\ P\mathbf{r}_k \end{bmatrix}, \tag{5.15}$$

where P is a projection operator that removes the final two rows (entries) from a matrix (vector). For example, in standard MATLAB notation, $B = L(1\!:(n-2), :)$ and $P\mathbf{r}_k = \mathbf{r}_k(1\!:(n-2))$.

The results on parity in Section 3.4 show that whenever k is even, $\phi_k(\rho)$ is an even function. The odd-indexed Chebyshev coefficients for even functions are zero. Likewise, when k is odd, the even-indexed Chebyshev coefficients of $\phi_k(\rho)$ are zero. Applying this result makes one of the boundary conditions redundant, so that the system (5.15) can be reduced to a nearly tridiagonal system of size $n/2 \times n/2$ with one dense row at the top.

Without loss of generality, consider the case where k is even. Let

$$A = \begin{bmatrix} \mathbf{u} \\ P_+ B \end{bmatrix},$$

where $\mathbf{u} = [1,\, 0,\, 0,\, \cdots,\, 0]^T$ is of size $n/2 \times 1$, and P_+ is a projection operator that selects out only the terms associated with even coefficients in (5.15). For example, $P_+B = B(1\!:\!2\!:(n-2), 1\!:2\!:n)$, and $P_+P\mathbf{r_k} = \mathbf{r_k}(1\!:2\!:(n-2))$. Then, the system of equations we must solve is

$$\left[A + \mathbf{u}\mathbf{v}^T\right] P_+(X_k) = \underbrace{\begin{bmatrix} \gamma_k \\ P_+ P\mathbf{r}_k \end{bmatrix}}_{\mathbf{y}}.$$

Here, $\mathbf{v} = [0,\, 1,\, 1,\, \ldots,\, 1]^T$, and is of size $n/2 \times 1$.

We have decomposed the matrix in this equation into a sum of a tridiagonal matrix A and a rank 1 matrix $\mathbf{u}\mathbf{v}^T$. This allows us to solve the system using the Sherman–Morrison formula [58], which gives that

$$P_+X_k = A^{-1}\mathbf{y} - \frac{A^{-1}\mathbf{u}\mathbf{v}^T A^{-1}\mathbf{y}}{1 + \mathbf{v}^T A^{-1}\mathbf{u}}. \tag{5.16}$$

Since A is tridiagonal, $A^{-1}\mathbf{y}$ and $A^{-1}\mathbf{u}$ can each be solved in $\mathcal{O}(n)$ operations. The product of a matrix and the vector $\mathbf{v}$ can be computed in $\mathcal{O}(n)$ operations since $\mathbf{v}$ only has 1 nonzero entry. As a result, each column of P_+X_k is found in $\mathcal{O}(n)$ operations. This method is highly efficient and appears stable in practice, but if stability is a concern, then the system in (5.15) can also be solved with the adaptive QR algorithm in [48] at $\mathcal{O}(n)$ operations. The adaptive QR algorithm is known to be stable, even for very large n.

We will solve m such systems in total, so the overall computational cost for finding the coefficient matrix X is $\mathcal{O}(mn)$ operations, and we have saved a factor of 2 by applying parity properties. When $\tilde{g}$ is real, $\tilde{u}$ is also real, and we save an additional factor of 2 by using symmetry properties on the Fourier coefficients of $\tilde{u}$. We note that the coefficients for $\rho^2\tilde{g}$ from (5.9) can be computed using the ideas in Section 4.3.2 in an additional $\mathcal{O}(Kmn)$ operations, where K is the numerical rank of $\tilde{g}$, and it is often the case that $K \ll \min(m, n)$.

An Example

Figure 5.2 (left) displays the solution to $\nabla^2 u = g$ computed with the `poisson` command in Diskfun. Here, g is numerically a rank 16 function, given by

$$g(\theta, \rho) = e^{-40(\rho^2-1)^4} \sinh\left(5 - 5\rho^{11}\cos(11\theta - 11/\sqrt{2})\right), \tag{5.17}$$

and the boundary condition is $u(\theta, 1) = 1$. Figure 5.2 (right) shows the wall clock time in seconds of our algorithm vs. the degrees of freedom used, verifying the $\mathcal{O}(mn)$

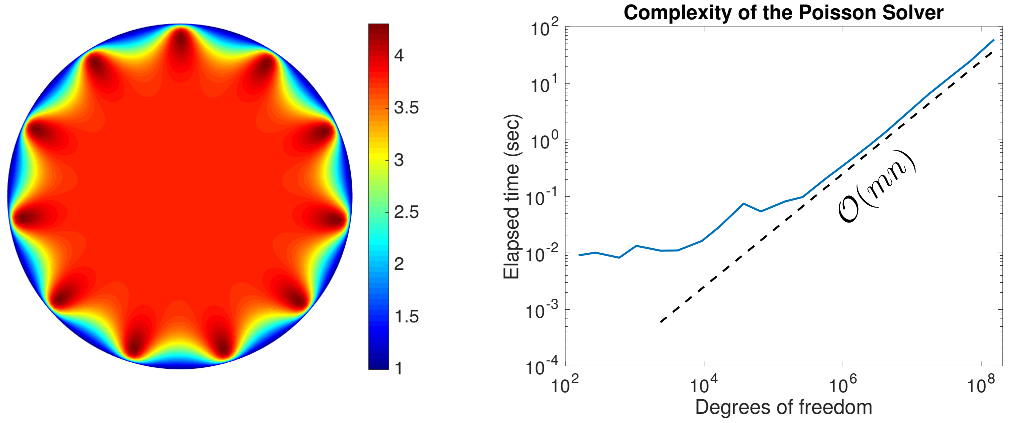


Figure 5.2: Left: Solution to $\nabla^2 u = g$ with boundary condition $u(\theta, 1) = 1$, where g is given in (5.17). Right: Execution (wall clock) time of the Poisson solver as a function of the number of unknowns, $mn/2$. We denote the degrees of freedom by $mn/2$ because this number is used to define the solution u on the disk, where mn unknowns are used on $\tilde{u}$.

complexity of the solver. This includes the cost of computing the coefficients for $\rho^2 g$ explicitly from a low rank approximation to $\rho^2 g$. On a laptop computer with no parallelization,[2] the numerical solution u can be computed using 10^8 degrees of freedom in just under a minute.

[2]Timings performed on a 2015 MacBook Pro using MATLAB 2015b, without explicit parallelization.

CHAPTER 6

CONCLUSIONS

This work has resulted in the creation of a new low rank approximation procedure for functions in polar and spherical geometries. We discuss this development in Chapter 3, where we synthesize the classic double Fourier sphere (DFS) method and its disk analogue with a newly-developed structure-preserving Gaussian elimination (GE) procedure. We re-examine classic observations on essential parity properties for functions in spherical and polar geometries through the more generalized lens of what we have coined *block-mirror-centrosymmetric* (BMC) functions (see Section 3.4). This generalization allows us to enforce parity properties associated with functions on the sphere and disk on function *values*, as opposed to *coefficients*, which is what the structure-preserving GE procedure requires. Through this procedure, we construct adaptive low rank approximations to functions on the sphere and disk that are smooth in their respective domains, near-optimal in their underlying discretizations, and allow for the use of FFT-based algorithms.

We have formulated a suite of algorithms that exploit the convenient properties of low rank approximants with BMC structure. This includes efficient schemes for differentiation, integration, vector calculus, and finding the solution to Poisson's equation, along with many other operations. These algorithms are detailed in Chapters 4 and 5, and they have been implemented in the Spherefun and Diskfun software, which is publicly available as a part of Chebfun (`www.chebfun.org`). Following the precedence set by the Chebfun developers, our software efficiently performs computations to approximately machine precision and is accessed through an interface that is highly intuitive to MATLAB users. As a result, researchers, instructors, and students can now easily explore and effectively solve many mathematical problems associated with functions on the sphere and disk, without concern for the underlying discretization.

REFERENCES

[1] P. AMORE, *Solving the Helmholtz equation for membranes of arbitrary shape: numerical results*, J. Phys. A: Math. Theor., 41 (2008), p. 265206.

[2] K. Atkinson and W. Han, *Spherical Harmonics and Approximations on the Unit Sphere: An Introduction*, Lecture Notes in Mathematics, Springer Berlin Heidelberg, 2012.

[3] R. Bartnik and A. Norton, *Numerical methods for the Einstein equations in null quasi-spherical coordinates*, SIAM J. Sci. Comput., 22 (2000), pp. 917–950.

[4] Z. Battles and L. N. Trefethen, *An extension of MATLAB to continuous functions and operators*, SIAM J. Sci. Comput., 25 (2004), pp. 1743–1770.

[5] J. R. Baumgardner and P. O. Frederickson, *Icosahedral discretization of the Two-Sphere*, SIAM J. Numer. Anal., 22 (1985), pp. 1107–1115.

[6] M. Bebendorf, *Approximation of boundary element matrices*, Numer. Math., 86 (2000), pp. 565–589.

[7] A. Bhatia and E. Wolf, *On the circle polynomials of Zernike and related orthogonal sets*, in Mathematical Proceedings of the Cambridge Philosophical Society, vol. 50, Cambridge Univ Press, 1954, pp. 40–48.

[8] J. P. Boyd, *The choice of spectral functions on a sphere for boundary and eigenvalue problems: A comparison of Chebyshev, Fourier and associated Legendre expansions*, Mon. Wea. Rev., 106 (1978), pp. 1184–1191.

[9] J. P. Boyd, *Chebyshev and Fourier Spectral Methods*, Courier Corporation, 2001.

[10] J. P. Boyd and F. Yu, *Comparing seven spectral methods for interpolation and for solving the Poisson equation in a disk: Zernike polynomials, Logan–Shepp ridge polynomials, Chebyshev–Fourier series, cylindrical Robert functions, Bessel–Fourier expansions, square-to-disk conformal mapping and radial basis functions*, J. Comput. Phys., 230 (2011), pp. 1408–1438.

[11] B. Brügmann, *A pseudospectral matrix method for time-dependent tensor fields on a spherical shell*, J. Comput. Phys., 235 (2013), pp. 216–240.

[12] L. Brutman, *On the Lebesgue function for polynomial interpolation*, SIAM J. Numer. Anal., 15 (1978), pp. 694–704.

[13] J. R. Bunch and B. N. Parlett, *Direct methods for solving symmetric indefinite systems of linear equations*, SIAM J. Numer. Anal., 8 (1971), pp. 639–655.

[14] L. Carlitz, *The product of two ultraspherical polynomials*, in Proceedings of the Glasgow Mathematical Association, vol. 5, Cambridge University Press, 1961, pp. 76–79.

[15] D. Carlson, E. Haynsworth, and T. Markham, *A generalization of the Schur complement by means of the Moore-Penrose inverse*, SIAM J. Appl. Math., 26 (1974), pp. 169–175.

[16] O. A. Carvajal, F. W. Chapman, and K. O. Geddes, *Hybrid symbolic-numeric integration in multiple dimensions via tensor-product series*, in Proceedings of the 2005 International Symposium on Symbolic and Algebraic Computation, ACM, 2005, pp. 84–91.

[17] H.-B. Cheong, *Application of double Fourier series to the shallow-water equations on a sphere*, J. Comput. Phys., 165 (2000), pp. 261–287.

[18] ——, *Double Fourier series on a sphere: Applications to elliptic and vorticity equations*, J. Comput. Phys., 157 (2000), pp. 327–349.

[19] R. Churchill, *Fourier Series and Boundary Value Problems*, McGraw-Hill, 1941.

[20] J. Coiffier, *Fundamentals of Numerical Weather Prediction*, Cambridge University Press, 2011.

[21] T. A. Driscoll, N. Hale, and L. N. Trefethen, eds., *Chebfun Guide*, Pafnuty Publications, Oxford, 2014.

[22] H. Eisen, W. Heinrichs, and K. Witsch, *Spectral collocation methods and polar coordinate singularities*, J. Comput. Phys., 96 (1991), pp. 241–257.

[23] G. E. Fasshauer and L. L. Schumaker, *Scattered data fitting on the sphere*, in Mathematical Methods for Curves and Surface, M. Daehlen, T. Lyche, and L. L. Schumaker, eds., Vanderbilt University Press, Tennessee, 1998.

[24] N. Flyer and G. B. Wright, *A radial basis function method for the shallow water equations on a sphere*, Proc. Royal Soc. A, 465 (2009), pp. 1949–1976.

[25] B. Fornberg, *A pseudospectral approach for polar and spherical geometries*, SIAM J. Sci. Comput., 16 (1995), pp. 1071–1081.

[26] B. Fornberg and N. Flyer, *A Primer on Radial Basis Functions with Applications to the Geosciences*, SIAM, 2015.

[27] B. Fornberg and C. Piret, *A stable algorithm for flat radial basis functions on a sphere*, SIAM J. Sci. Comput., 30 (2007), pp. 60–80.

[28] L. V. Foster and X. Liu, *Comparison of rank revealing algorithms applied to matrices with well defined numerical ranks*, Manuscript, (2006).

[29] P. Godon, *Numerical modeling of tidal effects in polytropic accretion disks*, Astrophys. J., 480 (1997), p. 329.

[30] G. Golub and C. Van Loan, *Matrix Computations*, Johns Hopkins University Press, 2012.

[31] K. M. Gorski, E. Hivon, A. Banday, B. D. Wandelt, F. K. Hansen, M. Reinecke, and M. Bartelmann, *HEALPix: a framework for high-resolution discretization and fast analysis of data distributed on the sphere*, Astrophys. J., 622 (2005), p. 759.

[32] N. Halko, P.-G. Martinsson, and J. A. Tropp, *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, SIAM Rev., 53 (2011), pp. 217–288.

[33] A. Hammerstein, *Über die entwickelung des kernes linearer integralgleichungen nach eigenfunktionen*, Sitzungsberichte Preuss. Akad. Wiss, (1923), pp. 181–184.

[34] W. Heinrichs, *Spectral collocation schemes on the unit disc*, J. Comput. Phys., 199 (2004), pp. 66–86.

[35] P. Henrici, *Barycentric formulas for interpolating trigonometric polynomials and their conjugates*, Numer. Math., 33 (1979), pp. 225–234.

[36] A. R. H. Heryudono and T. A. Driscoll, *Radial basis function interpolation on irregular domains through conformal transplantation*, J. Sci. Comput., 44 (2010), pp. 286–300.

[37] C. Jekeli, *Spherical harmonic analysis, aliasing, and filtering*, Journal of Geodesy, 70 (1996), pp. 214–223.

[38] A. Karageorghis, C. Chen, and Y.-S. Smyrlis, *A matrix decomposition RBF algorithm: Approximation of functions and their derivatives*, Appl. Numer. Math., 57 (2007), pp. 304–319.

[39] R. Kerswell, *Recent progress in understanding the transition to turbulence in a pipe*, Nonlinearity, 18 (2005), p. R17.

[40] A. T. Layton and W. F. Spotz, *A semi-Lagrangian double Fourier method for the shallow water equations on the sphere*, J. Comput. Phys., 189 (2003), pp. 180–196.

[41] V. N. Mahajan and G.-m. Dai, *Orthonormal polynomials in wavefront analysis: analytical solution*, JOSA A, 24 (2007), pp. 2994–3016.

[42] G. Martin, *Transformation Geometry: An Introduction to Symmetry*, Springer, New York, 2012.

[43] J. C. Mason and D. C. Handscomb, *Chebyshev Polynomials*, CRC Press, 2002.

[44] T. Matsushima and P. Marcus, *A spectral method for polar coordinates*, J. Comput. Phys., 120 (1995), pp. 365–374.

[45] P. E. Merilees, *The pseudospectral approximation applied to the shallow water equations on a sphere*, Atmosphere, 11 (1973), pp. 13–20.

[46] M. J. Mohlenkamp, *A fast transform for spherical harmonics*, J. Fourier Anal. Appl., 5 (1999), pp. 159–184.

[47] F. W. Olver, D. W. Lozier, R. F. Boisver, and C. W. Clark, *NIST Handbook of Mathematical Functions*, Cambridge University Press, 2010.

[48] S. Olver and A. Townsend, *A fast and well-conditioned spectral method*, SIAM Rev., 55 (2013), pp. 462–489.

[49] S. A. Orszag, *Fourier series on spheres*, Mon. Wea. Rev., 102 (1974), pp. 56–75.

[50] E. L. Ortiz, *The tau method*, SIAM J. Numer. Anal., 6 (1969), pp. 480–492.

[51] J. Pringle, *Accretion discs in astrophysics*, Ann. Rev. of Astronomy and Astrophysics, 19 (1981), pp. 137–162.

[52] R. Sadourny, *Conservative finite-difference approximations of the primitive equations on quasi-uniform spherical grids*, Mon. Wea. Rev., 100 (1972), pp. 136–144.

[53] E. Schmidt, *Zur theorie der linearen und nichtlinearen integralgleichungen. iii. teil*, Mathematische Annalen, 65 (1908), pp. 370–399.

[54] H. A. Schwarz, *Ueber einige abbildungsaufgaben*, J. für die Reine und Angewandte Mathematik, 70 (1869), pp. 105–120.

[55] E. Serre and J. Pulicani, *A three-dimensional pseudospectral method for rotating flows in a cylinder*, Comput. Fluids, 30 (2001), pp. 491–519.

[56] J. Shen, *Efficient spectral-Galerkin methods IV. Spherical geometries*, SIAM J. Sci. Comput., 20 (1999), pp. 1438–1455.

[57] ——, *A new fast Chebyshev–Fourier algorithm for Poisson-type equations in polar geometries*, Appl. Numer. Math., 33 (2000), pp. 183–190.

[58] J. Sherman and W. J. Morrison, *Adjustment of an inverse matrix corresponding to a change in one element of a given matrix*, Ann. Math. Stat., 21 (1950), pp. 124–127.

[59] W. F. Spotz, M. A. Taylor, and P. N. Swarztrauber, *Fast shallow-water equation solvers in latitude-longitude coordinates*, J. Comput. Phys., 145 (1998), pp. 432–444.

[60] G. W. Stewart, *Updating a rank-revealing ULV decomposition*, SIAM J. Matrix Anal. Appl., 14 (1993), pp. 494–499.

[61] P. N. Swarztrauber, *The approximation of vector functions and their derivatives on the sphere*, SIAM J. Numer. Anal., 18 (1981), pp. 191–210.

[62] M. Taylor, J. Tribbia, and M. Iskandarani, *The spectral element method for the shallow water equations on the sphere*, J. Comput. Phys., 130 (1997), pp. 92–108.

[63] W. Tichy, *Black hole evolution with the BSSN system by pseudospectral methods*, Phys. Rev. D, 74 (2006), pp. 1–10.

[64] A. Townsend, *Computing with functions in two dimensions*, PhD thesis, University of Oxford, 2014.

[65] ——, *Gaussian elimination corrects pivoting mistakes*, arXiv preprint arXiv:1602.06602, (2016).

[66] A. Townsend and S. Olver, *The automatic solution of partial differential equations using a global spectral method*, J. Comput. Phys., 299 (2015), pp. 106–123.

[67] A. Townsend and L. N. Trefethen, *An extension of Chebfun to two dimensions*, SIAM J. Sci. Comput., 35 (2013), pp. C495–C518.

[68] ——, *Gaussian elimination as an iterative algorithm*, SIAM News, 46 (2013).

[69] ——, *Continuous analogues of matrix factorizations*, Proc. Royal Soc. A, 471 (2015), pp. 1–21.

[70] A. Townsend, H. Wilber, and G. B. Wright, *Computing with functions in spherical and polar geometries I. The sphere*, SIAM J. Sci. Comput., (2016). to appear.

[71] A. Townsend, H. Wilber, and G. B. Wright, *Computing with functions in spherical and polar geometries II. The disk*, SIAM J. Sci. Comput., (2016). submitted.

[72] L. N. Trefethen, *Spectral methods in MATLAB*, SIAM, 2000.

[73] ——, *Householder triangularization of a quasimatrix*, IMA J. Numer. Anal., (2009), p. drp018.

[74] ——, *Approximation theory and approximation practice*, SIAM, 2013.

[75] W. F. Trench, *Characterization and properties of matrices with generalized symmetry or skew symmetry*, Lin. Alg. App., 377 (2004), pp. 207–218.

[76] M. Tygert, *Fast algorithms for spherical harmonic expansions, III*, J. Comput. Phys., 229 (2010), pp. 6181–6192.

[77] G. M. Vasil, K. J. Burns, D. Lecoanet, S. Olver, B. P. Brown, and J. S. Oishi, *Tensor calculus in polar coordinates using Jacobi polynomials*, arXiv preprint arXiv:1509.07624, (2015).

[78] Z. von F, *Beugungstheorie des schneidenver-fahrens und seiner verbesserten form, der phasenkontrastmethode*, Physica, 1 (1934), pp. 689–704.

[79] D. L. Williamson, J. B. Drake, J. J. Hack, R. Jakob, and P. N. Swarztrauber, *A standard test set for numerical approximations to the shallow water equations in spherical geometry*, J. Comput. Phys., 102 (1992), pp. 211–224.

[80] G. B. Wright, N. Flyer, and D. A. Yuen, *A hybrid radial basis function–pseudospectral method for thermal convection in a 3-D spherical shell*, Geochem. Geophys. Geosyst., 11 (2010), p. Q07003.

[81] G. B. Wright, M. Javed, H. Montanelli, and L. N. Trefethen, *Extension of Chebfun to periodic functions*, SIAM J. Sci. Comput., 37 (2015), pp. C554–C573.

[82] S. Y. K. Yee, *Studies on Fourier series on spheres*, Mon. Wea. Rev., 108 (1980), pp. 676–678.

[83] ——, *Solution of Poisson's equation on a sphere by truncated double Fourier series*, Mon. Wea. Rev., 109 (1981), pp. 501–505.

APPENDIX A

PROPERTIES OF BMC FUNCTIONS

A.1 Linear Algebra for BMC Matrices

Given a BMC function $\tilde{f}(\xi, \eta)$ (see Def. 3.3), a discretization of $\tilde{f}$ on a $2n \times 2n$ grid that is symmetric about $\xi = 0$ and $\eta = 0$ results in a matrix $M \in \mathbb{R}^{2n\times 2n}$ that is said to have BMC symmetry.[1] Using J_n to denote the $n \times n$ row exchange operator,[2] the matrix M can be written using $n \times n$ blocks as

$$M = \begin{bmatrix} B & C \\ J_n C & J_n B \end{bmatrix}. \tag{A.1}$$

We can also write M as the product

$$M = \begin{bmatrix} I_n & 0 \\ 0 & J_n \end{bmatrix} \underbrace{\begin{bmatrix} B & C \\ C & B \end{bmatrix}}_{A}, \tag{A.2}$$

where I_n is the $n \times n$ identity matrix. The matrix A has a special structure, and by writing M in terms of A, we observe that BMC matrices are closely connected to a broader class of matrices known as R-symmetric matrices.

Definition A.1 (R-symmetric matrices)**.** *A real matrix R is said to be involuntary if $R^{-1} = R$. Given an involuntary matrix $R \in \mathbb{R}^{2n\times 2n}$, a matrix $A \in \mathbb{R}^{2n\times 2n}$ is called*

[1]Here, we consider BMC matrices with even dimensions for the sake of simplicity in exposition, and note that similar, but slightly more complicated results hold for BMC matrices with odd dimensions.

[2]The entries of J_n are 1 along its main antidiagonal, and zero elsewhere.

R–symmetric if $RAR = A$.

The matrix A in (A.2) is R-symmetric with the following block-centrosymmetric matrix:

$$R = \begin{bmatrix} 0 & I_n \\ I_n & 0 \end{bmatrix}. \tag{A.3}$$

We refer to R as a *2-by-2 block exchange* matrix, and so A is said to be *block exchange symmetric* (BES). In [75], several properties of R-symmetric, and therefore BES, matrices are discussed. By relating BMC matrices to BES matrices via (A.2), these results can be used to understand certain properties of BMC matrices. Here, we use properties of BES matrices to derive the SVD for BMC matrices.

First, observe that the spectral decomposition of the block exchange matrix R in (A.3) is

$$R = \begin{bmatrix} P & Q \end{bmatrix} \begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix} \begin{bmatrix} P^T \\ Q^T \end{bmatrix}, \quad P = \frac{1}{\sqrt{2}} \begin{bmatrix} I_n \\ I_n \end{bmatrix}, \quad Q = \frac{1}{\sqrt{2}} \begin{bmatrix} I_n \\ -I_n \end{bmatrix}.$$

Define $A_{PP} = P^T AP$, and $A_{QQ} = Q^T AQ$, observing that these simplify to $A_{PP} = B + C$, and $A_{QQ} = B - C$, respectively. Let the SVD for each of these matrices be

$$A_{PP} = U_P \Sigma_P V_P^T, \qquad A_{QQ} = U_Q \Sigma_Q V_Q^T.$$

Theorem 11 in [75] shows that the SVD for the R-symmetric matrix A is then given by

$$A = \begin{bmatrix} PU_P & QU_Q \end{bmatrix} \begin{bmatrix} \Sigma_P & 0 \\ 0 & \Sigma_Q \end{bmatrix} \begin{bmatrix} V_P^T P^T \\ V_Q^T Q^T \end{bmatrix}, \tag{A.4}$$

which can be re-written as

$$A = \begin{bmatrix} \frac{1}{\sqrt{2}} U_P & \frac{1}{\sqrt{2}} U_Q \\ \frac{1}{\sqrt{2}} U_P & -\frac{1}{\sqrt{2}} U_Q \end{bmatrix} \begin{bmatrix} \Sigma_P & 0 \\ 0 & \Sigma_Q \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} V_P^T & \frac{1}{\sqrt{2}} V_P^T \\ \frac{1}{\sqrt{2}} V_Q^T & -\frac{1}{\sqrt{2}} V_Q^T \end{bmatrix}. \tag{A.5}$$

We now consider the BMC matrix M, and observe that by substituting (A.5) into (A.2),

$$M = \begin{bmatrix} I_n & 0 \\ 0 & J_n \end{bmatrix} \begin{bmatrix} B & C \\ C & B \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} U_P & \frac{1}{\sqrt{2}} U_Q \\ \frac{1}{\sqrt{2}} J_n U_P & -\frac{1}{\sqrt{2}} J_n U_Q \end{bmatrix} \begin{bmatrix} \Sigma_P & 0 \\ 0 & \Sigma_Q \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} V_P^T & \frac{1}{\sqrt{2}} V_P^T \\ \frac{1}{\sqrt{2}} V_Q^T & -\frac{1}{\sqrt{2}} V_Q^T \end{bmatrix}. \tag{A.6}$$

Since the matrices

$$\frac{1}{\sqrt{2}} \begin{bmatrix} U_P & U_Q \\ J_n U_P & -J_n U_Q \end{bmatrix}, \qquad \begin{bmatrix} \frac{1}{\sqrt{2}} V_P^T & \frac{1}{\sqrt{2}} V_P^T \\ \frac{1}{\sqrt{2}} V_Q^T & -\frac{1}{\sqrt{2}} V_Q^T \end{bmatrix}$$

are unitary, (A.6) must be the SVD of M. Written in a more revealing form, this is

$$M = \frac{1}{2} \sum_{k=1}^{r} (\sigma_P)_k \begin{bmatrix} U_P(:,k) \\ JU_P(:,k) \end{bmatrix} \begin{bmatrix} V_P(k,:) & V_P(k,:) \end{bmatrix} \tag{A.7}$$

$$+\frac{1}{2} \sum_{k=1}^{s} (\sigma_Q)_k \begin{bmatrix} U_Q(:,k) \\ -JU_Q(:,k) \end{bmatrix} \begin{bmatrix} V_Q(k,:) & -V_Q(k,:) \end{bmatrix},$$

where $r = \text{rank}(B+C)$ and $s = \text{rank}(B-C)$. In this form, it is clear that each rank 1 term in M has BMC symmetry. Furthermore, if we view M as the discretization of the BMC function $\tilde{f}$, then the SVD separates the decomposition into two matrices: the even-symmetric part of $\tilde{f}$ is associated with the SVD of A_{PP}, and the odd-antisymmetric part of $\tilde{f}$ is associated with the SVD of A_{QQ}. We can find the SVD of M by operating independently on A_{PP} and A_{QQ}.

A.2 The SVD for BMC Functions

The results for the discrete case parallel results in the continuous setting discussed in Section 3.3, and this can be used to find an explicit expression of the SVD for BMC functions. Here, we consider the BMC-I function $\tilde{f}(\lambda, \theta)$ defined in (3.1), noting that an equivalent result holds for a BMC-II function derived from the disk via (3.4). Note that there is a significant distinction between the SVD of the BMC function $\tilde{f}$ and the weighted SVD of the function f associated with $\tilde{f}$ and defined on the sphere. A discussion of the weighted SVD for f can be found in Section 4.2.5.

If $\tilde{f}(\lambda, \theta)$ is a BMC function with $(\lambda, \theta) \in [-\pi, \pi]^2$ that is Lipschitz continuous in both variables, then the SVD of $\tilde{f}$ converges absolutely and uniformly to $\tilde{f}$. Then, $\tilde{f}$ can be written as [53]

$$\tilde{f}(\lambda, \theta) = \sum_{j=1}^{\infty} \sigma_j u_j(\theta) v_j(\lambda), \qquad (\lambda, \theta) \in [-\pi, \pi]^2, \tag{A.8}$$

where $\sigma_1, \sigma_2, \ldots$ is a nonnegative and nonincreasing sequence called the *singular values*, $\{u_1, u_2, \ldots\}$ and $\{v_1, v_2, \ldots\}$ are two sets of orthonormal functions on $L^2([-\pi, \pi])$, as discussed in Section 2.2.2.

We may equivalently write the SVD of $\tilde{f}$ as a decomposition,

$$\tilde{f} = U \Sigma V^T, \tag{A.9}$$

where $U(\theta) = [u_1(\theta) \,|\, u_2(\theta) \,|\, \cdots]$ and $V(\lambda) = [v_1(\lambda) \,|\, v_2(\lambda) \,|\, \cdots]$ are $[-\pi, \pi] \times \infty$ quasimatrices, and $\Sigma = \operatorname{diag}(\sigma_1, \sigma_2, \ldots)$ is a diagonal matrix.

By (3.15), we can express $\tilde{f}$ as a sum involving $f^+ = p + q$ and $f^- = p - q$, where p and q are defined in (3.1). Since these functions are also Lipschitz continuous in both variables, we can write the SVDs of f^+ and f^- as $f^+ = U^+ \Sigma^+ (V^+)^T$ and $f^- = U^- \Sigma^- (V^-)^T$, respectively. Here, the columns of U^+, U^-, V^+, and V^- are orthonormal with respect to the $L^2([-\pi, \pi])$ inner-product. Letting $\mathcal{J}$ be the reflection operator defined in Lemma 3.2 and carefully amalgamating the SVDs of f^+ and f^- together as in (A.6), we obtain the decomposition

$$\begin{bmatrix} p & q \\ \mathcal{J}q & \mathcal{J}p \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{2}}{2} U^+ & \frac{\sqrt{2}}{2} U^- \\ \frac{\sqrt{2}}{2} \mathcal{J}(U^+) & -\frac{\sqrt{2}}{2} \mathcal{J}(U^-) \end{bmatrix} \begin{bmatrix} \frac{1}{2}\Sigma^+ & \\ & \frac{1}{2}\Sigma^- \end{bmatrix} \begin{bmatrix} \frac{\sqrt{2}}{2} V^+ & \frac{\sqrt{2}}{2} V^- \\ \frac{\sqrt{2}}{2} V^+ & -\frac{\sqrt{2}}{2} V^- \end{bmatrix}^T, \tag{A.10}$$

which is the SVD of $\tilde{f}$, up to a reordering of the singular values and corresponding singular vectors so that the singular values are in nonincreasing order. By uniqueness of the SVD, (A.10) is the same as (A.8). Therefore, any rank 1 term in (A.8) must take the form

$$\sigma_j u_j v_j = \sigma_j \begin{bmatrix} u \\ \pm\mathcal{J}(u) \end{bmatrix} \begin{bmatrix} v & \pm v \end{bmatrix} = \sigma_j \begin{bmatrix} uv & \pm uv \\ \pm\mathcal{J}(uv) & \mathcal{J}(uv) \end{bmatrix},$$

and is itself a BMC function. Since a sum of BMC functions is also a BMC function, truncating the SVD of $\tilde{f}$ in (A.8) after K terms gives a rank K approximation to $\tilde{f}$ that preserves BMC structure.

However, the SVD does not always preserve BMC-I (or BMC-II) structure, since the rank 1 terms need not be constant for $\theta = 0$, $\theta = \pm\pi$. In the case where $\tilde{f}(\cdot, 0) = \tilde{f}(\cdot, \pm\pi) = 0$, BMC-I structure is preserved. It is possible, therefore, to use a related SVD decomposition to construct low rank approximations to $\tilde{f}$. First, apply an initial modification step that subtracts a rank 1 term from $\tilde{f}$ to produce $\tilde{f}_{proj}$, which is a BMC function with zero values along $\theta = 0$ and $\theta = \pm\pi$. This can be done by choosing

$$\tilde{f}_{proj}(\lambda, \theta) \leftarrow \tilde{f}(\lambda, \theta) - \tilde{f}(\lambda^*, \theta) \tag{A.11}$$

for some $\lambda^* \in [-\pi, \pi]$. Finding the SVD of $\tilde{f}_{proj}$ and adding it to the subtracted rank 1 term in (A.11) gives a low rank approximation to $\tilde{f}$.

A natural question is whether this results in a better scheme for constructing low rank approximants than the structure-preserving GE procedure in Chapter 3. When $\tilde{f}$ is nonzero along $\theta = 0$ or $\theta = \pm\pi$, both procedures involve equivalent initial steps, and then both procedures operate on $\tilde{f}_{proj}$. The SVD does this optimally with respect to the L_2 norm, but it is computationally expensive, requiring $\mathcal{O}(N^3)$ operations, where N is the number of samples of $\tilde{f}$ required to resolve $\tilde{f}$ to some desired tolerance. For reasonably smooth functions, the structure-preserving GE procedure is more efficient at a cost of only $\mathcal{O}(K^3 + K^2(m+n))$, where K is the numerical rank of the function (see Section 3.2), and it produces *near-optimal* approximants to $\tilde{f}$. As discussed in Section 3.6, we observe convergence properties in the L_2 norm that asymptotically identical for GE and the SVD.

APPENDIX B

THE ULTRASPHERICAL SPECTRAL METHOD

In [48], a new spectral method is developed for solving linear ordinary differential equations (ODEs) of arbitrary differential order. It uses recurrence relationships between the ultraspherical polynomials and their derivatives [47, (18.9.19), (18.9.21)] to develop sparse representations for differential operators. As the order of the differential operator increases, the ultraspherical parameter changes, too. For this reason, the method also includes tridiagonal *conversion* operators that are based on ultraspherical polynomial recurrence relationships [47, (18.9.7), (18.9.9)]. Use of these operators leads to linear systems involving almost-banded matrices, meaning that they are banded everywhere except for the first K rows, which are dense (an example is depicted in Figure B.1). Here, K is the number of boundary conditions prescribed with the differential equation. The structure of this system facilitates the use of an algorithm based on QR factorization through Givens rotations that is numerically stable, regardless of the system's size or the differential order of the ODE.

In this section, we describe the ultraspherical spectral method by using it to discretize a general linear second-order ODE. We apply these results in Section 5.2 to formulate an optimal complexity solver for Poisson's equation on the disk.

Discretization

Consider the following second-order linear ODE:

$$\frac{\partial^2 u}{\partial x^2} + a(x)\frac{\partial u}{\partial x} + b(x)u = f, \quad x \in [-1,1], \tag{B.1}$$

with boundary conditions prescribed as $u(-1) = \alpha,\ u(1) = \beta$. If u is continuous with bounded variation on $[-1,1]$, then the solution $u(x)$ can be represented by a Chebyshev series,

$$u(x) = \sum_{k=0}^{\infty} u_k T_k(x), \tag{B.2}$$

where $T_k(x)$ is the Chebyshev polynomial of degree k. The solution to (B.1) is completely characterized by the set of Chebyshev coefficients, $\{u_k\}_{k=0}^{\infty}$, and this is the form of the solution we will seek. Let $\mathbf{u}$ be an infinite vector of these coefficients, i.e., $\mathbf{u} = [u_0,\, u_1,\, \cdots]^T$. Likewise, define the infinite vector $\mathbf{f}$ as the vector of Chebyshev coefficients for $f(x)$.

To discretize (B.1), we must define differentiation operators D_1 and D_2 that act on $\mathbf{u}$. By using recurrence relations between the ultraspherical polynomials, we can formulate D_1 and D_2 so that they are sparse.

The ultraspherical polynomials are orthogonal polynomials on the interval $[-1,1]$. They are denoted by $C_k^{(\lambda)}$, where each set $C_k^{(\lambda)}$ is orthogonal with respect to the weight function $(1-x^2)^{\lambda-1/2}$. In addition to the Chebyshev polynomials, we will make use of the ultraspherical polynomials where λ is chosen as a positive integer. These can be defined uniquely via the normalization of the leading coefficient, so that

$$C_k^{(\lambda)}(x) = \frac{2^k(\lambda)_k}{k!}x^k + \mathcal{O}(x^{k-1}), \quad x \in [-1,1], \quad \lambda = 1, 2, \ldots, \tag{B.3}$$

where $(\lambda)_k$ is the Pochhammer symbol, i.e.,

$$(\lambda)_k = \frac{(\lambda+k-1)!}{(\lambda-1)!}.$$

The Chebyshev polynomials of the second kind are produced when $\lambda = 1$. A useful recurrence relation exists between the Chebyshev polynomials of the first and second kinds, and there is a generalized recurrence relation between the ultraspherical polynomials with $\lambda \in \mathbb{Z}^+$ [47, (18.9.19), (18.9.21)]. These are respectively given as follows:

$$\frac{dT_k}{dx} = \begin{cases} kC^{(1)}_{k-1}, & k \geq 1, \\ 0, & k = 0, \end{cases} \qquad \frac{dC^{(\lambda)}_k}{dx} = \begin{cases} 2\lambda C^{(\lambda+1)}_{k-1}, & k \geq 1, \\ 0, & k = 0. \end{cases} \tag{B.4}$$

Applying (B.4) to (B.2), we find

$$\frac{du}{dx} = \sum_{k=1}^{\infty} ku_k C^{(1)}_{k-1}(x), \qquad \frac{d^\lambda u}{dx^\lambda} = 2^{\lambda-1}(\lambda-1)! \sum_{k=\lambda}^{\infty} ku_k C^{(\lambda)}_{k-\lambda}(x), \tag{B.5}$$

where $\{u_k\}$ are the Chebyshev coefficients in (B.2), and we have applied the generalized recurrence relation repeatedly to find $d^\lambda u/dx^\lambda$. We use these to create the differentiation operators D_1 and D_λ. The first equation in (B.5) gives the operator

$$D_1 = \begin{bmatrix} 0 & 1 & & & \\ & 0 & 2 & & \\ & & 0 & 3 & \\ & & & \ddots & \ddots \\ & & & & \ddots \end{bmatrix}, \tag{B.6}$$

and the second equation in (B.5) gives

$$D_\lambda = 2^{\lambda-1}(\lambda-1)! \begin{bmatrix} \overbrace{0 \; \cdots \; 0}^{\lambda \text{ times}} & \lambda & & & \\ & \ddots & \lambda+1 & & \\ & & \ddots & \lambda+2 & \\ & & & \ddots & \ddots \\ & & & & \ddots \end{bmatrix}. \tag{B.7}$$

When D_1 is applied to $\mathbf{u}$, the result is an infinite vector of the $C^{(1)}$ coefficients of $u'(x)$. Similarly, the λ-order differentation operator D_λ finds the $C^{(\lambda)}$ coefficients of the λth derivative of $u(x)$.

For the purposes of discretizing (B.1), we require the differentiation operator D_2. Using (B.7), we have

$$D_2 = 2 \begin{bmatrix} 0 & 0 & 2 & & & \\ & & 0 & 3 & & \\ & & & 0 & 4 & \\ & & & & \ddots & \ddots \\ & & & & & \ddots \end{bmatrix}. \tag{B.8}$$

At this point, consider a simplified example of the differential equation (B.1) where $a(x) = b(x) = 1$. Discretizing this equation with the given differentiation operators, we find that

$$\underbrace{D_2 u}_{C^{(2)}\text{ basis}} + \underbrace{D_1 u}_{C^{(1)}\text{ basis}} + \underbrace{u}_{\text{Chebyshev basis}} = \underbrace{f}_{\text{Chebyshev basis}}. \tag{B.9}$$

Each term in (B.9) is an expression of coefficients associated with various ultraspherical basis expansions. To represent every term using the same basis expansion, *conversion operators* are used. Using the following recurrence relationships [47, (18.9.7), (18.9.9)],

$$T_k = \begin{cases} \frac{1}{2}\left(C_k^{(1)} - C_{k-2}^{(1)}\right), & k \geq 2, \\ \frac{1}{2} C_1^{(1)}, & k = 1, \\ C_0^{(1)} & k = 0, \end{cases} \qquad C_{(k)}^{\lambda} = \begin{cases} \frac{\lambda}{\lambda+k}\left(C_k^{(\lambda+1)} - C_{k-2}^{(\lambda+1)}\right), & k \geq 2, \\ \frac{\lambda}{\lambda+1} C_1^{(\lambda+1)}, & k = 1, \\ C_0^{(\lambda+1)} & k = 0, \end{cases} \tag{B.10}$$

we derive the following conversion operators:

$$S_0 = \begin{bmatrix} 1 & 0 & -\frac{1}{2} & & & \\ & \frac{1}{2} & 0 & -\frac{1}{2} & & \\ & & \frac{1}{2} & 0 & -\frac{1}{2} & \\ & & & \ddots & & \ddots \end{bmatrix}, \quad S_\lambda = \begin{bmatrix} 1 & 0 & -\frac{\lambda}{\lambda+2} & & & \\ & \frac{\lambda}{\lambda+1} & 0 & -\frac{\lambda}{\lambda+3} & & \\ & & \frac{\lambda}{\lambda+2} & 0 & -\frac{\lambda}{\lambda+4} & \\ & & & \ddots & & \ddots \end{bmatrix}. \tag{B.11}$$

We can use S_0, for example, to convert a vector of Chebyshev coefficients into $C^{(1)}$ coefficients: since $\mathbf{u}$ contains the Chebyshev coefficients of $u(x)$, $S_0\mathbf{u}$ expresses the $C^{(1)}$ coefficients for $u(x)$. Similarly, S_λ is used to convert $C^{(\lambda)}$ coefficients into $C^{(\lambda+1)}$ coefficients.

Using these conversion operators, the simple case of (B.1) where $a(x) = b(x) = 1$ can now be expressed in the $C^{(2)}$ basis as follows:

$$(D_2 + S_1 D_1 + S_1 S_0)\mathbf{u} = S_1 S_0 \mathbf{f}.$$

Now suppose that $a(x)$ and $b(x)$ are arbitrary continuous functions of bounded variation on the interval $[-1, 1]$. If $b(x)$ is sufficiently smooth [74, Ch. 7, 8], the Chebyshev coefficients for $b(x)$ decay rapidly. Since we are ultimately interested in approximating b to within some desired tolerance ϵ, we consider the truncated Chebyshev expansion of $b(x)$ to m terms, where m satisfies the following:

$$\left\| b(x) - \sum_{k=0}^{m-1} b_k T_k(x) \right\|_{L^\infty([-1,1])} < \epsilon.$$

Define $\mathbf{b}$ as an infinite vector where the first m entries are the first m Chebyshev coefficients for $b(x)$, and the remaining entries are zeros.

The product $b(x)u(x)$ can be expressed as a Chebyshev series,

$$b(x)u(x) = \sum_{j=0}^{\infty} \sum_{k=0}^{\infty} b_j u_k T_j(x) T_k(x) = \sum_{k=0}^{\infty} c_k T_k(x), \tag{B.12}$$

and we seek an efficient method for deriving the vector $\mathbf{c} = [c_0,\, c_1,\, c_2,\, \cdots]^T$. In [48], it is shown that

$$c_k = \begin{cases} b_0 u_0 + \frac{1}{2}\sum_{j=1}^{\infty} b_j u_j, & k = 0, \\ \frac{1}{2}\sum_{j=0}^{k-1} b_{k-j}u_j + b_0 u_k + \frac{1}{2}\sum_{j=1}^{\infty} b_j u_{j+k} + \frac{1}{2}\sum_{j=0}^{\infty} b_{j+k}u_j, & k \geq 1. \end{cases} \tag{B.13}$$

To reveal the structure inherent in (B.13) more clearly, we write it as a product between the vector $\mathbf{u}$ and the sum of a Toeplitz and almost Hankel matrix:

$$M_{0[b]}\mathbf{u} = \frac{1}{2}\left(\begin{bmatrix} 2b_0 & b_1 & b_2 & b_3 & \cdots \\ b_1 & 2b_0 & b_1 & b_2 & \ddots \\ b_2 & b_1 & 2b_0 & b_1 & \ddots \\ b_3 & b_2 & b_1 & 2b_0 & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & \cdots \\ b_1 & b_2 & b_3 & b_4 & \iddots \\ b_2 & b_3 & b_4 & b_5 & \iddots \\ b_3 & b_4 & b_5 & b_6 & \iddots \\ \vdots & \iddots & \iddots & \iddots & \iddots \end{bmatrix} \right) \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \\ u_4 \\ \vdots \end{bmatrix}. \tag{B.14}$$

The matrix $M_{0[b]}$ is an operator for multiplying by $b(x)$. The subscript relates to the fact that $M_{0[b]}$ is an operator on vectors of Chebyshev coefficients. The matrix $M_{0[b]}$ is banded with a bandwidth μ, where μ is the number of essentially nonzero Chebyshev coefficients needed to approximate $b(x)$ within some specified tolerance.

To develop an operator for multiplying by $a(x)$ as in (B.1), recall that $D_1\mathbf{u}$ in (B.9) gives the $C^{(1)}$ coefficients for $u'(x)$. For multiplication to be meaningful, we must express $a(x)$ as an expansion in the $C^{(1)}$ basis. Since this method can be generalized to ODEs of any differential order, we consider the more general case, replacing $C^{(1)}$ with $C^{(\lambda)}$. Given two functions expressed as

$$a(x) = \sum_{j=0}^{\infty} a_j C_j^{(\lambda)}(x), \quad v(x) = \sum_{k=0}^{\infty} v_k C_k^{(\lambda)}(x),$$

we seek the coefficients of the product

$$a(x)v(x) = \sum_{j=0}^{\infty}\sum_{k=0}^{\infty} a_j v_k C_j^{(\lambda)}(x) C_k^{(\lambda)}(x). \tag{B.15}$$

Applying a linearization formula given by Carlitz [14], it can be shown that $a(x)v(x)$ is equivalent to applying a multiplication operator $M_{\lambda[a]}$ to the vector $\mathbf{v}$, where the (j,k)th entry of $M_{\lambda[a]}$ is given by

$$M_{\lambda[a]j,k} = \sum_{s=\max(0,k-j)}^{k} a_{2s+j-k} c_s^{(\lambda)}(k, 2s+j-k), \qquad j,k \geq 0, \tag{B.16}$$

and

$$c_s^{(\lambda)}(j,k) = \frac{j+k+\lambda-2s}{j+k+\lambda-s} \frac{(\lambda)_s(\lambda)_{j-s}(\lambda)_{k-s}}{s!(j-s)!(k-s)!} \frac{(2\lambda)_{j+k-s}}{(\lambda)_{j+k-s}} \frac{(j+k-2s)!}{(2\lambda)_{j+k-2s}}. \tag{B.17}$$

A numerically stable alternative to (B.17) is given in [48], as well as a recurrence relation enabling computational efficiency in the calculation.

Using the operators defined by (B.13) and (B.17), we can now discretize a generalized expression of (B.1) in the following way:

$$\frac{\partial^2 u}{\partial x^2} + a(x)\frac{\partial u}{\partial x} + b(x)u = f \longrightarrow \underbrace{\left(D_2 + S_1 M_{1[a]} D_1 + S_1 S_0 M_{0[b]}\right)}_{=L} \mathbf{u} = S_1 S_0 \mathbf{f}. \tag{B.18}$$

The differential operator L is a banded matrix, with a bandwidth determined by the maximum number of nonzero entries in $\mathbf{a}$ or $\mathbf{b}$, or the differential order of the equation, if it is larger. The discretization of the ODE (B.1) can be written compactly as the linear system $L\mathbf{u} = S_1 S_0 \mathbf{f}$.

The boundary conditions must also be included in the discretization. The boundary conditions are given by $u(-1) = \alpha$ and $u(1) = \beta$. Using (B.2) and observing that for all k, $T_k(-1) = (-1)^k$, and $T_k(1) = 1$, we find

$$u(-1) = \sum_{k=0}^{\infty} u_k (-1)^k = \alpha, \qquad u(1) = \sum_{k=0}^{\infty} u_k = \beta.$$

These two equations can be expressed succinctly as follows:

$$\begin{bmatrix} 1 & -1 & 1 & -1 & \cdots \\ 1 & 1 & 1 & 1 & \cdots \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ \vdots \end{bmatrix} = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}. \tag{B.19}$$

To practically implement (B.18), the system is truncated to form an $n \times n$ linear

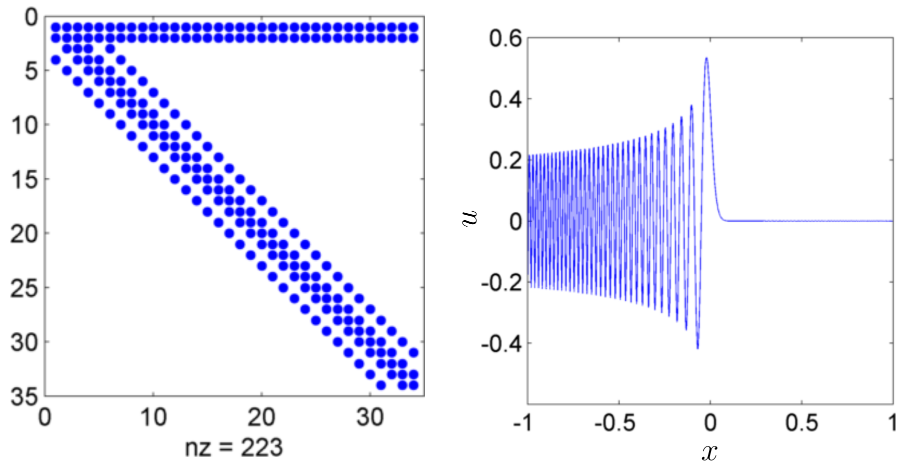


Figure B.1: Left: The left-hand side of the $n \times n$ linear system representing the discretization of the boundary value problem, $\partial^2 u/\partial x^2 - xu = 0$, $u(-1) = \mathrm{Ai}(-\sqrt[3]{1/\varepsilon})$, $u(1) = \mathrm{Ai}(\sqrt[3]{1/\varepsilon})$, where $\varepsilon = 1 \times 10^{-5}$, and Ai are the Airy functions. This is called Airy's equation. Here, $n = 34$ and there are 223 nonzero entries. The two dense upper rows are the result of the boundary conditions. The final system used to compute the solution to machine precision will be similarly sparse, but of size $n = 258$. Right: The solution to the boundary value problem. The L_2 norm of the error for the solution is $\mathcal{O}(10^{-14})$.

system. To include the boundary conditions, we use the *boundary bordering* technique [50]. Observing that D_2 contains only zeros in its last two rows, this method proceeds by replacing the last two rows of the system in (B.18) with (B.19). We also permute these rows to the first two rows of the new system, so that the final linear system takes the form

$$\begin{bmatrix} 1 & -1 & 1 & -1 & \cdots & (-1)^{n-1} \\ 1 & 1 & 1 & 1 & \cdots & 1 \\ & & & & & \\ & & PL & & & \\ & & & & & \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{n-2} \\ u_{n-1} \end{bmatrix} = \begin{bmatrix} \alpha \\ \beta \\ \\ P(S_0 S_1 \mathbf{f}) \\ \\ \end{bmatrix}, \qquad \text{(B.20)}$$

where P is the projection operator that removes the final two rows (entries) of the matrix (vector) it is operating on. The resulting system of equations is almost-banded. By this, we mean that excepting the top two rows, which are dense, the system is banded. An example of the structure is given in Figure B.1. There are efficient and numerically stable ways to solve such a system, such as the adaptive QR algorithm described in [48]. Solving (B.20) yields the vector $\mathbf{u}_n$, which contains approximations to the first n Chebyshev coefficients for $u(x)$.

The ultraspherical spectral method has been implemented within Chebfun as a part of a generalized solver for linear ODEs with variable coefficients, and it is also crucial to a new method that automates the solving of a large class of linear PDEs [66]. In Diskfun, the ultraspherical spectral method is implemented within the `poisson` command to formulate an optimal complexity solver for Poisson's equation on the disk (see Section 5.2).