A NONLINEAR OPTIMIZATION PROCEDURE FOR GENERALIZED GAUSSIAN QUADRATURES

JAMES BREMER[*,§], ZYDRUNAS GIMBUTAS[†], AND VLADIMIR ROKHLIN[‡]

Abstract. We present a new nonlinear optimization procedure for the computation of generalized Gaussian quadratures for a broad class of square integrable functions on intervals. While some of the components of this algorithm have been previously published, we present a simple and robust scheme for the determination of a sparse solution to an underdetermined nonlinear optimization problem which replaces the continuation scheme of the previously published works. The new algorithm successfully computes generalized Gaussian quadratures in a number of instances in which the previous algorithms fail. Four applications of our scheme to computational physics are presented: the construction of discrete plane wave expansions for the Helmholtz Green's function, the design of linear array antennae, the computation of a quadrature for the discretization of Laplace boundary integral equations on certain domains with corners, and the construction of quadratures for the discretization of Laplace and Helmholtz boundary integral equations on smooth surfaces.

1. Introduction

In this paper, we describe a new numerical procedure for the construction of quadrature rules for functions of the form

$$\sum_{j=1}^{2n} \alpha_j \phi_j(x), \tag{1.1}$$

where $\phi_1, \ldots, \phi_{2n}$ are more-or-less arbitrary square integrable functions on an interval $[a, b]$. A $2n$-point quadrature formula for functions of this form is known as a generalized Chebyshev quadrature rule, while an n-point quadrature rule is said to be a generalized Gaussian quadrature. In most cases, the algorithm of this paper produces Gaussian or very nearly Gaussian quadrature rules for collections of functions of the form (1.1).

Classical quadrature techniques, such as Gaussian quadratures, substitution, and subtraction of singularities, often perform poorly or even fail to converge entirely when the functions ϕ_j in (1.1) exhibit more than one type of singular behavior. In this case, the number of quadrature nodes required by classical approaches generally depends on the *form* of the singularities of the functions ϕ_j. This is in stark contrast to generalized quadrature rules, whose size depends on the *dimension* of the space of integrands under consideration.

Integrals of the form (1.1) occur commonly in computational physics. For instance, [8] describes a scheme for accelerating fast multipole methods for the Helmholtz equation at low frequencies based on the integral representation

$$\frac{e^{i\omega r}}{r} = \int_0^\infty e^{-z\sqrt{\xi^2-\omega^2}} J_0(\xi\sqrt{x^2+y^2}) \frac{\xi}{\sqrt{\xi^2-\omega^2}}\, d\xi \tag{1.2}$$

of the Helmholtz Green's function. The scheme requires the construction of quadratures which hold for ranges of the parameters x, y, z and ω. The integrand in (1.2) is both singular and oscillatory with its precise behavior varying widely with different values of these parameters. As a result,

*Department of Mathematics, University of California, Davis, Davis CA 95616 USA.
†Courant Institute of Mathematical Sciences, New York University, 251 Mercer Street, New York, NY 10012 USA.
‡Department of Computer Science, Yale University, New Haven, CT 06520 USA.
§Corresponding author. E-mail address: bremer@math.ucdavis.edu.

Key words and phrases. Quadrature, gerneralized quadrature, boundary integral equations, Laplace's equation, Helmholtz equation, planewave expansions, antennae.

classical techniques are highly inefficient in this instance. We describe in detail the construction of efficient quadrature rules for the discretization of (1.2) in Section 5.1 of this paper.

A similar problem arises in the design of linear array antennae. The synthesis of an antenna system is a two-step process. First, a continuous current distribution which gives rise to a desired radiation pattern is determined. Then, a discrete approximation of the current distribution is calculated. The cost and complexity of an antenna system is related to the efficiency of that approximation and its ability to correctly reproduce the desired radiation pattern depends on the approximation's accuracy. In the case of linear array antennae, the second step of the process amounts to the construction of a quadrature exact for a collection of functions on an interval. In Section 5.2, the algorithm of this paper is applied to the problem in order to generate highly accurate, yet efficient, quadratures discretizing a well-known continuous current distribution.

Generalized Gaussian quadratures also have obvious applications to the Nyström discretization of boundary integral equations. In particular, given an integral equation of the form

$$\lambda\sigma(x) + \int K(x,y)\sigma(y)\,dy = u(x), \tag{1.3}$$

Nyström discretization proceeds by choosing appropriate local representations

$$\sigma(y) \approx \sum_{j=1}^{n} \alpha_j \psi_j(y)$$

for the unknown σ and then approximating the integral in (1.3) via a collection of quadratures which integrate functions of the form

$$\sum_{j=1}^{n} \alpha_j K(x,y)\psi_j(y) \tag{1.4}$$

for various ranges of values of x. In Section 5.3, our scheme is used to construct quadrature rules integrating functions of the form

$$|x|^{\alpha}\,(p(x) + q(x))$$

over the interval $[-1,1]$, where $\frac{1}{2} \le \alpha \le 1$, $p(x)$ is a smooth function on the interval $[-1,0]$, and $q(x)$ is a smooth function on the interval $[0,1]$. These quadratures provide an efficient means of discretizing the boundary integral equation arising from the interior Dirichlet problem for Laplace's equation on certain domains with corners. The authors have constructed several considerably more elaborate schemes for the efficient discretization and rapid inversion of various Laplace and Helmholtz boundary integral equations on large scale two-dimensional domains with corner points which operate by constructing quadratures for functions of the form (1.4). These results will be reported at a later date.

Similarly, Section 5.4 describes the construction of a quadrature rule for the evaluation of the integral

$$\iint_{\Sigma} \nu(y) \cdot \frac{(x-y)}{|x-y|^3}\; \sigma(y)\; dS(y),$$

where Σ is a smooth surface in $\mathbb{R}^3$ parameterized over a triangle in the plane, $\nu(y)$ is the the outward unit normal to Σ at the point y, σ is a smooth function on Σ, and $x \in \Sigma$. Quadrature rules of this type are needed for the Nyström discretization of the integral equations arising from boundary value problems for Laplace and Helmholtz equations on surfaces in $\mathbb{R}^3$. Their construction is complicated by the fact that the double layer kernel in three dimensions on a smooth surface, which is defined by

$$K(x,y) = \nu(y) \cdot \frac{(x-y)}{|x-y|^3},$$

where $\nu(y)$ denotes the outward pointing unit normal vector to the surface at the point y, exhibits singularities which depend on the local geometry of the surface.

Like the previously published works [16, 5, 20], the algorithm of this paper is based on the observation that the nodes x_j and weights w_j of a quadrature exact for the collection of functions $\phi_1, \dots, \phi_m$ satisfy the nonlinear system of equations

$$\begin{aligned} F_1(x_1, \dots, x_n, w_1, \dots, w_n) &= b_1 \\ F_2(x_1, \dots, x_n, w_1, \dots, w_n) &= b_2 \\ &\vdots \\ F_m(x_1, \dots, x_n, w_1, \dots, w_n) &= b_m, \end{aligned} \tag{1.5}$$

where

$$F_i(x_1, \dots, x_n, w_1, \dots, w_n) = \sum_{j=1}^{n} \phi_i(x_j) w_j, \tag{1.6}$$

and

$$b_i = \int_a^b \phi_i(x)\, dx. \tag{1.7}$$

Moreover, the algorithm of this paper, like [16, 5, 20], operates by first constructing an $2m$-point quadrature rule and then applying Newton-type iterations to the system (1.5) in order produce an m-point rule. However, previous approaches utilize a continuation scheme with two key disadvantages: (1) the continuation scheme encounters numerical difficulties for collections of functions at different scales, and (2) it either succeeds in constructing a Gaussian quadrature or fails outright. The second is a particularly serious problem since many collections of functions do not admit Gaussian quadratures even though *nearly* Gaussian quadratures for those functions exist. The scheme of this paper, on the other hand, proceeds by downsampling the quadrature rule one point at a time using Gauss-Newton type iterations. It is not sensitive to differences in scale, and in the case of collections of functions for which no Gaussian quadrature exists, it is still usually capable of producing a nearly Gaussian formula. Indeed, this work was motivated in part by the failure of the continuation scheme for functions of the form (1.2) for exactly the two reasons described above.

This paper proceeds as follows: in Section 2 we discuss numerical and mathematical preliminaries, Section 3 describes three simple but fundamental procedures used by our scheme, in Section 4, we detail the algorithm of this paper, Section 5 describes several applications of our algorithm, and finally, we close with a discussion in Section 6 of possible future applications and generalizations of our algorithm.

2. Preliminaries

In this section, when X is an $n \times m$ matrix, we will denote by $\sigma_j(X)$ its jth largest singular value, by $\sigma_{\min}(X)$ its smallest singular value, and by $\sigma_{\max}(X)$ its largest singular value. Moreover, for $j > \min(n, m)$, we define $\sigma_j(X) = 0$.

2.1. Numerical rank.

If X is a $n \times m$ matrix, then we shall call the smallest nonnegative integer k such that $\sigma_{k+1}(X) < \epsilon$ the numerical rank of X to precision ϵ. If the numerical rank of an $n \times m$ matrix X is k to precision ϵ, then there exists an $n \times k$ matrix U whose columns are orthonormal and a $k \times m$ matrix V such that

$$\|X - UV\|_2 \le \epsilon. \tag{2.1}$$

A factorization of this type can be computed stably via a singular value decomposition (see, for instance, [7]). The k columns of the matrix U in (2.1) of course form an orthonormal basis for the span of the columns of X to precision ϵ.

We shall say that the numerical rank of a collection of vectors $v_1, \dots, v_n$ in $\mathbb{R}^m$ to precision ϵ is the numerical rank to precision ϵ of the matrix

$$\begin{pmatrix} v_1 & v_2 & \dots & v_n \end{pmatrix}$$

whose columns consist of the vectors v_j.

2.2. **Rank-revealing QR decompositions.** A partial QR decomposition for an $n \times m$ matrix A is a factorization of the form

$$A\Pi = Q \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix}, \tag{2.2}$$

where Q is an $n \times k$ orthogonal matrix, Π is an $m \times m$ permutation matrix, R_{11} is a $k \times k$ upper triangular matrix with nonnegative diagonal entries, R_{12} is a $k \times (m-k)$ matrix, and R_{22} is an $(n-k) \times (m-k)$ matrix. Such a factorization is said to be rank-revealing[1] if

$$\sigma_{\min}(R_{11}) \geq \frac{\sigma_k(A)}{p(k,n)} \quad \text{and} \quad \sigma_{\max}(R_{22}) \leq \sigma_{k+1}(A)p(k,n)$$

for a function $p(k,n)$ which is bounded by a low-degree polynomial in k and n.

In [10], it is shown that for any $n \times m$ matrix A and any integer $1 \leq k \leq \min(m,n)$, there exists a factorization of this form such that $\|R_{11}^{-1}R_{12}\|_\infty \leq d(k,n)$,

$$\sigma_i(A) \leq c(k,n)\ \sigma_i(R_{11})$$

holds for $i = 1, \ldots, k$, and

$$c(k,n)\ \sigma_{k+j}(A) \geq \sigma_j(R_{22})$$

holds for $j = 1, \ldots, m-k$, where $c(k,n) = \sqrt{1 + k(n-k)}$ and $d(k,n) = 1$. Moreover, [10] gives a stable algorithm for the computation of partial QR factorizations satisfying bounds of this type with $c = \sqrt{1 + nk(n-k)}$ and $d = \sqrt{n}$ which requires $O(n^2 m)$ floating point operations — that is, the same asymptotic complexity as the well-known pivoted Gram-Schmidt algorithm[2]. Factorizations with these properties are examples of strong rank-revealing QR decompositions (see [10] for a precise definition), which allow a number of important numerical computations to be performed in a stable fashion.

For instance, the following theorem is a consequence of the results of [10] described above.

THEOREM 2.1. *If the linear system*

$$Ax = b,$$

where A is an $n \times m$ matrix with $n \leq m$, admits a solution x, then for any positive integer $k \leq n$ there exists a vector $\hat{x}$ in $\mathbb{R}^m$ with no more than k nonzero entries such that

$$\|\hat{x}\|_\infty \leq 2\|x\|_\infty.$$

and

$$\|A\hat{x} - b\|_2 \leq \sigma_{k+1}(A)\sqrt{1 + k\,(n-k)}\ \|x\|_2.$$

Moreover, for any positive integer $k \leq n$, a vector $\tilde{x}$ in $\mathbb{R}^m$ with no more than k nonzero entries such that

$$\|\tilde{x}\|_\infty \leq \left(1 + \sqrt{n}\right)\|x\|_\infty \qquad \text{and} \qquad \|A\tilde{x} - b\|_2 \leq \sigma_{k+1}(A)\sqrt{1 + nk\,(n-k)}\ \|x\|_2.$$

can be constructed in a stable fashion via a rank-revealing QR decomposition using $O\left(n^2 m\right)$ floating point operations.

Proof. Suppose that

$$A\Pi = Q \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix} \tag{2.3}$$

is a partial QR decomposition for the matrix A such that $\|R_{11}^{-1}R_{12}\|_\infty \leq d(k,n)$,

$$\sigma_i(A) \leq c(k,n)\ \sigma_i(R_{11})$$

holds for $i = 1, \ldots, k$, and

$$c(k,n)\ \sigma_{k+j}(A) \geq \sigma_j(R_{22})$$

[1] Here we use the definition given in [10]. Slightly different formulations can be found, for instance, in [11].

[2] In fact, the performance of the scheme described in [10] depends on a parameter which affects both its running time and the resulting bounds. The values reported here represent but one possible configuration.

holds for $j = 1, \ldots, m-k$, and let x be a vector in $\mathbb{R}^m$ such that $Ax = b$. Denote by y_1 the first k entries of the vector $\Pi^{-1}x$ and by y_2 the last $(m-k)$ entries of the vector $\Pi^{-1}x$. If we let $\hat{x}$ be the vector in $\mathbb{R}^m$

$$\hat{x} = \Pi \begin{pmatrix} y_1 + R_{11}^{-1} R_{12} y_2 \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

then

$$\begin{aligned} \|A\hat{x} - b\|_2 &= \left\| Q \begin{pmatrix} R_{11} & R_{12} \\ 0 & 0 \end{pmatrix} \Pi^{-1}x - Q \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix} \Pi^{-1}x \right\|_2 \\ &= \left\| \begin{pmatrix} 0 & 0 \\ 0 & R_{22} \end{pmatrix} \Pi^{-1}x \right\|_2 \\ &\le c(k,n)\ \sigma_{k+1}(A)\ \|x\|_2 \end{aligned}$$

and

$$\begin{aligned} \|\hat{x}\|_\infty &\le \|y_1\|_\infty + \|R_{11}^{-1} R_{12} y_2\|_\infty \\ &\le \|y_1\|_\infty + d(k,n)\|y_2\|_\infty \\ &\le (1 + d(k,n))\ \|x\|_\infty. \end{aligned}$$

The result now follows immediately from Theorem 3.2 of [10]. □

Remark 2.1. *Our implementation of the algorithm described in this paper uses the pivoted Gram-Schmidt procedure with reorthogonalization in place of the algorithm of* [10] *to compute rank-revealing QR factorizations. As is observed in* [10] *and discussed thoroughly in the monograph* [4]*, the pivoted Gram-Schmidt algorithm with reorthogonalization, which is easier to implement and somewhat faster than the algorithm of* [10]*, works well in practice despite the existence of counterexamples which show that it can fail in certain circumstances.*

2.3. **Discretization of square integrable functions.** We shall say that a quadrature rule with nodes $x_1, \ldots, x_n \in [a, b]$ and positive weights $w_1, \ldots, w_n$ discretizes a collection of square integrable functions $f_1, \ldots, f_m$ defined on $[a, b]$ if

$$\int_a^b f_i(x) f_j(x)\, dx = \sum_{l=1}^n f_i(x_l) f_j(x_l) w_l$$

holds for all $i = 1, \ldots, m$ and $j = 1, \ldots, m$.

If $x_1, \ldots, x_n, w_1, \ldots, w_n$ is a quadrature discretizing a collection of functions $f_1, \ldots, f_m$ in $L^2([a, b])$, then the map T from the span S of the f_j to the Euclidean space $\mathbb{R}^n$ taking the function f to the vector

$$\begin{pmatrix} f(x_1)\sqrt{w_1} \\ f(x_2)\sqrt{w_2} \\ \vdots \\ f(x_n)\sqrt{w_n} \end{pmatrix} \tag{2.4}$$

is a Hilbert space isomorphism (a bijection which preserves inner products) of the subspace S onto the subspace of the Euclidean space $\mathbb{R}^n$ spanned by the vectors

$$\begin{pmatrix} f_1(x_1)\sqrt{w_1} \\ f_1(x_2)\sqrt{w_2} \\ \vdots \\ f_1(x_n)\sqrt{w_n} \end{pmatrix}, \ldots, \begin{pmatrix} f_m(x_1)\sqrt{w_1} \\ f_m(x_2)\sqrt{w_2} \\ \vdots \\ f_m(x_n)\sqrt{w_n} \end{pmatrix}.$$

This mapping allows linear algebraic computations to be performed on the functions $f_1, \ldots, f_m$.

For instance, an orthonormal basis $\phi_1, \dots, \phi_k$ for the span of the functions $f_1, \dots, f_m$ can be constructed by computing a rank-revealing QR decomposition

$$A = QR$$

for the matrix

$$A = \begin{pmatrix} f_1(x_1)\sqrt{w_1} & f_2(x_1)\sqrt{w_1} & \dots & f_m(x_1)\sqrt{w_1} \\ f_1(x_2)\sqrt{w_2} & f_2(x_2)\sqrt{w_2} & \dots & f_m(x_2)\sqrt{w_2} \\ \dots & \vdots & \ddots & \vdots \\ f_1(x_n)\sqrt{w_n} & f_2(x_n)\sqrt{w_n} & \dots & f_m(x_n)\sqrt{w_n} \end{pmatrix} \tag{2.5}$$

and defining the functions ϕ_j via the expression

$$\phi_j(x_i) = Q_{ij}/\sqrt{w_i},$$

where Q_{ij} denotes the entry in the ith row and jth column of the matrix Q. We shall, in fact, define the numerical rank of the functions $f_1, \dots, f_m$ to precision ϵ to be the numerical rank to precision ϵ of the matrix A (see [5, 20] for equivalent definitions which do not depend on an embedding).

In the case of an orthonormal collection of functions $\phi_1, \dots, \phi_m$, the nodes of a quadrature $x_1, \dots, x_n$ integrating products of the functions ϕ_j serve as stable interpolation nodes for the span of the ϕ_j. In particular, given the values of a function f of the form

$$f(x) = \sum_{j=1}^{m} \alpha_j \phi_j(x), \tag{2.6}$$

at the quadrature nodes x_i, the coefficients α_j can be computed by first applying the matrix

$$\begin{pmatrix} \phi_1(x_1)\sqrt{w_1} & \phi_1(x_2)\sqrt{w_2} & \dots & \phi_1(x_n)\sqrt{w_n} \\ \phi_2(x_1)\sqrt{w_1} & \phi_2(x_2)\sqrt{w_2} & \dots & \phi_2(x_n)\sqrt{w_n} \\ \dots & \vdots & \ddots & \vdots \\ \phi_m(x_1)\sqrt{w_1} & \phi_m(x_2)\sqrt{w_2} & \dots & \phi_m(x_n)\sqrt{w_n} \end{pmatrix} \tag{2.7}$$

to the vector $(f(x_1), \dots, f(x_n))$ and then scaling the resulting quantities by the square roots of the appropriate quadrature weights. Since the rows of the matrix (2.7) are orthonormal, this is an entirely stable mechanism for computing the coefficients in the expansion (2.6). Values of the function $f(x)$ can then be computed using the representation (2.6), assuming of course a mechanism for evaluating the ϕ_j at arbitrary points $x \in [a, b]$ is available.

Remark 2.2. *The omission of quadrature weights in (2.4) is a common but very serious error. Without them, the mapping T discretizing the function f_j is not an isometry and linear algebraic operations performed in Euclidean space do not correspond to the appropriate operations in the function space $L^2([a, b])$. Moreover, the resulting operations in Euclidean space are often ill-conditioned as a consequence of the omission, even in the case of extremely well-behaved smooth functions. An example of the latter hazard is given in Section 2.4.*

2.4. **Generalized Chebyshev quadratures.** A quadrature formula will be referred to as a Chebyshev quadrature for a set of $2n$ linearly independent $\phi_1, \dots, \phi_{2n} : [a, b] \to \mathbb{R}$ if it consists of $2n$ nodes and $2n$ weights and integrates exactly the functions ϕ_i, for all $i = 1, \dots, 2n$. The weights and nodes of a Chebyshev quadrature will be referred to as Chebyshev weights and nodes, respectively.

The construction of a Chebyshev quadrature for an orthonormal collection of square integrable functions defined on an interval $[a, b]$ is trivial given a preexisting quadrature integrating products of those functions. That a quadrature rule with nodes $x_1, \dots, x_n$ and weights $w_1, \dots, w_n$ integrates the functions $u_1, \dots, u_k$ can be expressed via the equation

$$\begin{pmatrix} u_1(x_1) & u_1(x_2) & \cdots & u_1(x_n) \\ u_2(x_1) & u_2(x_2) & \cdots & u_2(x_n) \\ \vdots & & \dots & \vdots \\ u_k(x_1) & u_k(x_2) & \cdots & u_k(x_n) \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_k \end{pmatrix}, \tag{2.8}$$

where r_i, $i = 1, \ldots, k$, is defined by

$$r_i = \int_a^b u_j(x)\,dx.$$

If we let

$$A = \begin{pmatrix} u_1(x_1)\sqrt{w_1} & u_1(x_2)\sqrt{w_2} & \cdots & u_1(x_n)\sqrt{w_n} \\ u_2(x_1)\sqrt{w_1} & u_2(x_2)\sqrt{w_2} & \cdots & u_2(x_n)\sqrt{w_n} \\ \vdots & & \cdots & \vdots \\ u_k(x_1)\sqrt{w_1} & u_k(x_2)\sqrt{w_2} & \cdots & u_k(x_n)\sqrt{w_n} \end{pmatrix}, \tag{2.9}$$

then Equation (2.8) can be rewritten as

$$A \begin{pmatrix} \sqrt{w_1} \\ \sqrt{w_2} \\ \vdots \\ \sqrt{w_n} \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_k \end{pmatrix}, \tag{2.10}$$

which shows that the vector $(r_1, \ldots, r_k)$ is in the span of the columns of the matrix A. By virtue of the orthonormality of the u_k and the use of a quadrature integrating products of the u_j, the rows of the matrix A are orthonormal. It follows that if

$$A = U\Sigma V^*$$

is a singular value decomposition for the matrix A, then

$$I = AA^* = (U\Sigma V^*)\,(V\Sigma U^*) = U\Sigma^2 U^*.$$

In other words, $U\Sigma^2U^*$ is a singular value decomposition for the $k \times k$ identity I. The uniqueness of singular values implies that Σ is the $k \times k$ identity matrix — that is, A has k nonzero singular values all of which are 1.

It now follows from Theorem 2.1 that a solution with no more than k nonzero entries for the equation

$$\begin{pmatrix} u_1(x_1)\sqrt{w_1} & u_1(x_2)\sqrt{w_2} & \cdots & u_1(x_n)\sqrt{w_n} \\ u_2(x_1)\sqrt{w_1} & u_2(x_2)\sqrt{w_2} & \cdots & u_2(x_n)\sqrt{w_n} \\ \vdots & & \cdots & \vdots \\ u_k(x_1)\sqrt{w_1} & u_k(x_2)\sqrt{w_2} & \cdots & u_k(x_n)\sqrt{w_n} \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ \cdots \\ z_n \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_k \end{pmatrix}$$

can be constructed via a rank-revealing QR decomposition. If $z_{i_1}, \ldots, z_{i_k}$ are the nonzero entries of such a solution, then the nodes $x_{i_1}, \ldots, x_{i_k}$ and weights $z_{i_1}\sqrt{w_{i_1}}, \ldots, z_{i_k}\sqrt{w_{i_k}}$ constitute a Chebyshev quadrature for the functions $u_1, \ldots, u_k$. Indeed, we can say slighty more:

THEOREM 2.2. *If there exists a quadrature $x_1, \ldots, x_n, w_1, \ldots, w_n$ with positive weights which integrates products of the orthonormal functions $u_1, \ldots, u_k$ in $L^2([a,b])$, then a Chebyshev quadrature $z_1, \ldots, z_k, v_1, \ldots, v_k$ for $u_1, \ldots, u_k$ such that*

$$|v_j| \le \left(1 + \sqrt{n}\right) w_j,$$

for all $j = 1, \ldots, k$, can be constructed in a stable fashion using $O(k^2 n)$ floating point operations.

Remark 2.3. *Equation (2.8) is an excellent example of the importance of proper discretization of functions. In the event the quadrature weights are omitted, the matrix appearing on the left hand side of (2.10) is generally ill-conditioned. For instance, if the functions are taken to be $u_k(x) = x^k$, it is a Vandermonde matrix. On the other hand, the matrix A, which has been scaled by the appropriate weights, is quite well conditioned even when $u_k(x) = x^k$.*

2.5. **Generalized Gaussian quadratures.** A quadrature formula will be referred to as Gaussian with respect to a set of $2n$ linearly independent functions $\phi_1, \dots, \phi_{2n} : [a,b] \to \mathbb{R}$ if it consists of n nodes and n weights and integrates exactly the functions ϕ_i, for all $i = 1, \dots, 2n$. The weights and nodes of a Gaussian quadrature will be referred to as Gaussian weights and nodes, respectively.

It has long been known that Gaussian quadratures admit generalization to fairly broad classes of functions and results in this direction can be found in [12, 13, 17, 18, 14]. In this paper, we will use the observation that a quadrature exact for a collection of functions $f_1, \dots, f_n$ satisfies the nonlinear system of equations

$$\begin{aligned} G_1(x_1, \dots, x_m, w_1, \dots, w_m) &= b_1 \\ G_2(x_1, \dots, x_m, w_1, \dots, w_m) &= b_2 \\ &\vdots \\ G_n(x_1, \dots, x_m, w_1, \dots, w_m) &= b_n, \end{aligned} \tag{2.11}$$

where

$$G_i(x_1, \dots, x_m, w_1, \dots, w_m) = \sum_{j=1}^{m} f_i(x_j) w_j$$

and

$$b_i = \int_a^b f_i(x)\, dx,$$

in order to construct Gaussian quadratures. Our numerical experiments indicate that Gaussian or near-Gaussian formulas exist for many collections of functions for which the theory of [12, 13, 17, 18, 14] does not apply.

2.6. **The damped Gauss-Newton method.** The damped Gauss-Newton method is a well-known iterative technique for the solution of nonlinear least-squares problems. It converges under very general conditions, and does not require that the Jacobian of the system be nonsingular. Here we give only elementary details; a more thorough treatment can be found in [6].

Suppose that $R : \mathbb{R}^n \to \mathbb{R}^m$ is a continuously differentiable function of the form

$$R(x) = \begin{pmatrix} r_1(x) \\ r_2(x) \\ \vdots \\ r_m(x) \end{pmatrix} \tag{2.12}$$

and let $J(x)$ be the Jacobian

$$J(x) = \begin{pmatrix} \frac{\partial r_1}{\partial x_1}(x) & \dots & \frac{\partial r_1}{\partial x_n}(x) \\ \vdots & & \vdots \\ \frac{\partial r_m}{\partial x_1}(x) & \dots & \frac{\partial r_m}{\partial x_n}(x) \end{pmatrix} \tag{2.13}$$

of R at the point x. The damped Gauss-Newton method is a numerical method for minimizing the function

$$f(x) = \frac{1}{2} \sum_{j=1}^{m} |r_j(x)|^2. \tag{2.14}$$

It belongs to a broad class of numerical optimization methods which proceed from an initial guess x_0 by forming a sequence $x_1, x_2, \dots$ of iterates via the formula

$$x_{k+1} = x_k + \alpha_k d_k \tag{2.15}$$

where d_k is referred to as the search direction at iteration k and α_k is a carefully chosen step size.

The primary purpose of this section is the statement Theorem 2.3 below, which gives conditions under which an iteration of the form (2.15) converges. We start with the following definition:

DEFINITION 2.1. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function and consider any iteration of the form (2.15). We say that the step length α_k and step direction d_k satisfy the Wolfe conditions at the point x_k if*

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \lambda \alpha_k \nabla f(x_k) \cdot d_k, \tag{2.16}$$

and

$$\nabla f(x_k + \alpha_k d_k) \cdot d_k \geq \beta \nabla f(x_k) \cdot d_k \tag{2.17}$$

for some constants $\lambda \in (0, 1)$ and $\beta \in (\lambda, 1)$.

The following theorem can be found in [6]:

THEOREM 2.3. *Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable function bounded from below, and assume that there exists $\gamma \geq 0$ such that*

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq \gamma \|x - y\|_2 \tag{2.18}$$

for every x and y in $\mathbb{R}^n$. Consider any iteration of the form (2.15) such that for each $k = 0, 1, \ldots$ the pair (d_k, α_k) satisfies the Wolfe conditions. If, in addition, one of the conditions

$$\nabla f(x_k) \cdot d_k < 0 \tag{2.19}$$

or

$$\nabla f(x_k) = 0 \quad \textit{and} \quad d_k = 0 \tag{2.20}$$

holds for each $k = 0, 1, \ldots$, then either

$$\nabla f(x_k) = 0 \textit{ for some } k \geq 0, \tag{2.21}$$

or

$$\lim_{k \to \infty} \frac{\nabla f(x_k) \cdot d_k}{\|d_k\|_2} = 0. \tag{2.22}$$

Remark 2.4. *Theorem 2.3 states that either the sequence x_k converges to a critical point for the function f or the direction d_k become orthogonal to the gradient of f. In practice, it is easy to avoid the latter condition, thus ensuring the convergence of $\nabla f(x_k)$ to 0.*

Remark 2.5. *Under mild conditions on the function f, given a sequence of $\{d_k\}$ satisfying condition (2.19), there exists a sequence of α_k satisfying the Wolfe conditions (see, for example,* [6] *Theorem 6.3.2).*

In the case of the damped Gauss-Newton method, the search direction d_k is chosen as a solution to the least-squares problem

$$\min_{d_k} \; ||J(x_k)d_k + R(x_k)||_2, \tag{2.23}$$

which is an affine approximation to the nonlinear least-squares problem

$$\min_{x} \; ||R(x)||_2.$$

Since

$$\nabla f(x_k) = J^*(x_k) R(x_k), \tag{2.24}$$

we have

$$\langle \nabla f(x_k), d_k \rangle = \langle R(x_k), J(x_k) d_k \rangle. \tag{2.25}$$

The search direction d_k is chosen so that $J(x_k)d_k$ is the projection of $-R(x_k)$ onto the column space of $J(x_k)$. It follows that

$$\langle \nabla f(x_k), d_k \rangle = \langle R(x_k), J(x_k) d_k \rangle \leq 0. \tag{2.26}$$

If we choose d_k to be 0 in the event that $\langle \nabla f(x_k), d_k \rangle = 0$, then we obtain the following theorem:

THEOREM 2.4. *Suppose that $R : \mathbb{R}^n \to \mathbb{R}^m$ is a continuously differentiable function of the form (2.12), $f : \mathbb{R}^n \to \mathbb{R}$ is given by (2.14), and the Jacobian, $J(x)$, of R is given by (2.13). Further suppose that $\|J(x)\|_2$ is bounded on $\mathbb{R}^n$ and there exists a constant $\gamma > 0$ such that*

$$\|J(x) - J(y)\|_2 \leq \gamma \|x - y\|_2$$

for all x and y in $\mathbb{R}^n$. If x_k is defined by the iteration

$$x_{k+1} = x_k + \alpha_k d_k$$

where, for each $k = 1, 2, \ldots$, d_k is a solution of the least-squares problem

$$||J(x_k)d_k + R(x_k)||_2 = \min_v \; ||J(x_k)v + R(x_k)||_2,$$

and the sequence $\{\alpha_k\}$ is chosen so that the pairs (d_k, α_k) satisfy the Wolfe conditions for $k = 0, 1, \ldots$, then either

$$\nabla f(x_k) = 0 \text{ for some } k \geq 0$$

or

$$\lim_{k \to \infty} \frac{\nabla f(x_k) \cdot d_k}{\|d_k\|_2} = 0.$$

2.7. **The Sherman-Morrison-Woodbury update formula.** The Sherman-Morrison-Woodbury formula gives an expression for the rank-k update

$$(A + UV^t)^{-1}$$

of the inverse of a matrix A. The following Lemma can be found, for instance, in [7]:

LEMMA 2.1. *Suppose that A is an invertible $n \times n$ matrix, U is an $n \times k$ matrix, and V is an $k \times n$ matrix. If the rank-k update*

$$(A + UV^t) \tag{2.27}$$

of the matrix A is invertible, then its inverse is

$$(A + UV^t)^{-1} = A^{-1} - A^{-1}U(I + V^t A^{-1} U)^{-1} V^t A^{-1}.$$

In this paper, given an $m \times n$ real matrix A, we will utilize the Sherman-Morrison-Woodbury formula to perform a specific type of update to the inverse of the $m \times m$ matrix AA^t. In particular, we wish to update the inverse of AA^t in order to form the inverse of the matrix BB^t, where B is obtained from A by deleting its j^{th} column. That is,

$$B = A - uv^t,$$

where u is the $m \times 1$ vector which is the j^{th} column of the matrix A and v is the $n \times 1$ vector with entries

$$v_i = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases} \tag{2.28}$$

A simple calculation shows that

$$\left(A - uv^t\right)\left(A - uv^t\right)^t = AA^t - uv^t A^t - Au^t v + uv^t vu^t \tag{2.29}$$
$$= AA^t - uu^t. \tag{2.30}$$

In other words, if we form the matrix B by deleting the j^{th} column of the matrix A, then BB^t can be formed from AA^t by a rank-1 update. The Sherman-Morrison-Woodbury formula then implies that the inverse of BB^t can be computed from $(AA^t)^{-1}$ via a rank-1 update.

3. Numerical apparatus

In this section, we describe three fundamental procedures which are utilized by the algorithm of this paper.

3.1. **Adaptive discretization of functions.** We now describe a simple mechanism for the construction of a quadrature formula discretizing (in the sense of Section 2.3) a collection of square integrable functions $\phi_1, \dots, \phi_m$ defined on an interval $[a, b]$. The input to this procedure is a collection $\phi_1, \dots, \phi_m$ of square integrable functions on $[a, b]$ (meaning that those functions can be evaluated at arbitrary points on $[a, b]$), a precision $\epsilon > 0$, and a reasonably large integer k which controls the number of interpolation nodes used on each subinterval (for the computations in this paper, we used $k = 30$). The algorithm proceeds in three stages:

Stage 1. In the first stage, the following procedure is used to discretize each of the ϕ_i separately. That is, the following sequence of steps is repeated for $i = 1, \dots, m$:

1. Construct the $2k$ Legendre nodes $x_1, \dots, x_{2k}$ on the interval $[a, b]$ as well as the $2k \times 2k$ matrix Q taking the values of polynomials of degree $2k - 1$ at the quadrature nodes x_j to the coefficients of their Legendre expansions.

2. Construct the coefficients $\alpha_0, \dots, \alpha_{2k-1}$ in the Legendre expansion

$$\phi_i(x) \approx \sum_{j=0}^{2k-1} \alpha_j L_j(x)$$

of ϕ_i, where L_j denotes the jth order Legendre polynomial on the interval $[a, b]$, by applying the $2k \times 2k$ matrix Q to the vector of values of the function ϕ_i at the quadrature nodes $x_1, \dots, x_{2k}$.

3. If the inequality

$$\sum_{j=k+1}^{2k} |\alpha_j|^2 < \epsilon^2 \tag{3.1}$$

is satisfied, then we conclude that the order k Legendre expansion for ϕ_i on the interval $[a, b]$ is sufficient. Otherwise, we split the interval $[a, b]$ into two subintervals, $[a, (a+b)/2]$ and $[(a+b)/2, b)]$, and repeat the procedure of this stage recursively for each of the subintervals.

Stage 2. Store the endpoints of each subinterval constructed in Stage 1 in an array. Sort the array and eliminate multiple elements. The resulting array of points on the interval $[a, b]$ is the array of endpoints of the subintervals of the final subdivision.

Stage 3. Construct a quadrature formulae for the interval $[a, b]$ by amalgamating the k-point Legendre quadratures on each subinterval.

The resulting piecewise Legendre quadrature on the interval $[a, b]$ integrates products of the functions $\phi_i(x)$ to high precision (assuming ϵ is chosen to be small). The functions ϕ_i can be represented by their piecewise Legendre expansions and this provides an efficient mechanism for evaluating them and their derivatives.

Remark 3.1. *The scheme of this section is a reasonably reliable mechanism for the discretization of sets of functions with singularities. The stopping condition (3.1) is analogous to, but much stronger than, that usually used to terminate an adaptive quadrature procedure. Just as any such quadrature procedure can fail for carefully designed counterexamples, so too can the procedure of this section fail under certain circumstances. The problem, however, is not encountered in the examples of this paper and whenever the authors have encountered it in practice, it has been easy to rectify.*

3.2. **Compression of sequences of functions.** The algorithm described in this section computes an orthonormal basis spanning approximately the same space as the input sequence. It "compresses" the input functions in the sense that a large rank-deficient collection of functions whose can be replaced for many purposes with the resulting orthonormal basis whose dimension is the numerical rank of the input functions. It uses as input a sequence of functions $\phi_1, \dots, \phi_m : [a, b] \to \mathbb{R}$, a quadrature with nodes $x_1, \dots, x_n$ and weights $w_1, \dots, w_n$ which integrates products of the ϕ_j, and a precision ϵ. The algorithm proceeds as follows:

1. Construct the $n \times m$ matrix A with entries

$$A_{ij} = \phi_j(x_i)\sqrt{w_i}.$$

2. Compute a precision ϵ rank-revealing QR decomposition for the matrix A in order to construct a factorization of the form

$$A \approx QR,$$

where $Q = (Q_{ij})$ is an $n \times k$ matrix with orthonormal columns and R is a n $k \times m$ upper triangle matrix such that

$$\|A - QR\|_2 \leq \epsilon.$$

3. Construct the k functions $u_1, \ldots, u_k$ defined by their values

$$u_j(x_i) = \frac{Q_{ij}}{\sqrt{w_i}}. \tag{3.2}$$

at the quadrature nodes.

The functions $u_1, \ldots, u_k$, which are completely determined by their values at the nodes x_i, constitute an orthonormal basis for the span of the input functions $\phi_1, \ldots, \phi_m$ to precision ϵ. In the event that the quadrature $x_1, \ldots, x_n, w_1, \ldots, w_n$ was constructed in the manner described in Section 3.1, the resulting basis functions and their derivatives can be cheaply evaluated once their piecewise Legendre expansions have been computed.

3.3. **Construction of Chebyshev quadratures.** In this section, we describe a numerical algorithm for the construction of a Chebyshev quadrature for an orthonormal sequence of functions. The algorithm uses as input an orthonormal collection $u_1, \ldots, u_k$ of functions and a pre-existing quadrature $x_1, \ldots, x_n, w_1, \ldots, w_n$, with $n > k$, which exactly integrates the functions $u_1, \ldots, u_k$. It produces as output a k-point quadrature formula consisting of k nodes $\tilde{x_1}, \ldots, \tilde{x_k} \in \{x_1, \ldots, x_n\}$ and k weights $\tilde{w_1}, \ldots, \tilde{w_k}$. The algorithm proceeds as follows:

1. Form the vector $r \in \mathbb{R}^k$ whose ith entry is the sum

$$r_i = \sum_{j=1}^{n} u_i(x_j) w_j,$$

   which is the value of the integral

$$\int_a^b u_i(x)\, dx.$$

2. Form the $k \times n$ matrix B with entries

$$B_{ij} = u_i(x_j)\sqrt{w_j}.$$

3. Construct a rank-revealing QR decomposition

$$B\Pi = Q \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix}$$

   such that Q is a $k \times k$ orthogonal matrix, R_{11} is a $k \times k$ upper triangular matrix, and Π is an $n \times n$ permutation matrix. Let i_j, $j = 1, \ldots, k$, denote the index of the nonzero entry in the jth column of the permutation matrix Π.
4. Use back substitution to construct a solution $z \in \mathbb{R}^k$ to the $k \times k$ system of equations

$$R_{11} z = Q^* r,$$

   which is, of course, a solution of the system

$$\begin{pmatrix} B_{i_1} & B_{i_2} & \cdots & B_{i_k} \end{pmatrix} z = r.$$

5. Form the new k-point quadrature $\tilde{x_1}, \ldots, \tilde{x_k}$, $\tilde{w_1}, \ldots, \tilde{w_k}$ by letting

$$\tilde{x_j} = x_{i_j} \quad \text{and} \quad \tilde{w_j} = z_j \sqrt{w_j}$$

   for all $j = 1, \ldots, k$.

4. Numerical algorithm

This section describes a numerical algorithm for the construction of the nodes and weights of a Gaussian quadrature rule for a sequence of functions. It proceeds in three stages. In the first stage, the input functions are discretized and compressed. In the second, a suboptimal Chebyshev quadrature rule is obtained for the compressed collection of functions. Finally, in the third phase, an optimization procedure is used to reduce the number of points needed by the quadrature.

The algorithm's input is a sequence of square integrable functions

$$\phi_1, \ldots, \phi_m : [a, b] \to \mathbb{R} \tag{4.1}$$

and three accuracies, ϵ_{disc}, ϵ_{comp}, and ϵ_{quad}. The first, ϵ_{disc}, controls the accuracy of the discretization of the input functions $\phi_1, \ldots, \phi_m$, the second, ϵ_{comp}, the accuracy of the scheme used to compress them, and the third, ϵ_{gauss}, is the accuracy for the optimization scheme. It is important to ensure that

$$\epsilon_{\text{disc}} < \epsilon_{\text{comp}}. \tag{4.2}$$

If the functions $\phi_1, \ldots, \phi_m$ are discretized to precision ϵ_{disc}, then singular functions u_j corresponding to singular values λ_j comparable to ϵ_{disc} contain little information about the functions $\phi_1, \ldots, \phi_m$. Indeed, they are little more than numerical noise and their inclusion can cause numerical difficulties with the algorithm. There is no particular harm in choosing $\epsilon_{\text{gauss}} < \epsilon_{\text{comp}}$, but will generally not result in a quadrature with accuracy greater than ϵ_{comp}.

The algorithm's output is a quadrature rule consisting of a set of nodes $x_1, \ldots, x_l$ and a set of weights $w_1, \ldots, w_l$ such that

$$\int_a^b \phi_i(x)\, dx \approx \sum_{j=1}^{l} \phi_i(x_j) w_j \tag{4.3}$$

for all $i = 1, \ldots, m$. It is important to note that the integer l depends on the numerical rank of the input set $\phi_1, \ldots, \phi_m$ to precision ϵ_{comp} rather than on the actual number of input functions. In typical cases, the input to this algorithm is a large collection of functions (e.g., $10,000$) the rank of which is modest (e.g., 40). A generalized Gaussian quadrature for such a collection would consist of 20 quadrature nodes and 20 quadrature weights. The accuracy of the resulting quadrature is typically limited by ϵ_{comp}.

Stage 1: Discretization and compression.

In this stage the following sequence of steps is performed to discretize and compress the input functions.

1. Use the technique of Section 3.1 to construct a quadrature with nodes $x_1, \ldots, x_n$ and weights $w_1, \ldots, w_n$ discretizing the input functions $\phi_1, \ldots, \phi_m$ to precision ϵ_{disc}.
2. Apply the procedure of Section 3.2 to compress the sequence $\phi_1, \ldots, \phi_n$ using a rank-revealing QR decomposition with precision ϵ_{comp}. Denote the resulting orthonormal functions by $u_1, \ldots, u_k$.
3. Compute the piecewise Legendre expansions for the orthonormal functions $u_1, \ldots, u_k$. Recall that the quadrature used to discretize $\phi_1, \ldots, \phi_m$ is a piecewise Legendre quadrature. It follows that the functions $u_1, \ldots, u_k$ — which are sums of the ϕ_j — can be represented using piecewise Legendre expansions and evaluated rapidly at arbitrary points in the interval $[a, b]$ using these expansions.

Stage 2: Construction of a k-point quadrature rule.

We now apply the procedure of Subsection 3.3 to construct a k-point quadrature formula for the orthonormal functions $u_1, \ldots, u_k$. Note that since the $u_1, \ldots, u_k$ are sums of the input functions, the quadrature $x_1, \ldots, x_n, w_1, \ldots, w_n$ is exact for the functions $u_1, \ldots, u_k$.

Denote the resulting k quadrature nodes by $\tilde{x_1}, \ldots, \tilde{x_k}$ and the k quadrature weights by $\tilde{w_1}, \ldots, \tilde{w_k}$.

Stage 3: Point-by-point reduction of the quadrature rule.

In this stage, beginning with the k-point quadrature rule $\hat{x_1}, \ldots, \hat{x_k}, \hat{w_1}, \ldots, \hat{w_k}$, we repeatedly apply the following sequence of steps, which attempt to reduce an n-point quadrature rule $x_1, \ldots, x_n, w_1, \ldots, w_n$ for $u_1, \ldots, u_k$ to an $(n-1)$-point quadrature rule for $u_1, \ldots, u_k$.

Step 1: Rank the remaining nodes.

1. For each $i = 1, \ldots, k$, compute the integrals
$$\int_a^b u_i(x)\,dx$$
using the original nested Legendre quadrature rule formed in Step 1. Form the vector
$$r = \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_k \end{pmatrix}.$$
2. Form the Jacobian matrix
$$J = \begin{pmatrix} u_1'(x_1)w_1 & u_1'(x_2)w_2 & \ldots & u_1'(x_n)w_n & u_1(x_1) & u_1(x_2) & \ldots & u_1(x_n) \\ u_2'(x_1)w_1 & u_2'(x_2)w_2 & \ldots & u_2'(x_n)w_n & u_2(x_1) & u_2(x_2) & \ldots & u_2(x_n) \\ \vdots & & \vdots & & & & \vdots & \\ u_k'(x_1)w_1 & u_k'(x_2)w_2 & \ldots & u_k'(x_n)w_n & u_k(x_1) & u_k(x_2) & \ldots & u_k(x_n) \end{pmatrix}$$
for the nonlinear system (2.11) in Section 2.5 and the inverse
$$A = (JJ^t)^{-1}.$$
of the product JJ^t.
3. For each node x_k, first use the Sherman-Morrison-Woodbury formula (see Subsection 2.7) to form the matrix
$$A_k = (J_k {J_k}^t)^{-1}$$
via two rank-1 updates to A, where the matrix J_k is obtained from J by deleting its kth and (k+1)st columns. That is, J_k is obtained from J by deleting the contributions from the node x_k and its corresponding weight w_k.

   Next, compute the damped Gauss-Newton step direction Δx_k for the nonlinear system obtained by omitting the point x_k; i.e., find a solution Δx_k to the least squares problem
$$\underset{x}{\operatorname{argmin}} \|J_k x - r\|_2,$$
where r is the vector of definite integrals computed above. The solution to the minimization problem is found by solving the normal equations
$$\Delta x_k = (J_k J_k^t)^{-1} J_k^t r = A_k J_k^t r.$$
4. For each $k = 1, \ldots, n$, let η_k denote the l^2-norm of the solution vector Δx_k. We will refer to the value η_k as the significance of the node x_k.
5. Renumber the nodes and weights of the quadrature so that $\{x_1, \ldots, x_n\}$ are arranged in order of increasing η_j.

Step 3: Attempt to remove a node.

For each $j = 1, \ldots, n$ we perform the following sequence of steps:

1. Form an initial guess $x_1, \ldots, \hat{x_j}, \ldots, x_n$ and $w_1, \ldots, \hat{w_j}, \ldots, w_n$ for the damped Gauss-Newton method by excluding the node x_j and its corresponding weight w_j.
2. Use the damped Gauss-Newton algorithm to form an $(n-1)$-point quadrature rule $\tilde{x_1}, \ldots, \tilde{x_{n-1}}$, $\tilde{w_1}, \ldots, \tilde{w_{n-1}}$ for the functions $u_1, \ldots, u_k$.

3. Measure the approximation error

$$\epsilon_j = \sum_{i=1}^{k} \left| \sum_{j=1}^{n-1} u_i(\tilde{x}_j)\tilde{w}_j - \int_a^b u_i(x)\,dx \right|^2$$

for this new quadrature rule.

4. If the error ϵ_j for the quadrature rule is sufficiently small (i.e., $\epsilon_j \le \epsilon_{\text{quad}}^2$), then we accept this $(n-1)$-point quadrature rule and go to Step 4.

Step 3: Form the $(n-1)$-point quadrature.

If an $(n-1)$ point quadrature rule with sufficient precision has been found, then we accept this rule and repeat the procedure of this stage for the newly formed $(n-1)$ point quadrature, beginning with Step 1. Otherwise, we accept the n-point quadrature rule which was the input to this stage and the algorithm terminates.

Remark 4.1. *The process terminates when an n-point quadrature rule cannot be reduced to an $(n-1)$-point quadrature rule without an unacceptable loss of precision.*

5. Numerical Experiments

We have implemented the algorithm of this paper in Fortran 77. The resulting code was compiled using the Lahey/Fujitsu Linux64 Fortran Compiler Release L8.10a. All reported timings were performed on a PC with a 2GHz Intel Core 2 Duo processor and 4GB of RAM.

Where possible, computations were performed in double precision (FORTRAN REAL*8) arithmetic. However, in order to full achieve double precision accuracy for the quadrature rules, it is sometimes necessary to perform the quadrature computations in extended (FORTRAN REAL*16) arithmetic. Note that this is not a result of inherent instability in the algorithm; rather, some (small) loss of precision in the discretization and optimization processes is inevitable. Generally, formulas accurate to near-double precision can be computed by performing computations in double precision arithmetic; e.g., the plane wave expansions of Section 5.1 can be constructed to precision approximately 1.0×10^{-13} without the use of extended precision. The machine used for our experiments does not support extended precision in hardware, which means that such calculations are lengthy. However, since the procedure of this paper is intended to be used in most instances in order to precompute quadratures for later use, long computations times are acceptable. All timings reported for computations performed in extended precision arithmetic are indicated with parentheses.

5.1. **Plane wave expansions.**

The Green's function for the Helmholtz equation in $\mathbb{R}^3$ satisfies the following identity, valid for $z > 0$:

$$\frac{e^{i\omega r}}{r} = \int_0^\infty e^{-z\sqrt{\xi^2-\omega^2}} J_0(\xi\sqrt{x^2+y^2}) \frac{\xi}{\sqrt{\xi^2-\omega^2}}\,d\xi, \tag{5.1}$$

where J_0 denotes the Bessel function of the first kind of order 0, $r = \sqrt{x^2+y^2+z^2}$, and the integral is understood as the limit as $\epsilon \to 0$ of the integral over the contour with parameterization

$$\xi(t) = t - \epsilon i e^{-(t-\omega)^2}, \quad 0 \le t \le \infty.$$

A derivation of a number of formulas of this type can be found in [19]. In [8], a scheme for accelerating fast multipole methods for the Helmholtz equations at low frequencies was introduced. It operates via discretizations of formula (5.1) which hold for x, y, and z satisfying

$$\begin{aligned} L \le z \le 4L \\ -4L \le x,y \le 4L. \end{aligned} \tag{5.2}$$

The appropriate quadrature rule ostensibly depends on the two parameters ω and L. In fact, by making the substitutions

$$\lambda = \frac{\xi}{L} \quad \text{and} \quad \omega = L\omega_0$$

in (5.1), we obtain the equivalent representation

$$\frac{e^{i\omega r}}{r} = L \int_0^\infty e^{-(zL)\sqrt{\lambda^2-\omega_0^2}} J_0(\lambda\sqrt{(xL)^2+(yL)^2}) \frac{\lambda}{\sqrt{\lambda^2-\omega_0^2}}\, d\lambda, \tag{5.3}$$

which shows that up to rescaling factors, the quadrature rule depends only on the product ωL — that is, the size of the box (5.2) in wavelengths. Moreover, since $J_0(z)$ satisfies the well-known

	Expansion order		
	$\epsilon = 1.0 \times 10^{-3}$	$\epsilon = 1.0 \times 10^{-7}$	$\epsilon = 1.0 \times 10^{-15}$
$.1 \le \omega L \le .11$	13	29	56
$1 \le \omega L \le 1.1$	17	34	60
$5 \le \omega L \le 6$	20	38	70
$10 \le \omega L \le 12$	27	42	97

Table 1: Orders of the discrete plane wave expansions (5.5) as a function of ωL and the accuracy of the expansion.

identity

$$J_0(z) = \frac{1}{\pi} \int_0^\pi \cos(z \sin(\theta))\, d\theta, \tag{5.4}$$

which can be found as formula (9.1.18) in [1], in order to generate a discretization of (5.1) that holds for

$$a \le \omega L \le b,$$

it suffices to take as input to the algorithm of Section 5 functions of the form

$$f_{\alpha,\beta,\omega}(\lambda) = e^{-\alpha\sqrt{\lambda^2-\omega^2}} \frac{\lambda \cos(\beta\lambda)}{\sqrt{\lambda^2-\omega^2}}, \tag{5.5}$$

where α, β, and ω are allowed to vary as

$$1 \le \alpha \le 4$$
$$0 \le \beta \le 4\sqrt{2}b,$$
$$a \le \omega \le b.$$

	Computation time		
	$\epsilon = 1.0 \times 10^{-3}$	$\epsilon = 1.0 \times 10^{-7}$	$\epsilon = 1.0 \times 10^{-15}$
$.1 \le \omega L \le .11$	0.44	3.20	(1997.12)
$1.0 \le \omega L \le 1.1$	2.63	5.84	(3001.18)
$5 \le \omega L \le 6$	1.53	11.48	(7921.28)
$10 \le \omega L \le 12$	6.10	27.05	(12001.11)

Table 2: Computation times (in seconds) for each of the plane wave quadratures. Times in parentheses refer to computations performed in extended precision arithmetic.

The algorithm of this paper was applied to functions of the form (5.5) in order to generate discretizations of formula (5.1) of the form

$$\frac{e^{i\omega r}}{r} \simeq \sum_j e^{-z\sqrt{\xi_j^2-\omega^2}} J_0(\xi_j\sqrt{x^2+y^2}) \frac{\xi_j}{\sqrt{\xi_j^2-\omega^2}} w_j \tag{5.6}$$

for boxes of varying sizes. The number of terms in the plane wave expansion is given in Table 1 for various ranges of ωL as a function of the accuracy of the representation in formula (5.6). Table 2 gives the computation times for each of the formulae.

Remark 5.1. *Plane wave expansions for complex-valued wavenumbers ω lying in a subset of the upper-half plane can be constructed by observing that the integrands appearing in (5.1) are analytic as a function of ω. It follows from the Maximum Modulus Principle that expansions which hold for ω lying in a complex domain can constructed by computing quadratures for ω on the boundary of the domain.*

5.2. **Linear array antennas.** The synthesis of an antenna system is a two-step process. First, a continuous current distribution which gives rise to a desired radiation pattern is determined. Then, a discrete approximation of the current distribution is calculated.

The radiation pattern generated by a current distribution $I(z)$ arranged on the subinterval $[-l/2, l/2]$ of the z-axis is given by

$$i\eta \frac{ke^{-ikr}}{4\pi r} \sin(\theta) \int_{-\frac{l}{2}}^{\frac{l}{2}} I(z) e^{ikz\cos(\theta)} dz,$$

where (r, θ, ϕ) is the spherical coordinate system defined by

$$\begin{aligned} x &= r\sin(\theta)\cos(\phi) \\ y &= r\sin(\theta)\sin(\phi) \\ z &= r\cos(\theta), \end{aligned}$$

η is the impedance of the medium, and k is the wavenumber of the resulting radiation (see, for instance, [3]). The integral identity

k	Approx. Frequency	Nodes	Computation time
2π	300MHz	8	0.11
10π	1500MHz	15	0.73
20π	3GHz	22	0.92
100π	15GHz	65	50.22

Table 3: Quadratures for the design of linear array antennas.

$$\int_{-\frac{l}{2}}^{\frac{l}{2}} J_0\left(i\pi B\, \sqrt{1-(2z/l)^2}\right) e^{ikz\cos(\theta)} dz = S(\theta), \tag{5.7}$$

where

$$S(\theta) = \begin{cases} l\frac{\sinh\left(\pi\sqrt{B^2-u^2}\right)}{\pi\sqrt{B^2-u^2}} & u^2 < B^2 \\ l\frac{\sin\left(\pi\sqrt{u^2-B^2}\right)}{\pi\sqrt{u^2-B^2}} & u^2 > B^2 \end{cases}$$

and

$$u = \frac{2\pi l}{k}\cos(\theta),$$

shows that the current distribution

$$I(z) = J_0\left(i\pi B\, \sqrt{1-(2z/l)^2}\right) e^{ikz\cos(\theta)}$$

gives rise to the radiation pattern

$$i\eta \frac{ke^{-ikr}}{4\pi r} \sin(\theta) S(\theta). \tag{5.8}$$

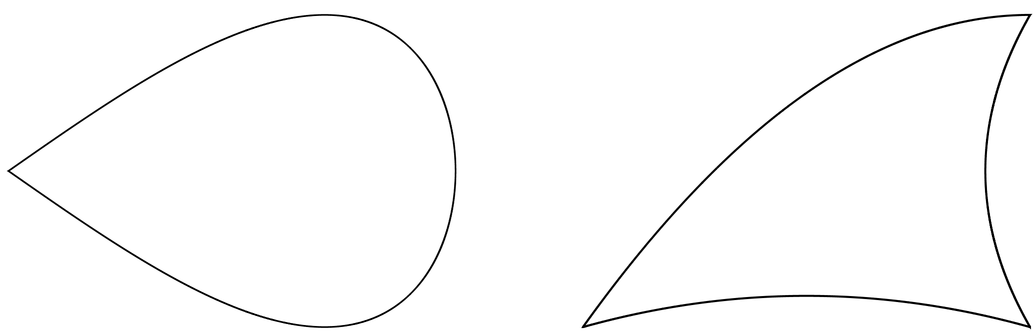
In order to design a physical antenna with the radiation pattern (5.8), it is necessary to discretize the continuous current distribution. This amounts to the construction of a quadrature for the integral appearing in (5.8) which holds for $0 \le \theta \le 2\pi$. The scheme of this paper was used to construct a number of such quadratures for various wavenumbers k which hold to precision 1.0×10^{-14}. The constant B (which controls the behavior of the radiation pattern's sidelobes) and the length l of the antenna array were both fixed at 1. Table 3 presents the computational results.

5.3. **Discretization of boundary integral equation on domains with corners.** As is well-known, solutions of the interior Dirichlet problem for Laplace's equation on a planar Lipschitz domain Ω can be uniquely represented in the form

$$u(x) = \frac{1}{2\pi} \int_{\partial\Omega} \nu(y) \cdot \frac{(x-y)}{|x-y|^2} \sigma(y)\ dy, \tag{5.9}$$

where $\nu(y)$ denotes the outward-pointing unit normal to the curve $\partial\Omega$ at the point y. The dipole distribution σ which gives rise to a particular potential u can be determined by solving the boundary integral equation

$$\frac{1}{2}\sigma(x) + \frac{1}{2\pi} \int_{\partial\Omega} \nu(y) \cdot \frac{(x-y)}{|x-y|^2}\ \sigma(y)\ dy = u(x). \tag{5.10}$$



(a) The "snowcone" domain Ω_1. (b) The "shark's fin" domain Ω_2.

Figure 1: The two domains under consideration in Section 5.3.

The Nyström discretization of the boundary integral equation (5.10) proceeds by fixing local approximate representations of the type

$$\sigma(y) \approx \sum_{j=1}^{N} \alpha_j \phi_j(y) \tag{5.11}$$

and discretizing the integral in Equation (5.10) via a collection of quadratures which integrate functions of the form

$$\sum_{j=1}^{N} K(x,y)\alpha_j \phi_j(y), \tag{5.12}$$

where $K(x,y)$ is the integral kernel appearing in (5.10), for various ranges of values of x. The monograph [2] contains a careful description of this approach to the discretization of boundary integral equations.

When $\partial\Omega$ is smooth, so are σ and the integral kernel

$$\nu(y) \cdot \frac{(x-y)}{|x-y|^2}.$$

As a result, in this case rapidly converging discretizations of (5.10) can be constructed easily using, for instance, high order piecewise Gaussian quadratures (see [15]). However, when $\partial\Omega$ has corner points, not only is the integral kernel no longer smooth, but solutions σ of Equation (5.10) can exhibit various types of singular behavior at corner points. In this circumstance, Gaussian quadratures — which are only efficient in the case of smooth functions or the products of smooth functions with a particular known singularity — are wholly inadequate.

Precision	Quadrature size	Computation time
1.0×10^{-3}	12	0.31
1.0×10^{-7}	18	0.77
1.0×10^{-15}	34	(225.01)

Table 4: Computational results for quadratures integrating functions of the form (5.13). Times are given in seconds and those in parentheses refer to computations performed in extended precision arithmetic.

x_j	w_j
$-0.9824461906113242E+00$	$0.4628691073069664E-01$
$-0.9009177578764879E+00$	$0.1173107532359643E+00$
$-0.7521057858284561E+00$	$0.1759654119695937E+00$
$-0.5602591877554817E+00$	$0.2013387420765489E+00$
$-0.3623339758008324E+00$	$0.1884029271037804E+00$
$-0.1938931776758727E+00$	$0.1444592854171595E+00$
$-0.7831414881302060E-01$	$0.8598806412895997E-01$
$-0.1967720750912080E-01$	$0.3414440079130691E-01$
$-0.1882916292803454E-02$	$0.6041433370195317E-02$
$+0.1882916293013683E-02$	$0.6041433370751212E-02$
$+0.1967720751032462E-01$	$0.3414440079264503E-01$
$+0.7831414881561512E-01$	$0.8598806413026824E-01$
$+0.1938931776795076E+00$	$0.1444592854178800E+00$
$+0.3623339758048276E+00$	$0.1884029271037858E+00$
$+0.5602591877591727E+00$	$0.2013387420759701E+00$
$+0.7521057858313670E+00$	$0.1759654119686442E+00$
$+0.9009177578783104E+00$	$0.1173107532347371E+00$
$+0.9824461906118565E+00$	$0.4628691072951182E-01$

Table 5: The 18 nodes and weights of a quadrature which integrates functions of the form (5.13) to precision 1.0×10^{-7}.

The authors have devised a number of schemes for the rapid solution of boundary integral equations on domains with corners. They operate by computing numerically efficient local representations of the form (5.11) for solutions of boundary integral equations near a corner point and using the machinery of this paper to construct extremely efficient quadratures for the discretization of boundary integral equations on domains with corners. These schemes are beyond the scope of this paper and will be reported at a later date. Instead, we present here a simple application of the algorithm of this paper to the discretization of boundary integral equations on certain domains with corners.

It is well-known (see, for instance, [9]) that solutions $\sigma(x)$ of the boundary integral equation (5.10) near a non-reentrant corner x_0 (one whose interior angle is less than π) behave as $O\left(|x-x_0|^\alpha\right)$ for some $1/2 \le \alpha \le 1$. The scheme of this paper was used to compute quadratures of various precisions for functions on the interval $[-1,1]$ of the form

$$|x|^\alpha \; (\; p(x) + q(x) \;)\,, \tag{5.13}$$

		Ω_1		Ω_2
ϵ_{quad}	N	E_{pot}	N	E_{pot}
1.0×10^{-3}	132	9.54×10^{-6}	276	6.95×10^{-8}
1.0×10^{-7}	138	3.24×10^{-10}	294	1.82×10^{-12}
1.0×10^{-15}	154	1.90×10^{-14}	342	1.53×10^{-14}

Table 6: Computational results obtained by applying quadratures for functions of the form (5.13) to to the solution of the interior Dirichlet problem on the domains Ω_1 and Ω_2.

where $1/2 \leq \alpha \leq 1$, $p(x)$ is a polynomial of degree 9 on $[-1, 0]$, and $q(x)$ is a polynomial of degree 9 on the interval $[0, 1]$. The computational results are summarized in Table 4.

The interior Dirichlet problem for Laplace's equation was solved on the two domains Ω_1 and Ω_2 shown in Figure 1 by inverting the integral equation (5.10). In each case, the boundary data was taken to be an electromagnetic potential generated by a collection of 10 random charges in the exterior of the domain. The corresponding boundary integral equations were discretized using the Nyström method. In particular, a region around each corner point was discretized using quadratures for functions of the form (5.13) while the smooth portions of the curve were discretized using 30th order piecewise Legendre quadrature formulae. The number of quadrature nodes used to discretize the smooth portion of the curve was held constant (independent of the precision of the corner quadrature formula used).

For each problem, the error in the representation (5.9) was measured at 100 randomly sampled points in the interior of the domain. The results are given in Table 6; ϵ_{quad} denotes the precision of the quadrature for functions of the form (5.13) used to discretize corner regions, N is the total number of discretization nodes, and E_{pot} denotes the largest absolute error measured in the representation (5.9).

5.4. **Discretization of boundary integral equations on surfaces.** In three dimensions, the double layer kernel on a smooth surface Σ is defined by

$$K(x, y) = \nu(y) \cdot \frac{(x - y)}{|x - y|^3},$$

where $\nu(y)$ denotes the outward unit normal to the surface Σ at the point y. It appears in Laplace and Helmholtz boundary integral equations on smooth surfaces in three dimensions, which include terms of the form

$$\iint_{\Sigma} \nu(y) \cdot \frac{(x - y)}{|x - y|^3} \sigma(y)\ dS(y), \tag{5.14}$$

where $\sigma(y)$ is a smooth function. Unlike the two-dimensional case, where the double layer kernel over a smooth curve is smooth, $K(x, y)$ is singular when $x = y$. Moreover, the construction of quadratures for the discretization of integrals of the form (5.14) for points $x \in \Sigma$ is complicated by the fact that the singularities in $K(x, y)$ depend on the local geometry of the surface Σ. The following observation, however, makes the task of constructing quadratures for the evaluation of the integrals (5.14) tractable.

LEMMA 5.1. *If $K(x, y)$ is the double layer potential on a smooth surface Σ, $x \in \Sigma$, and (r, θ) denotes the usual polar coordinate system centered at the point $x \in \mathbb{R}^3$, then the function*

$$L(r, \theta) = K(r \cos(\theta), r \sin(\theta))r$$

is smooth as a function of r and θ whenever $r > 0$.

It follows that (5.14) can be discretized using properly designed quadratures integrating functions of the form $r^j \theta^i$. However, standard practice calls for integrals of the type (5.14) to be discretized by triangulating the domain of a parameterization for Σ and using a collection of quadratures defined on triangles to approximate the integral. This means that to be useful in practice, any quadrature for the discretization of (5.14) must be given over a triangle.

We now describe the construction of quadratures for the evaluation of integrals of the form

$$\iint_T \sum_{i=0}^{n-1} \sum_{j=0}^{n} \alpha_{ij} r^{i-1} \theta^j \; dA, \tag{5.15}$$

where T is a triangle in the plane containing the origin and (r, θ) are the usual polar coordinates centered at the origin. In light of Lemma 5.1, it is apparent that a quadrature of this type can be used to discretize the integral (5.14) over a small portion of a smooth surface Σ parameterized over a triangle and containing the point x. It will be convenient to assume that n is even.

Since T contains the origin, it can be described in the usual polar coordinates system (r, θ) as

$$\begin{aligned} -\pi \le \theta \le \pi \\ 0 \le r \le f(\theta), \end{aligned}$$

where $f(\theta)$ is a continuous nonnegative function defined on $[-\pi, \pi]$, and the integral (5.14) can be written as a sum of terms which are constant multiples of

$$\int_{-\pi}^{\pi} \int_0^{f(\theta)} r^i \theta^j \; dr d\theta, \tag{5.16}$$

where $i = 0, \ldots, n-1$ and $j = 0, \ldots, n$.

In order to construct a quadrature evaluating integrals of this form, the algorithm of this paper is first used to generate a generalized Gaussian quadrature

$$\theta_1, \ldots, \theta_k, u_1, \ldots, u_k$$

for the functions

$$g_{i,j}(\theta) = \int_0^{f(\theta)} \theta^j r^i \; dr = \frac{\theta^j [f(\theta)]^{i+1}}{i+1}, \tag{5.17}$$

$i = 0, \ldots, n-1$, $j = 0, \ldots, n$, which are defined over the interval $[-\pi, \pi]$. Note that the function $f(\theta)$ is continuous but not smooth, so Gaussian quadrature would be highly suboptimal. For each of the resulting quadrature nodes θ_l, let

$$r_1^l, \ldots, r_{n/2}^l, v_1^l, \ldots, v_{n/2}^l$$

denote the $n/2$-point Legendre quadrature on the interval $[0, f(\theta_j)]$ (which integrates polynomials of degree $n-1$). The $kn/2$-point quadrature which consists of the nodes

$$\begin{aligned} x_{lm} &= r_m^l \cos(\theta_l) \\ y_{lm} &= r_m^l \sin(\theta_l) \end{aligned}$$

and weights

$$w_{lm} = r_m^l u_l v_m^l,$$

$1 \le l \le k$, $1 \le m \le n/2$, integrates functions of the form (5.15).

This procedure was applied in order to construct quadrature formulae for functions of the form (5.15) on the equilateral triangle T_0 in the plane with vertices

$$(-1, -1/\sqrt{3}), (1, -1/\sqrt{3}), (0, 2/\sqrt{3}).$$

Three quadratures rules for functions of the form (5.15) on T_0 were constructed to double precision accuracy using extended precision arithmetic, one for each of the values $n = 10$, 20, and 30. The resulting quadratures were tested by evaluating the integral of the double layer kernel over the surface

$$z(x, y) = x^2 + 2y^2 - x^3 y,$$

$(x, y) \in T_0$, using both the quadrature rules constructed here and a very high accuracy adaptive Gaussian quadrature procedure (which utilizes extended precision arithmetic). In Figure 2, we plot the function $f(\theta)$ for the triangle T_0. Computational results for the quadratures are given in Table 7.

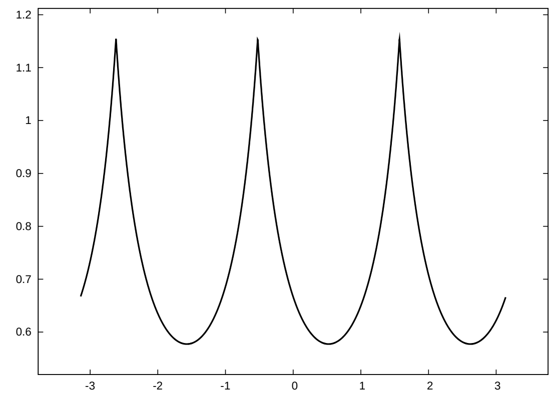


Figure 2: The function $f(\theta)$ on the equilateral triangle T_0.

Order (n)	Quadrature size	Computation time	Accuracy
10	210	(489.73)	4.07×10^{-6}
20	580	(1586.10)	5.13×10^{-11}
30	1035	(3385.42)	8.17×10^{-14}

Table 7: Computation results for the quadratures of Section 5.4. All computations were performed using extended precision arithmetic. Reported times are in seconds.

6. Generalizations and Conclusions

We have presented a simple and robust scheme for the computation of efficient quadrature rules for a wide class of square integrable functions which overcomes many of the difficulties of preceding works. In particular, it successfully computes Gaussian or nearly-Gaussian quadrature rules in a number of important cases for which the previous algorithms failed. Such quadrature rules have many applications in computation physics, some of which we have described here. We close with a number of observations and possible generalizations of this work:

1. The results of this paper are purely experimental. While there is a framework for proving (under certain conditions) the existence of generalized Gaussian quadratures [12, 13, 17, 18, 14], it does not apply to many of the examples of this paper. Indeed, the numerical experiments of this paper and those of [16, 5, 20] seem to indicate that such quadratures exist under very general conditions.
2. Stage 3 of the algorithm of Section 4, wherein quadrature nodes are eliminated one-by-one, is applicable to a wide range of problems. The method applies wherever a sparse solution for an underdetermined nonlinear system of equations is being sought.
3. There is no fundamental barrier to generalizing the procedure of this paper to higher dimensions. A procedure for generating efficient quadrature rules for collections of functions defined on domains in $\mathbb{R}^2$ would have many applications in numerical analysis. This topic is being vigorously investigated by the authors.

7. Acknowledgements

The second author was supported in part by the Air Force Office of Scientific Research under contract FA9550-06-1-0337 and in part by a research grant from Meyer Sound. The third author was supported in part by the Office of Naval Research under grant N00014-07-1-0711 and in part by DARPA-AFOSR grant FA9550-071-0541.

References

[1] M. Abramowitz and I. Stegun(editors), *Handbook of Mathematical Functions*, National Bureau of Standard, 1964.

[2] K. Atkinson, *The Numerical Solution of Integral Equations of the Second Kind*, Cambridge University Press, 1997.

[3] C. A. Balanis, *Antenna Theory. Third Edition*, John Wiley and Sons, Hoboken, New Jersey, 2005.

[4] A. Björck, *Numerical Methods for Least Squares Problems*, SIAM, Philadephia, 1996.

[5] H. Cheng, V. Rokhlin, and N. Yarvin, *Nonlinear optimization, quadrature, and interpolation*, SIAM J. Optim, 9 (1999), pp. 901–923.

[6] J. Dennis and R. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Englewood Cliffs, New Jersey, 1983.

[7] G. Golub and C. V. Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, 1983.

[8] L. Greengard, J. Huang, V. Rokhlin, and S. Wandzura, *Accelerating fast multipole methods for the helmholtz equation at low frequencies*, IEEE Comput. Sci. Eng, 5 (1998), pp. 32–38.

[9] P. Grisvard, *Singularities in Boundary Value Problems*, Springer-Verlag, 1992.

[10] M. Gu and S. Eisenstat, *Efficient algorithms for computing a strong rank-revealing QR factorization*, SIAM J. Sci. Comput., 17 (1996), pp. 848–869.

[11] H. Hong and C. Pan, *Rank-revealing QR factorization and SVD*, Math. Comp., 58 (1992), pp. 213–232.

[12] S. Karlin, *The existence of eigenvalues for integral operators*, Trans. Amer. Math Soc., 113 (1964), pp. 1–17.

[13] S. Karlin and W. Studden, *Tchebycheff systems with applications in Analysis and Statistics*, Wiley-Interscience, New York, 1966.

[14] M. Krein, *The Ideas of P.L. Chebyshev and A.A. Markov in the Theory of Limiting Values of Integrals, Amer. Math. Soc. Transl. Ser. 2, 12*, AMS, Providence, R.I., 1959.

[15] R. Kress, *Integral Equations*, Springer-Verlag, New York, 1999.

[16] J. Ma, V. Rokhlin, and S. Wandzura, *Generalized Gaussian quadrature rules for systems of arbitrary functions*, SIAM J. Numer. Anal., 33 (1996), pp. 971–996.

[17] A. Markov, *On the limiting value of integrals in connection with interpolation*, Zap. Imp. Akad. Nauk. Fix-Mat. Otd., 6 (1898). (in Russian).

[18] ———, *Selected Papers on Continued Fractions and the Theory of Functions Deviating Least from Zero*, OGIZ, Moscow, Leningrad, 1948. (in Russian).

[19] P. Morse and H. Feshbach, *Methods of Mathematical Physics*, Feshbach Publishing, Minneapolis, 1981.

[20] N. Yarvin and V. Rokhlin, *Generalized Gaussian quadratures and singular value decompositions of integral operators*, SIAM J. Sci. Comput., 20 (1998), pp. 699–718.