ON GENERALIZED GAUSSIAN QUADRATURES FOR BANDLIMITED EXPONENTIALS

MATTHEW REYNOLDS, GREGORY BEYLKIN AND LUCAS MONZÓN

ABSTRACT. We review the methods in [4] and [24] for constructing quadratures for bandlimited exponentials and introduce a new algorithm for the same purpose. As in [4], our approach also yields generalized Gaussian quadratures for exponentials integrated against a non-sign-definite weight function. In addition, we compute quadrature weights via ℓ^2 and ℓ^∞ minimization and compare the corresponding quadrature errors.

1. INTRODUCTION

We revisit the construction of quadratures for bandlimited exponentials $\left\{e^{ibx}\right\}_{|b|\leq c}$ integrated against a real-valued weight function w on the interval $|x| \leq 1$. These functions are not necessarily periodic in $[-1,1]$. Unlike the classical Gaussian quadratures for polynomials which integrate exactly a subspace of polynomials up to a fixed degree, Gaussian type quadratures for exponentials use a finite set of nodes in order to integrate the infinite set of functions $\left\{e^{ibx}\right\}_{|b|\leq c}$. While it is not possible to construct exact quadratures in this case, those introduced in [4] integrate with (user-selected) accuracy ϵ all exponentials with $|b| \leq c$. We note that, for a given bandlimit c and accuracy ϵ, quadratures of this type are not unique.

For a given accuracy ϵ, bandlimit c, and weight function w, the Gaussian-type quadratures in [4] are designed to integrate functions in the linear space

$$\mathcal{E}_c = \left\{ f \in L_\infty[-1,1] \mid f(x) = \sum_{k\in Z} a_k e^{ib_k x} \text{ with } \{a_k\} \in l^1 \text{ and } |b_k| \leq c \right\},$$

so that

$$\left| \int_{-1}^{1} f(x) w(x)\, dx - \sum_{m=1}^{M} f(x_m) w_m \right| < \epsilon, \quad f \in \mathcal{E}_c.$$

Note that functions in $\mathcal{E}_c$ may be approximated by a linear combination of exponentials $\left\{e^{icx_m x}\right\}_{m=1}^{M}$ with accuracy ϵ, if the quadrature nodes $\{x_m\}_{m=1}^{M}$ and corresponding weights are constructed for accuracy ϵ^2 and bandlimit $2c$ [4].

Key words and phrases. quadratures for bandlimited exponentials, prolate spheroidal wave functions, ℓ^∞-minimization, Gaussian quadratures, generalized Gaussian quadratures, matrix pencil method, HSVD method .

This research was partially supported by NSF grants DMS-1009951, DGE-0801680, DMS-0602284, and DOE/ORNL grant 4000038129.

Appl. Comput. Harmon. Anal. 34 (2013) 352-365, http://dx.doi.org/10.1016/j.acha.2012.07.002
.

An alternative approach in [24] yields quadratures to integrate bandlimited functions in

$$\mathcal{B}_c = \left\{ f \in L^2(\mathbb{R}) \mid \hat{f}(\omega) = 0 \text{ for } |\omega| \geq c \right\},$$

with the weight function $w(x) = 1$. The approach is based on explicitly constructing and using the Prolate Spheroidal Wave Functions (PSWFs), a basis of $\mathcal{B}_c$. The PSWFs form a Chebyshev system, leading to a classical recipe to find quadrature nodes as the zeros of an appropriately selected PSWF. To improve the accuracy of the quadrature, the positions of the nodes and the values of the weights are optimized via a Newton-type procedure.

Since the space $\mathcal{E}_c$ is dense in $\mathcal{B}_c$ and vice versa, the quadratures in [4] for $w = 1$ and [24] may be used interchangeably (we discuss this further in this paper). We note that the method in [4] allows us to construct quadratures for a weight function that does not have to be positive (see e.g. [3, Section 5]).

We present a new approach for designing quadratures in $\mathcal{E}_c$ using a setup similar to [4] but computing nodes as eigenvalues of matrices rather than zeros of an eigen-polynomial. This establishes a connection between the computation of quadratures and the so-called HSVD and Matrix Pencil methods in signal processing.

We also introduce an alternative approach for computing weights that yields an essentially uniform error within the bandwidth of validity of these quadratures. These quadrature weights are obtained by minimizing the ℓ^∞ error over the bandlimit of interest. Formulating the problem of finding weights as that of convex nonlinear optimization, we solve it using the software package CVX [13] and check the results using our own implementation of the primal-dual potential reduction algorithm [20]. Additionally, for the weight $w(x) = 1$, we compare the accuracies and behavior of the error of our new quadratures and those obtained using [4] and [24]. We also discuss the computational cost for obtaining the quadratures on all three approaches.

One of the reasons for our study is to facilitate applications of these quadratures. Since their introduction, quadratures for bandlimited exponentials found applications in solving partial differential equations (see e.g. [5, 8, 21]). In particular, they allow us to discretize operators using their spectral representation while avoiding the spurious eigenvalues appearing in other spectral discretizations. It is a significant improvement since, otherwise, these spurious eigenvalues increase the norm of the matrices (representing differential operators) by an order of magnitude (see e.g. [21]). Another application of quadratures for integration of bandlimited exponentials (with a weight) yields a fast Discrete Fourier transform in polar and spherical coordinates in the Fourier space [2] (see also [1] for integration on the sphere).

Another important property of these quadratures is that, for a fixed number of nodes, we can trade accuracy for bandwidth. This trade-off is not available for standard polynomial quadratures and is a significant advantage in applications. This is especially useful in signal processing, where the measured data may be of low precision. We note that, in practice, the accuracy of any quadrature is limited either by the accuracy of the projection onto functions for which the quadrature is exact or by the floating point arithmetic (e.g., double precision). Thus, approximate quadratures may be viewed as setting the accuracy of integration upfront.

In Section 2 we briefly describe the two methods for finding Gaussian-type quadratures for bandlimited functions. In Section 3 we consider a method for finding quadrature nodes for bandlimited functions as the eigenvalues of a matrix.

In Section 4 we develop approaches to finding quadrature weights by minimizing either ℓ^2 or ℓ^∞ error over the bandlimit of interest. We present examples of computing these new quadratures in Section 5. Finally, in Section 6 we compare the new quadratures with those obtained in [4] and [24].

2. Preliminaries

2.1. Quadratures for bandlimited functions via trigonometric moments.

Let us briefly summarize a method in [4] for generating quadratures to integrate the family of exponentials $\left\{e^{ibx}\right\}_{|b|\le c}$ with a real-valued weight function w. First, we compute the trigonometric moments

$$u_n = \int_{-1}^{1} e^{icxn/N} w\left(x\right) dx, \quad -N \le n \le N, \tag{2.1}$$

where $c>0$ is the bandlimit. The number of moments, $2N+1$, is chosen sufficiently large so that the function

$$u(y) = \int_{-1}^{1} e^{icxy} w\left(x\right) \mathrm{d}x, \quad y \in \left[-1,1\right],$$

is oversampled. We then arrange the trigonometric moments $\left\{u_n\right\}_{n=-N}^{N}$ as the entries of a self-adjoint Toeplitz matrix $\mathbf{T} = \left\{u_{n-n'}\right\}_{0\le n,n'\le N}$. If the weight function w is non-negative, then this matrix coincides with the Gram matrix $\mathbf{G}$,

$$\mathbf{G}_{n-n'} = \int_{-1}^{1} e^{ic\frac{n}{N}x} e^{-ic\frac{n'}{N}x}\, w(x)dx,$$

for a collection of linearly independent functions $\left\{e^{ic\frac{n}{N}x}\right\}_{n=0,\dots,N}$. We exploit this connection later in the paper. However, we also note that if no assumption on the sign of w is made, we still can use the matrix $\mathbf{T}$ of trigonometric moments for computing quadratures (see [3, Section 5]).

Computing the eigenvector $\mathbf{q}^{(s)} = \left[q_0, \dots q_N\right]^t$ of the matrix $\mathbf{T}$ corresponding to a small eigenvalue $\lambda^{(s)} > 0$, we form the eigenpolynomial $q^{(s)}(z) = \sum_{n=0}^{N} q_n z^n$. Assuming that this polynomial has only simple roots $\left\{\gamma_j\right\}_{j=1}^{N}$, $\gamma_j \ne 0$, it is shown in [4, Theorem 4.1] that there exist weights $\left\{w_j\right\}_{j=1}^{N}$ such that for all Laurent polynomials $P(z)$ of degree at most N,

$$\int_{-1}^{1} P(e^{i\pi t})w(t)dt = \sum_{j=1}^{N} w_j P(\gamma_j) + \frac{1}{2}\lambda^{(s)} \int_{-1}^{1} P(e^{i\pi t})dt.$$

This implies

$$\left| \int_{-1}^{1} P(e^{i\pi t})w(t)dt - \sum_{j=1}^{N} w_j P(\gamma_j) \right| \le \frac{1}{2}\lambda^{(s)} \left| \int_{-1}^{1} P(e^{i\pi t})dt \right| = \frac{1}{2}\lambda^{(s)} \left|p_0\right|,$$

where p_0 is the constant coefficient of P. In this approximate quadrature the error is controlled by the eigenvalue $\lambda^{(s)}$ and the quadrature nodes, γ_j, $j=1,\dots,N$ depend on the bandlimit c and the selected accuracy ϵ.

A numerical algorithm for computing quadratures via this method is formally $\mathcal{O}\left(N\left(log\, N\right)^{2}\right)$. However, in its current implementation, the step that solves equation $\mathbf{Tx}_0 = \mathbf{e}_0$, where $\mathbf{e}_0 = [1, 0, \dots 0]^t$, uses the Wiener-Levinson algorithm of complexity $\mathcal{O}\left(N^2\right)$ with a small constant which is sufficiently fast for $N \approx 10^4$.

We also note that the number of nodes with a significant weight is controlled by the index of the eigenvalue. Among the N roots of the eigenpolynomial $q^{(s)}(z)$, typically only s of them correspond to nodes with significant weights. Indeed, in most cases, solving the Vandermonde system for the weights w_j, $j = 1, \dots, N$ gives only s weights with absolute value greater than the eigenvalue $\lambda^{(s)}$. In practice, it is not difficult to identify the nodes corresponding to the significant weights since they are located inside the interval of integration. Computing high accuracy quadratures ($\epsilon < 10^{-12}$, for example) involves small eigenvalues, so we must use extended precision arithmetic. Importantly, when these quadratures are used, no extra precision is required.

If the weight function $w = 1$, then the eigenpolynomial $q^{(s)}(z)$ is a Discrete Prolate Spheroidal Wave Function (DPSWF) (see [22, Sections 2.1-2.3]) and the nodes are zeros of the DPSWF corresponding to the eigenvalue $\lambda^{(s)}$. The quadratures obtained for $w = 1$ may be compared with those in [24] obtained by a different approach that uses the PSWFs.

2.2. **Quadratures for bandlimited functions via PSWFs.**

In [24] quadratures are constructed using the PSWFs, which form a basis for bandlimited functions. The approach closely follows the classical method of obtaining Gaussian quadratures for polynomials. The PSWFs satisfy

$$\int_{-1}^{1} e^{icxy}\psi_j\left(x\right)\mathrm{d}x = \lambda_j\psi_j\left(y\right), \quad j = 0, 1, \dots$$

where $c > 0$ is the bandlimit. They are the eigenfunctions of the operator

$$F_c\phi\left(y\right) = \int_{-1}^{1}\phi\left(x\right)e^{-icxy}dx,$$

as well as the eigenfunctions of the operator $Q_c = \frac{c}{2\pi}F_c^*F_c$,

$$\frac{1}{\pi}\int_{-1}^{1}\frac{\sin(c(y-x))}{y-x}\psi_j(x)\ dx = \mu_j\psi_j(y),$$

where

$$\mu_j = \frac{c}{2\pi}|\lambda_j|^2, \quad j = 0, 1, 2, \dots.$$

Slepian and Pollak [23] observed that ψ_j are also the eigenfunctions of the differential operator

$$\left(-(1-x^2)\frac{d^2}{dx^2} + 2x\frac{d}{dx} + c^2x^2\right)\psi_j(x) = \eta_j\psi_j(x), \tag{2.2}$$

i.e., they coincide with the classical Prolate Spheroidal Wave functions of mathematical physics. In (2.2), the eigenvalues η_j form a strictly increasing, positive sequence.

Since the PSWFs form a Chebyshev system, the approach for computing quadratures in [24] first finds ψ_j by solving (2.2) and then computes the M nodes as zeros of ψ_M, $\psi_M(x_j) = 0$, $j = 1, \dots M$. It is observed in [24] that the accuracy of quadratures may be improved by optimizing the positions of nodes and the values of weights further. A Newton-type optimization (using ℓ^2 norm) is shown to gain an extra $1-2$ digits in the accuracy of the quadratures.

A drawback of this approach is that it is not clear how to apply it for a general weight function since no differential operator is available (see [14]). On the other hand, given that a differential operator is available for the weight function $w=1$, positions of nodes may be found rapidly in $\mathcal{O}(M)$ operations using the algorithm in [10]. This fact that the PSWFs satisfy the second order differential equation in (2.2) implies that their zeros may be found without ever explicitly computing the functions themselves. We note that the DPSWFs (see previous section) also satisfy a second order differential equation and, hence, the algorithm in [10] is applicable in that case as well.

3. Computing quadrature nodes as eigenvalues

3.1. **Classical quadratures for polynomials.**

Let us illustrate finding nodes as eigenvalues of a matrix by constructing the classical Gaussian quadrature with M nodes $\{x_m\}_{m=1}^{M}$. Let us consider a basis $\{\phi_l(x)\}_{l=0}^{M-1}$ in the subspace of real-valued polynomials of degree up to $M-1$ equipped with the inner product

$$\langle p,q\rangle = \int_{-1}^{1} p(x)q(x)\, w(x)dx.$$

We form the square matrix $\mathbf{A}\in\mathbb{R}^{M\times M}$ of entries

$$\mathbf{A}_{ll'} = \int_{-1}^{1} \phi_l(x)\phi_{l'}(x)\, w(x)dx = \sum_{m=1}^{M} \phi_l(x_m) w_m \phi_{l'}(x_m),$$

where x_m are the desired quadrature nodes and w_m the corresponding quadrature weights. Since the product of two polynomials in this subspace has degree of at most $2M-2$, the exact quadrature should also compute the integral

$$\mathbf{B}_{ll'} = \int_{-1}^{1} \phi_l(x)\, x\, \phi_{l'}(x)\, w(x)dx = \sum_{m=1}^{M} \phi_l(x_m) w_m x_m \phi_{l'}(x_m).$$

Denoting the non-singular matrix $\boldsymbol{\Phi} = \{\phi_l(x_m)\}_{\substack{l=0,\dots,M-1\\ m=1,\dots,M}}$, we obtain $\mathbf{A}=\boldsymbol{\Phi}\mathbf{W}\boldsymbol{\Phi}^t$ and $\mathbf{B}=\boldsymbol{\Phi}\mathbf{X}\mathbf{W}\boldsymbol{\Phi}^t$, where $\mathbf{W}=\operatorname{diag}(w_1\dots w_M)$ and $\mathbf{X}=\operatorname{diag}(x_1\dots x_M)$ are diagonal matrices and $\mathbf{M}^t$ denotes the transpose of the matrix $\mathbf{M}$. Computing

$$\mathbf{C}=\mathbf{B}\mathbf{A}^{-1}=\boldsymbol{\Phi}\mathbf{X}\mathbf{W}\boldsymbol{\Phi}^t\left(\boldsymbol{\Phi}^t\right)^{-1}\mathbf{W}^{-1}\boldsymbol{\Phi}^{-1}=\boldsymbol{\Phi}\mathbf{X}\boldsymbol{\Phi}^{-1},$$

implies that the nodes of the quadrature are the eigenvalues of the matrix $\mathbf{C}$. We obtain the same quadrature nodes by considering $\mathbf{A}^{-1}\mathbf{B}$.

We note that if $\{\phi_l(x)\}_{l=0}^{M-1}$ are orthogonal polynomials, then the matrix $\mathbf{A}$ is diagonal and the matrix $\mathbf{B}$ is tridiagonal. Thus, as we show in Appendix 8.1, we recover the Golub-Welsch algorithm [11].

3.2. **Quadratures for inner products of bandlimited exponentials.**

Let us now apply the approach illustrated in Section 3.1 to finding quadratures for exponentials with bandlimit c. Since the collection of exponentials $\left\{e^{ibx}\right\}_{|b|\le c}$ is infinite, exact quadratures are not available and, instead, we construct approximate quadratures for an arbitrary user-selected accuracy ϵ. These quadratures integrate exponentials of bandlimit c against a real-valued weight function $w(x)$, so that

$$\left|\int_{-1}^{1} e^{ibx} w\left(x\right) dx - \sum_{m=1}^{M} e^{ibx_m} w_m\right| < \epsilon, \;\; |b| \leq c, \tag{3.1}$$

where $x_m \in [-1,1]$ and $w_m \in \mathbb{R}\setminus\{0\}$.

To solve this problem, we consider

$$G(b,b') = \int_{-1}^{1} e^{i\frac{b}{2}x} e^{-i\frac{b'}{2}x} w(x)dx, \;\; |b|\,,|b'| \leq c, \tag{3.2}$$

which we discretize as

$$\int_{-1}^{1} e^{i\frac{c}{2}\frac{n}{N}x} e^{-i\frac{c}{2}\frac{n'}{N}x}\, w(x)dx, \;\; n,n' = -N,\ldots,N, \tag{3.3}$$

where $N > M$ by an (oversampling) factor. However, it is more convenient to consider instead the Hermitian $(N+1)\times(N+1)$ matrix

$$\mathbf{G}_{nn'} = \int_{-1}^{1} e^{ic\frac{n}{N}x} e^{-ic\frac{n'}{N}x}\, w(x)dx, \;\; n,n' = 0,\ldots,N, \tag{3.4}$$

which oversamples the interval $[-c,c]$ in the same fashion with an appropriate N. Note that if $w \geq 0$, $\mathbf{G}$ is a Gram matrix of inner products. As discussed in Section 2.1, the resulting quadratures also depend weakly on the choice of N.

Let us seek $\{x_m\}_{m=1}^{M}$ and $\{w_m\}_{m=1}^{M}$, with $M < N$, so that

$$|\mathbf{G}_{nn'} - \mathbf{Q}_{nn'}| < \epsilon, \;\; n,n' = 0,\ldots,N, \tag{3.5}$$

where the quadrature matrix $\mathbf{Q}$ has entries

$$\mathbf{Q}_{nn'} = \sum_{m=1}^{M} e^{icx_m\frac{n}{N}} w_m e^{-icx_m\frac{n'}{N}}, \;\; n,n' = 0,\ldots,N. \tag{3.6}$$

First, we show that it is possible to obtain the quadrature nodes by finding eigenvalues of an appropriate matrix. We consider two submatrices of $\mathbf{Q}$, $\mathbf{A}$ and $\mathbf{B}$,

$$\mathbf{A} = \left\{\mathbf{Q}_{nn'}\right\}_{\substack{n=0,\ldots,N-1\\ n'=0,\ldots,N}}, \;\; \mathbf{B} = \left\{\mathbf{Q}_{nn'}\right\}_{\substack{n=1,\ldots,N\\ n'=0,\ldots,N}}.$$

These submatrices may be written as

$$\mathbf{A} = \widetilde{\mathbf{X}}\mathbf{W}\mathbf{Y}, \;\; \mathbf{B} = \widehat{\mathbf{X}}\mathbf{W}\mathbf{Y}, \tag{3.7}$$

where

$$\mathbf{Y} = \left\{e^{-icx_m\frac{n'}{N}}\right\}_{\substack{m=1,\ldots M\\ n'=0,\ldots,N}}, \;\; \mathbf{W} = \mathrm{diag}\left(w_1,\ldots,w_M\right),$$

and

$$\widetilde{\mathbf{X}} = \left\{e^{icx_m\frac{\hat{n}}{N}}\right\}_{\substack{\hat{n}=0,\ldots,N-1\\ m=1,\ldots M}}, \;\; \widehat{\mathbf{X}} = \left\{e^{icx_m\frac{\hat{n}}{N}}\right\}_{\substack{\hat{n}=1,\ldots,N\\ m=1,\ldots,M}}.$$

We note that the matrices $\widehat{\mathbf{X}}$ and $\widetilde{\mathbf{X}}$ are related,

$$\widehat{\mathbf{X}} = \widetilde{\mathbf{X}}\mathbf{E},$$

where $\mathbf{E} \in \mathbb{C}^{M\times M}$ is the diagonal matrix,

$$\mathbf{E} = \mathrm{diag}\left(e^{icx_1/N},\ldots,e^{icx_M/N}\right). \tag{3.8}$$

To obtain the set $\left\{e^{icx_m/N}\right\}_{m=1}^{M}$ as eigenvalues of a matrix, we apply the pseudo-inverse of $\mathbf{A}$, $\mathbf{A}^{\dagger}$, to derive the relation

$$\begin{aligned}\mathbf{A}^{\dagger}\mathbf{B} &= (\mathbf{WY})^{\dagger}\,\widetilde{\mathbf{X}}^{\dagger}\widehat{\mathbf{X}}\mathbf{WY} = (\mathbf{WY})^{\dagger}\left(\widetilde{\mathbf{X}}^{\dagger}\widetilde{\mathbf{X}}\right)\mathbf{EWY}\\ &= (\mathbf{WY})^{\dagger}\,\mathbf{E}\,(\mathbf{WY})\,, \qquad (3.9)\end{aligned}$$

using that $\mathbf{WY}$ has full rank and $\widetilde{\mathbf{X}}^{\dagger}\widetilde{\mathbf{X}} = \mathbf{I}_{M\times M}$. Thus, since the non-zero eigenvalues of $(\mathbf{WY})^{\dagger}\,\mathbf{E}\,(\mathbf{WY})$ coincide with those of $\mathbf{E}$, we have shown that the nodes may be obtained by finding the non-zero eigenvalues of $\mathbf{A}^{\dagger}\mathbf{B}$.

To obtain the approximation (3.5), we need to form $\mathbf{A}^{\dagger}\mathbf{B}$ from the matrix $\mathbf{G}$ in (3.4). However, since the matrix $\mathbf{G}$ is extremely ill-conditioned (due to oversampling), we use instead its rank M approximation computed via the SVD,

$$\mathbf{G} = \mathbf{U}\Sigma\mathbf{V}^{*}. \qquad (3.10)$$

Given $\epsilon > 0$, we find the (smallest) index M such that $\sigma_M/\sigma_0 < \epsilon$ and denote by Σ_M the diagonal matrix with the first M singular values, $\Sigma_M = \mathrm{diag}\,(\sigma_0, \sigma_2, \ldots \sigma_{M-1})$. We then truncate (3.10) as

$$\mathbf{G}_M = \mathbf{U}_M\boldsymbol{\Sigma}_M\mathbf{V}_M^{*}, \qquad (3.11)$$

where $\mathbf{U}_M$ and $\mathbf{V}_M$ are the submatrices of the unitary matrices $\mathbf{U}$ and $\mathbf{V}$ containing the first M singular vectors of $\mathbf{G}$. We have

$$\min_{\mathrm{rank}(\mathbf{G}')=M}\left\|\mathbf{G}-\mathbf{G}'\right\|_2 = \left\|\mathbf{G}-\mathbf{G}_M\right\|_2 = \sigma_M.$$

Following (3.7), we write the corresponding matrices $\mathbf{A}_M$ and $\mathbf{B}_M$ as

$$\mathbf{A}_M = \widetilde{\mathbf{U}}_M\boldsymbol{\Sigma}_M\mathbf{V}_M^{*},\;\; \mathbf{B}_M = \widehat{\mathbf{U}}_M\boldsymbol{\Sigma}_M\mathbf{V}_M^{*},$$

where $\widetilde{\mathbf{U}}_M$ and $\widehat{\mathbf{U}}_M$ are the submatrices of $\mathbf{U}_M$,

$$\widetilde{\mathbf{U}}_M = \left\{\mathbf{U}_{\tilde{n}m}\right\}_{\substack{\tilde{n}=0,\ldots,N-1\\ m=0,\ldots,M-1}},\;\; \widehat{\mathbf{U}}_M = \left\{\mathbf{U}_{\hat{n}m}\right\}_{\substack{\hat{n}=1,\ldots,N\\ m=0,\ldots,M-1}}. \qquad (3.12)$$

We note that the truncated version of $\mathbf{A}^{\dagger}\,\mathbf{B}$, $\mathbf{A}_M^{\dagger}\,\mathbf{B}_M$, has the same eigenvalues as $\widetilde{\mathbf{U}}_M^{\dagger}\widehat{\mathbf{U}}_M$,

$$\begin{aligned}\mathbf{A}_M^{\dagger}\,\mathbf{B}_M &= (\mathbf{V}_M^{*})^{\dagger}\,\boldsymbol{\Sigma}_M^{\dagger}\widetilde{\mathbf{U}}_M^{\dagger}\widehat{\mathbf{U}}_M\boldsymbol{\Sigma}_M\mathbf{V}_M^{*}\\ &= (\boldsymbol{\Sigma}_M\mathbf{V}_M^{*})^{\dagger}\,\widetilde{\mathbf{U}}_M^{\dagger}\widehat{\mathbf{U}}_M\boldsymbol{\Sigma}_M\mathbf{V}_M^{*}.\end{aligned}$$

Hence, we define the $M\times M$ matrix $\mathbf{C}_M = \widetilde{\mathbf{U}}_M^{\dagger}\widehat{\mathbf{U}}_M$ and calculate the eigenvalues $\left\{e^{icx_m/N}\right\}_{m=1}^{M}$ and, hence, the nodes $\{x_m\}_{m=1}^{M}$.

The fact that the quadrature nodes for bandlimited exponentials may be found as eigenvalues was also observed by Yu Chen [9].

3.3. Algorithm for computing quadrature nodes.

We describe the algorithm, derived above, for computing quadrature nodes for bandlimited functions given a weight function $w(x)$, bandlimit c, and accuracy ϵ. We address the computation of quadrature weights later in Section 4.

Algorithm 1.

(1) Form the $(N+1)\times(N+1)$ Toeplitz matrix $\mathbf{G}_{kl} = u\left((k-l)/N\right)$, where we choose N such that the function $u(t) = \int_{-1}^{1} e^{ictx}w(x)\,dx$ is sufficiently oversampled.

(2) Take the SVD of $\mathbf{G}$, $\mathbf{G} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$, and select the index M corresponding to the singular value σ_M such that σ_M/σ_0 is close to the desired accuracy ϵ.
(3) Truncate the matrix $\mathbf{U}$ (such that it contains the singular vectors corresponding to the singular values $\sigma_0, \ldots, \sigma_{M-1}$) and form the matrices $\widetilde{\mathbf{U}}_M$ and $\widehat{\mathbf{U}}_M$ from equation (3.12).
(4) Using the pseudo-inverse, form the matrix $\mathbf{C}_M = \widetilde{\mathbf{U}}_M^\dagger \widehat{\mathbf{U}}_M$ and find its eigenvalues, $\left\{e^{icx_m/N}\right\}_{m=1}^M$, from which we extract the nodes x_m, $m = 1, \ldots, M$.

Remark 1. Similar to the algorithms for finding quadratures in [4] and [24], if we compute high accuracy quadratures (e.g., $\epsilon < 10^{-12}$), we need to use extended precision arithmetic in our computations. Once the quadrature nodes and weights are obtained, no extra precision is needed for their use.

Remark 2. Algorithm 1 requires $\mathcal{O}\left(M^3\right)$ operations and is applicable to general weight functions (see examples below).

Remark 3. The explicit introduction of inner products (if applied to the case of decaying exponentials) provides an interpretation of the so-called HSVD [19] or the matrix-pencil method [15, 16, 17] algorithms (that are essentially the same). In our view, our approach simplifies the understanding of these algorithms originally introduced in electrical engineering literature as a sequence of steps similar to those in Algorithm 1.

4. Calculating quadrature weights

We calculate quadrature weights using two different approaches: standard least squares and ℓ^∞ residual minimization. The most straightforward approach is to use least squares. However, we may achieve a better maximum error if we use ℓ^∞ residual minimization. This approach leads us to set up the problem as a second order cone program (since our matrices are complex), and then apply an appropriate solver (see Section 8.2).

4.1. **Finding weights via least squares.** To find the weights w_m, $m = 1, \ldots, M$ that satisfy (3.1), we solve a rectangular Vandermonde system using least squares. The Vandermonde matrix $\mathbf{V} \in \mathbb{C}^{(2N+1)\times M}$ is defined as $\mathbf{V}_{nm} = e^{icx_m n/N}$, where x_m, $m = 1 \ldots M$, are the quadrature nodes, c is the bandlimit parameter and $n = -N, \ldots, N$. We solve the overdetermined system $\mathbf{V}\mathbf{w} = \mathbf{u}$, where $\mathbf{w} = \{w_m\}_{m=1}^M$ is the vector of weights and $\mathbf{u} = \{u_n\}_{n=-N}^N$ is the vector of trigonometric moments

$$u_n = u\left(\frac{n}{N}\right) = \int_{-1}^{1} e^{icx\frac{n}{N}} w(x)\, dx.$$

The performance of our quadrature nodes using least squares weights is illustrated in Table 2 and Figure 6.3(a).

This approach to finding weights is related to the method used in [4] since we also solve a Vandermonde system. However, in [4] the Vandermonde system size may vary between $M \times M$ and $(N+1)\times(N+1)$. The different sizes of the Vandermonde system are due to the knowledge, or lack thereof, of the general location of the nodes. If the nodes are known to belong to a particular subset of the unit circle, all nodes outside of this region are discarded, and the problem may be reduced to solving a smaller (e.g., $M \times M$) Vandermonde system. Since we find only the nodes corresponding to significant weights, we simply seek the least squares solution to the system $\mathbf{V}\mathbf{w} = \mathbf{u}$. Since $\mathbf{V}^*\mathbf{V}$ may be evaluated explicitly yielding a matrix of size $M \times M$, we solve $\mathbf{V}^*\mathbf{V}\mathbf{w} = \mathbf{V}^*\mathbf{u}$.

Remark 4. There is an alternative formulation for computing weights once the nodes $\{x_m\}_{m=1}^M$ are computed. In this approach, we first evaluate

$$S_{kk'} = \int_{-1}^{1} e^{i\frac{c}{2}x_k x} e^{-i\frac{c}{2}x_{k'} x} w(x)dx, \quad k, k' = 1, \ldots, M,$$

and then compute weights $\{w_m\}_{m=1}^M$ minimizing

$$\sum_{k,k'=1}^{M} \left| S_{kk'} - \sum_{m=1}^{M} w_m e^{i\frac{c}{2}x_k x_m} e^{-i\frac{c}{2}x_{k'} x_m} \right|^2$$

via least squares. This formulation avoids using the original oversampled trigonometric moments, which may be useful in some situations.

4.2. **Finding weights via ℓ^∞residual minimization.** In order to minimize the maximum absolute error of the quadrature on the interval of interest, we calculate weights via ℓ^∞ minimization of the residual, $\min_{\mathbf{w}} \|\mathbf{V}\mathbf{w} - \mathbf{u}\|_\infty$ using CVX [13]. Ideally, we would like to obtain the equioscillation property expected of optimal ℓ^∞ minimization. Since we are not optimizing the nodes and weights simultaneously, the error is not perfectly equioscillatory but the maximum error is smaller than that obtained via least squares. Nevertheless, we would like to identify a reason for not achieving the equioscillation property, namely, we would like to rule out a possible collapse of the algorithm for solving the second-order cone program due to ill conditioning of the matrices involved in our computations. For this reason, we implemented a version of the second cone program in Mathematica^{TM}, so that we may use arbitrarily high precision to compare results with those obtained via CVX. In spite of changing the internal precision to up to 64 digits, the error did not change in a significant manner.

We illustrate the performance of a quadrature with weights computed via ℓ^∞ minimization in Figure 6.3(b). We note that, as expected, within the effective bandlimit the quadrature with weights computed via ℓ^∞ minimization of the residual yield a smaller maximum error compared to the quadrature with weights found using least squares (see Figure 6.3(a)). The nodes and weights computed by both, least squares and ℓ^∞ minimization, are displayed in Table 1.

Our results point to the possibility of further improvement by a method that would accommodate a change in the position of the nodes. In [24] that is exactly what is done using an ℓ^2 type minimization. However, developing an approach involving both nodes and weights to obtain the equioscillation property of the error remains an open problem.

5. Examples

5.1. **An example of linear array antenna.** Let us find quadrature nodes for the integral

$$u^{(c)}\left(B,l,\cos\theta\right)=\frac{1}{2}\int_{-1}^{1}I_0\left(\pi B\sqrt{1-\left(\frac{x}{l}\right)^2}\right)e^{icx\cos(\theta)}dx, \tag{5.1}$$

where c is the bandlimit and I_0 is the modified Bessel function of order zero. This integral arises in antenna design and, for parameters $l=1$ and $B=1$, a quadrature for (5.1) is computed in [7, Eq. 6.7] by a different approach. However, our approach is simpler and yields similar results. Given the weight function

$$w\left(x\right)=I_0\left(\pi\sqrt{1-x^2}\right), \tag{5.2}$$

we obtain its trigonometric moments as

$$u_n^{(c)}=\frac{1}{2}\int_{-1}^{1}e^{icxn/N}w\left(x\right)dx=\operatorname{sinc}\left(\sqrt{(c\frac{n}{N})^2-\pi^2}\right),\quad n=-N,\dots N, \tag{5.3}$$

corresponding (up to a factor) to the samples of the radiation pattern. Identity (5.3) may be obtained extending formula 6.616.5 in [12, p. 698]. We also note that the weight function (5.2) is a scaled version of the so-called Kaiser window (see e.g. [18]).

We form

$$\mathbf{G}_{nn'}=u_{n-n'}^{(c)},\quad n,n'=0,\dots N,$$

with $N=252$ and $c=10\pi$, and use Algorithm 1 in Section 3.3. We truncate the SVD of the matrix $\mathbf{G}$ at the (normalized) singular value σ_{22}, $\sigma_{22}/\sigma_0\approx1.2\cdot10^{-15}$, yielding 22 quadrature nodes. Using the ℓ^∞ residual minimization (see Section 4.2), we compute the weights resulting in a quadrature with maximum absolute error $\epsilon=1.21\cdot10^{-14}$. We verify the accuracy of this quadrature numerically and illustrate the result in Figure 5.1. This quadrature should be compared with that corresponding to the bandlimit 20π in [7, Table 6.3] since we integrate on $[-1,1]$ instead of $[-1/2,1/2]$ as in [7].

5.2. **Non-sign-definite example.** We demonstrate that our method yields quadratures for weight functions w that are not sign-definite. For the weight function

$$w(x)=(x-1/10)\cdot e^{-(3\pi x/5-1/5)^2}+1/(5e), \tag{5.4}$$

we calculate the nodes and weights for the bandlimit $c=5\pi$, choosing $N=127$ and the singular value $\sigma_{14}/\sigma_0=5.0\cdot10^{-14}$. Figure 5.2(a) illustrates the weight function $w(x)$, $x\in[-1,1]$, and Figure 5.2(b) shows that the weights of the quadrature follow the shape of the weight function $w(x)$. The error of the quadrature with 14 nodes and weights is illustrated in Figure 5.2(c), where the maximum error is $6.68\cdot10^{-14}$.

We note that the approach in [4] also allows us to obtain quadratures for weight functions $w(x)$ that are not sign-definite as is shown in [2].

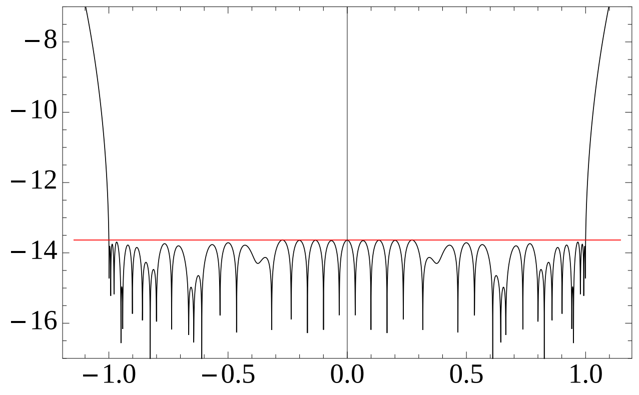

FIGURE 5.1. The logarithm of the error of the quadrature with 22 nodes ($c = 10\pi$) for the weight function (5.2). The quadrature weights were generated via ℓ^∞ minimization. The horizontal line at $2.32 \cdot 10^{-14}$ indicates the maximum ℓ^∞ error within the bandwidth.

6. Comparison with quadratures in [4] and [24]

Let us illustrate the impact of using weights obtained via ℓ^∞ minimization for the nodes computed in [24] and [4]. For this comparison we choose the weight function $w = 1$. In Table 2 we display the errors of these quadratures and compare them to the quadratures of this paper. Our quadratures yield a slightly better error than those of both [24] and [4]. In Table (3) we compare the maximum errors using different approaches to computing weights.

Next, Figure 6.1 compares the error of quadratures using nodes and weights from [24] and the same nodes but with weights found via ℓ^∞ minimization. A similar comparison for the quadratures from [4] is provided in Figure 6.2. As expected, in all cases the ℓ^∞ minimization produces a better maximum error.

Remark 5. We observe that it is possible to obtain equioscillatory behavior of the error by minimizing the ℓ^2 norm of the weights constrained by an error bound on the ℓ^∞ residual. The result of solving the optimization problem

$$\min \|\mathbf{w}\|_2 \quad \text{subject to} \quad \|\mathbf{V}\mathbf{w} - \mathbf{u}\|_\infty < \epsilon$$

is illustrated in Figure 6.4. However, the attained maximum error is significantly worse than using all other approaches to compute weights.

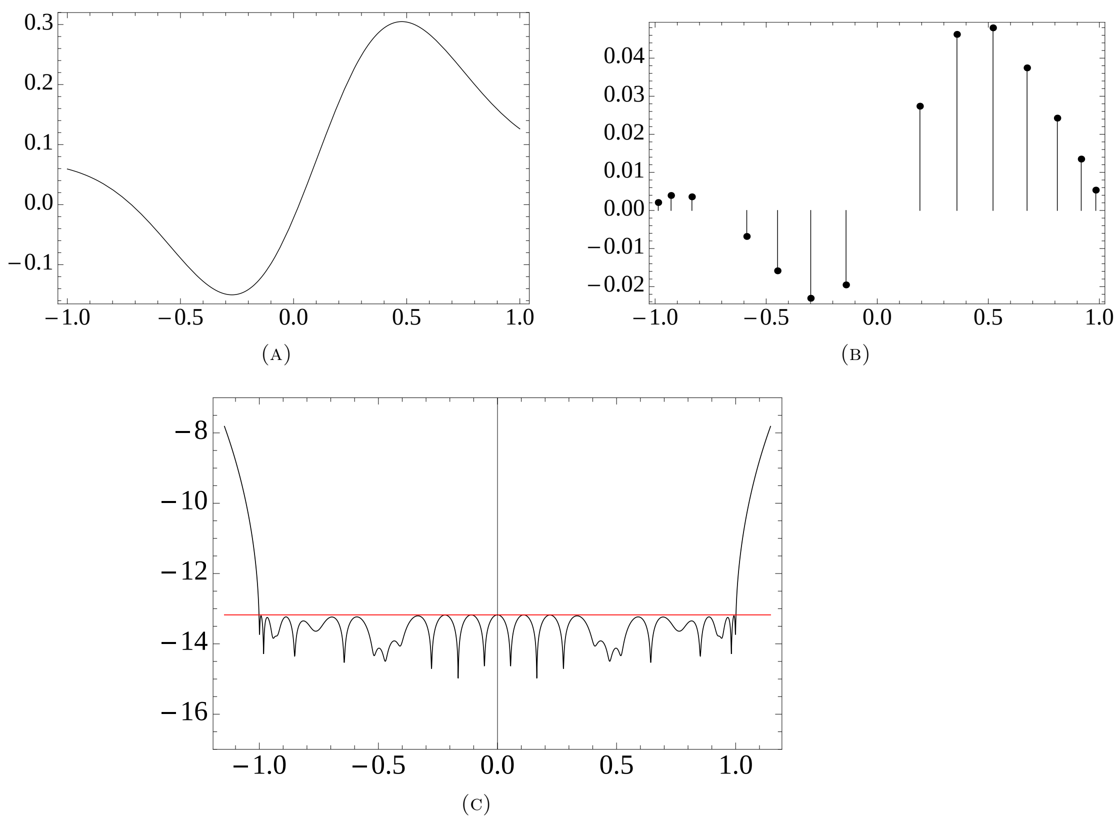

FIGURE 5.2. (a) The weight function w in (5.4) and (b) the corresponding quadrature weights computed via ℓ^∞-minimization. We note that the quadrature weights follow the shape of the weight function w. In (c) we display the logarithm of the error of the quadrature with 14 nodes ($c = 5\pi$). The horizontal line at $6.68 \cdot 10^{-14}$ indicates the maximum ℓ^∞-error within the bandwidth.

7. Conclusions

In this paper we introduced a new algorithm for finding quadrature nodes for bandlimited exponentials and considered two different approaches to compute the corresponding quadrature weights. As in [4], the accuracy of these quadratures is parametrized by the singular values of the Toeplitz matrix formed from the trigonometric moments of the weight function.

The two methods of finding weights used in this paper solve a rectangular Vandermonde system by minimizing a residual, either in the ℓ^2 or ℓ^∞ sense. This differs from [4], where such Vandermonde systems are square.

The new quadratures are slightly more accurate than those in [4] and [24], but their computation is currently more expensive. The new algorithm always produces a number of nodes that coincides with the index of the chosen singular value and,

Quadrature nodes and weights for $c = 50$		
Nodes	ℓ^2 min weights	ℓ^∞ min weights
0.05098496373726	$1.0194136874164 \cdot 10^{-1}$	$1.0194136790749 \cdot 10^{-1}$
0.15278216715085	$1.0159361655411 \cdot 10^{-1}$	$1.0159361762279 \cdot 10^{-1}$
0.25404711706787	$1.0086951579866 \cdot 10^{-1}$	$1.0086951557538 \cdot 10^{-1}$
0.35437535428814	$9.9706360031823 \cdot 10^{-2}$	$9.9706360549662 \cdot 10^{-2}$
0.45327769114752	$9.7994451679077 \cdot 10^{-2}$	$9.7994451352478 \cdot 10^{-2}$
0.55012209105782	$9.5552252896549 \cdot 10^{-2}$	$9.5552251399310 \cdot 10^{-2}$
0.64404102192821	$9.2079974254652 \cdot 10^{-2}$	$9.2079975898033 \cdot 10^{-2}$
0.73377426101324	$8.7072622729206 \cdot 10^{-2}$	$8.7072622960480 \cdot 10^{-2}$
0.81739106203437	$7.9658787303857 \cdot 10^{-2}$	$7.9658787375413 \cdot 10^{-2}$
0.89179797135367	$6.8331342878393 \cdot 10^{-2}$	$6.8331340338988 \cdot 10^{-2}$
0.95196091437069	$5.0710205180187 \cdot 10^{-2}$	$5.0710208528588 \cdot 10^{-2}$
0.99030088410242	$2.4489489924317 \cdot 10^{-2}$	$2.4489489733714 \cdot 10^{-2}$

TABLE 1. Quadrature nodes and weights for $w(x) = 1$ and $c = 50$. The weights are found either via ℓ^2 or ℓ^∞ minimization. Since the weight is symmetric about the origin, we only display the nodes in $[0, 1]$ and their corresponding weights.

		Maximum error from:		Maximum error using:	
c	# of nodes	[4]	[24]	ℓ^2min weights	ℓ^∞min weights
20	13	$1.2 \cdot 10^{-7}$	$9.4 \cdot 10^{-8}$	$3.8 \cdot 10^{-8}$	$3.5 \cdot 10^{-8}$
50	24	$1.2 \cdot 10^{-7}$	$8.3 \cdot 10^{-8}$	$3.0 \cdot 10^{-8}$	$2.3 \cdot 10^{-8}$
100	41	$1.6 \cdot 10^{-7}$	$9.1 \cdot 10^{-8}$	$2.7 \cdot 10^{-8}$	$2.3 \cdot 10^{-8}$
200	74	$1.8 \cdot 10^{-7}$	$8.6 \cdot 10^{-8}$	$2.7 \cdot 10^{-8}$	$2.1 \cdot 10^{-8}$
500	171	$1.4 \cdot 10^{-7}$	$8.8 \cdot 10^{-8}$	$2.7 \cdot 10^{-8}$	$2.0 \cdot 10^{-8}$
1000	331	$2.4 \cdot 10^{-7}$	$1.4 \cdot 10^{-7}$	$4.0 \cdot 10^{-8}$	$3.1 \cdot 10^{-8}$
2000	651	$1.2 \cdot 10^{-7}$	$6.4 \cdot 10^{-8}$	$2.6 \cdot 10^{-8}$	*
4000	1288	$3.7 \cdot 10^{-7}$	$1.7 \cdot 10^{-7}$	$3.2 \cdot 10^{-8}$	*

TABLE 2. Performance of quadratures for various bandlimits. (*) The ℓ^∞ minimization algorithm could not calculate weights in these cases due to the size of the Vandermonde systems.

in our experience, the nodes are always located inside the support of the weight function.

8. APPENDIX

8.1. **Golub-Welsch algorithm.**

We show how to derive the well known Golub-Welsch algorithm [11] using the results in Section 3.1. Let us to consider a subspace of polynomials spanned by the orthogonal basis $\{p_n(x)\}_{n=1}^N$. For such a set, there exists a three term recursion relation of the form

$$p_{n+1}(x) = (a_{n+1}x + b_{n+1})\, p_n(x) - c_{n+1} p_{n-1}(x), \tag{8.1}$$

	Maximum error with nodes from:		
Weights	[4]	[24]	this paper
From [4] and [24]	$1.2 \cdot 10^{-7}$	$8.3 \cdot 10^{-8}$	
Via ℓ^∞ minimization	$7.8 \cdot 10^{-8}$	$5.3 \cdot 10^{-8}$	$2.4 \cdot 10^{-8}$
Via ℓ^2 minimization			$2.8 \cdot 10^{-8}$

TABLE 3. Comparison of maximum absolute errors using the 24 nodes of different quadratures for fixed bandlimit $c = 50$. We compare the maximum error from the original references [4] and [24] to the maximum error using the same nodes but weights computed via ℓ^∞ minimization. We also compute the maximum errors of the quadratures of this paper with weights obtained via ℓ^2 and ℓ^∞ minimization.

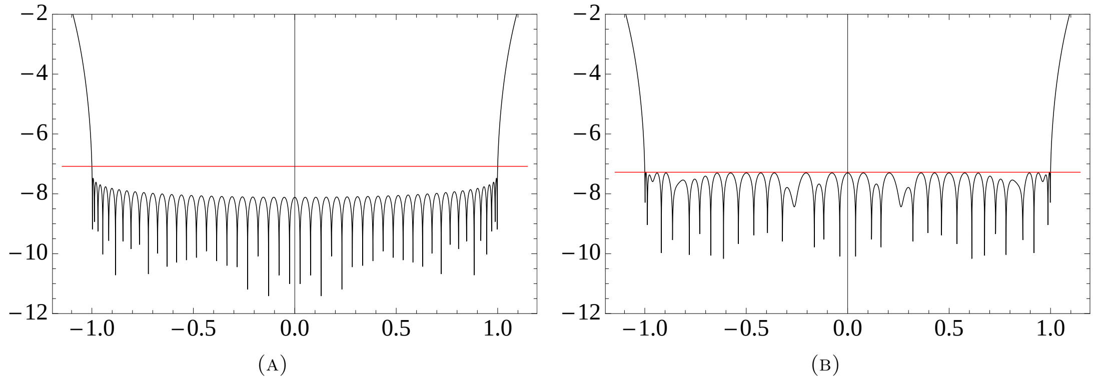

FIGURE 6.1. Logarithm of the error of the 24 node quadrature for bandlimit $c = 50$. In (a) we use nodes and weights from [24] and in (b) nodes from [24] and weights generated via ℓ^∞ minimization. The horizontal lines at $8.30 \cdot 10^{-8}$ in (a) and at $5.26 \cdot 10^{-8}$ in (b) indicate the maximum ℓ^∞ error within the bandwidth.

where $p_{-1}(x) \equiv 0$, $p_0 \equiv 1$, $a_n > 0$ and $c_n > 0$ for $n = 0, \dots, N-1$. Following [11], we write the recursion as the matrix equation

$$x\,\mathbf{p}(x) = \mathbf{T}\mathbf{p}(x) + (1/a_n)\,p_N(x)\,\mathbf{e}_n, \tag{8.2}$$

where $\mathbf{p}(x) = [p_0(x), \dots, p_{N-1}(x)]^t$, $\mathbf{e}_n = [0, \dots, 1]^t$, and

$$T = \begin{pmatrix} -b_1/a_1 & 1/a_1 & 0 & \dots & \\ c_2/a_2 & -b_2/a_2 & 1/a_2 & & \\ 0 & \ddots & \ddots & \ddots & \\ \vdots & & & & 1/a_{N-1} \\ & & & c_N/a_N & -b_N/a_N \end{pmatrix}.$$

Due to (8.2), $p_N(x_j) = 0$ if and only if x_j is an eigenvalue of $\mathbf{T}$, i.e., $\mathbf{T}\mathbf{p}(x_j) = x_j\mathbf{p}(x_j)$. Hence, we may recover the quadrature nodes x_j by solving the eigenvalue

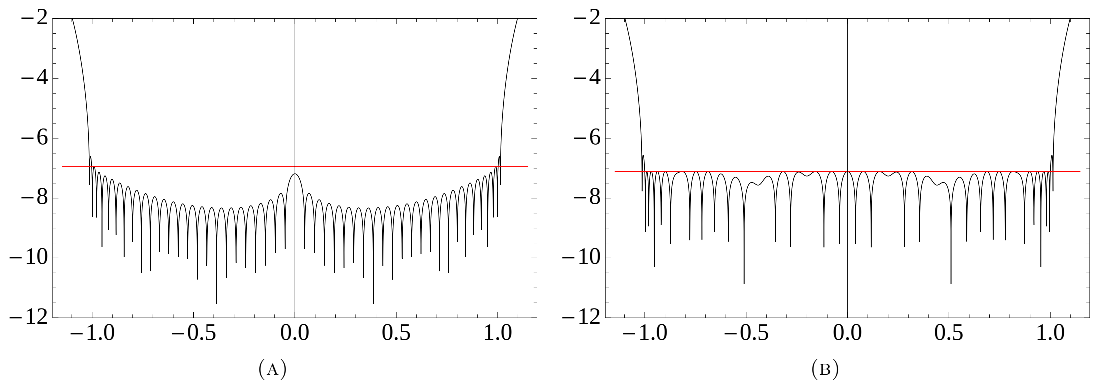

FIGURE 6.2. Logarithm of the error of the 24 node quadrature for bandlimit $c = 50$. In (a) we use nodes and weights from [4] and in (b) nodes from [4] and weights generated via ℓ^∞ minimization. The horizontal lines at $1.15 \cdot 10^{-7}$ in (a) and at $7.76 \cdot 10^{-8}$ in (b) indicate the maximum ℓ^∞ error within the bandwidth.

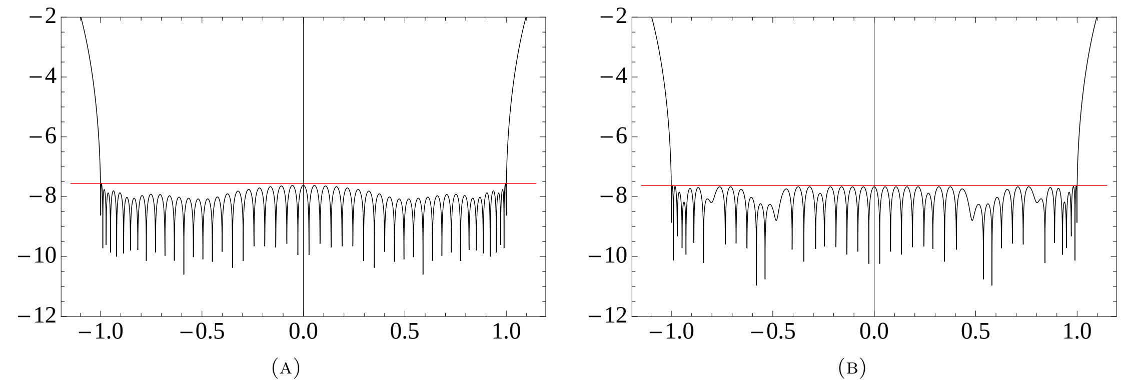

FIGURE 6.3. Logarithm of the error of the 24 node quadrature for bandlimit $c = 50$ of this paper. In (a) we show the error using weights obtained via ℓ^2 minimization and in (b) using weights obtained via ℓ^∞ minimization. The horizontal lines at $2.80 \cdot 10^{-8}$ in (a) and at $2.36 \cdot 10^{-8}$ in (b) indicate the maximum ℓ^∞ error within the bandwidth.

problem for T. We note that in the Golub-Welsch algorithm the quadrature weights are found from the eigenvectors of the matrix T.

We now show how to derive the matrix T using the approach in Section 3.1, that is, via matrices of inner products. We define the matrices $\mathbf{A}$ and $\mathbf{B}$ by

$$\mathbf{A}_{ij} = \langle p_i, p_j \rangle = \int_{-1}^{1} p_i(x)\; p_j(x)\; w(x)\; \mathrm{d}x, \quad \mathbf{B}_{ij} = \int_{-1}^{1} p_i(x)\; x\, p_j(x)\; w(x)\; \mathrm{d}x,$$

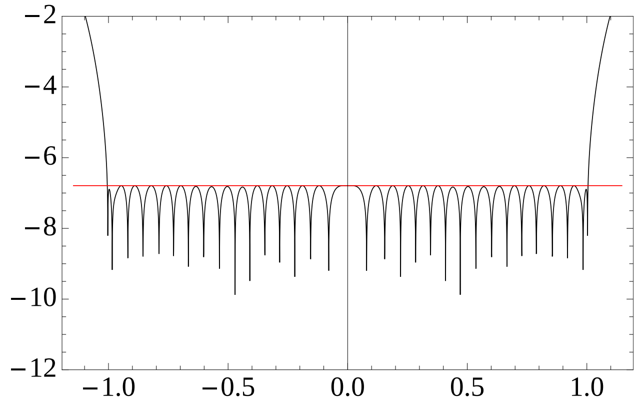

FIGURE 6.4. Logarithm of the error of the 24 node quadrature for bandlimit $c = 50$ with weights generated by minimizing an ℓ^2 residual with an ℓ^∞ constraint. Although the error has a near perfect behavior, the maximum absolute error $1.62 \cdot 10^{-7}$ is worse than in Figures (6.1), (6.2) and (6.3).

where $i, j \in \{0, \ldots, N-1\}$ and $w(x)$ is the weight function of the associated inner product. The three term recursion relation (8.1) implies that the matrix $\mathbf{B}$ is tridiagonal. In fact, because of (8.1), $x\, p_n(x)$ is a linear combination of p_{n-1}, p_n and p_{n+1} which gives

$$\mathbf{B}_{n\,n+2} = 0, \quad n = 0, \ldots, N-3,$$

and

$$\begin{aligned}
\mathbf{B}_{n\,n+1} &= \int_{-1}^{1} \left(\frac{p_{n+1}(x)}{a_{n+1}} \right) p_{n+1}(x)\, w(x)\, \mathrm{d}x \\
&= \frac{\|p_{n+1}\|^2}{a_{n+1}}, \quad n = 0, \ldots, N-2, \\
\mathbf{B}_{n\,n} &= \int_{-1}^{1} \left(-\frac{b_{n+1}}{a_{n+1}} p_n(x) \right) p_n(x)\, w(x)\, \mathrm{d}x \\
&= -\frac{b_{n+1}}{a_{n+1}} \|p_n\|^2, \quad n = 0, \ldots, N-1, \\
\mathbf{B}_{n\,n-1} &= \int_{-1}^{1} \left(\frac{c_{n+1}}{a_{n+1}} p_{n-1}(x) \right) p_{n-1}(x)\, w(x)\, \mathrm{d}x \\
&= \frac{c_{n+1}}{a_{n+1}} \|p_{n-1}\|^2, \quad n = 1, \ldots, N-1,
\end{aligned}$$

where $\|\cdot\|$ is the norm associated with the weight function $w(x)$. Furthermore, since $\{p_n(x)\}_{n=1}^{N}$ are orthogonal, we obtain $\mathbf{A} = \operatorname{diag}\left(\|p_0\|^2, \ldots, \|p_{N-1}\|^2 \right)$. Taking the inverse of $\mathbf{A}$, we recover the tridiagonal matrix $\mathbf{T}$ from the Golub-Welsch algorithm,

$$\mathbf{T} = \mathbf{B}\mathbf{A}^{-1}.$$

Note that our approach is more general since it may be applied to any basis $\{p_n(x)\}_{n=1}^N$, even if it is not orthogonal (no 3-term recurrence is available); it also generalizes to other sets of functions or non-positive weights.

8.2. **Formulation of ℓ^∞ residual minimization as a second-order cone program.** We review the primal-dual interior-point method of [20], the algorithm we implemented in extended precision to compare with the results obtained using CVX [13]. For further details we refer to [6].

We define a second order cone program (SOCP) as

$$\begin{aligned} &\text{minimize} \quad \mathbf{f}^t\mathbf{x} \\ &\text{subject to} \quad \|\mathbf{u}_i\| \le t_i, \qquad i=1,\dots,N, \\ &\qquad \mathbf{u}_i = \mathbf{A}_i\mathbf{x}+\mathbf{b}_i, \quad i=1,\dots,N, \\ &\qquad t_i = \mathbf{c}_i^T\mathbf{x}+d_i, \quad i=1,\dots,N, \end{aligned} \tag{8.3}$$

where $\mathbf{x}\in\mathbb{R}^n$ is the optimization variable and $\mathbf{f}\in\mathbb{R}^n$, $\mathbf{A}_i\in\mathbb{R}^{n_i\times n}$, $\mathbf{b}_i\in\mathbb{R}^{n_i}$, $\mathbf{c}_i\in\mathbb{R}^n$, and $d_i\in\mathbb{R}$ are the problem parameters. The primal-dual interior-point method simultaneously solves the SOCP and a dual problem, defined as

$$\begin{aligned} &\text{maximize} \quad -\sum_{i=1}^N \left(\mathbf{b}_i^t\mathbf{z}_i + d_i w_i\right) \\ &\text{subject to} \quad \sum_{i=1}^N \left(\mathbf{A}_i^t\mathbf{z}_i+\mathbf{c}_i w_i\right) = \mathbf{f}, \quad i=1,\dots,N, \\ &\qquad \|\mathbf{z}_i\|_2 \le w_i, \qquad i=1,\dots,N \end{aligned} \tag{8.4}$$

where $\mathbf{z}_i\in\mathbb{R}^{n_i}$ and $w_i\in\mathbb{R}$, $i=1,\dots,N$, are the dual optimization variables. The dual problem is convex, since we maximize a concave function subject to convex constraints. Next, we demonstrate how the ℓ^∞ residual minimization problem can be recast as a SOCP.

8.2.1. *Casting the ℓ^∞ residual minimization problem as a SOCP.* We need to find the solution of the ℓ^∞ residual minimization problem $\min_{\mathbf{w}}\|\mathbf{Az}-\mathbf{b}\|_\infty$, where $\mathbf{A}\in\mathbb{C}^{p\times q}$ and $\mathbf{b}\in\mathbb{C}^p$. We define

$$\mathbf{x} = \begin{bmatrix} \mathrm{Re}(\mathbf{z}) \\ \mathrm{Im}(\mathbf{z}) \\ t \end{bmatrix} \in \mathbb{R}^{2q+1}, \quad \mathbf{f}=\mathbf{c}_i = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \in \mathbb{R}^{2q+1},$$

where we introduce the new optimization variable $t>0$. We define

$$\mathbf{A}_i = \begin{bmatrix} \mathrm{Re}(\mathbf{a}_i) & -\mathrm{Im}(\mathbf{a}_i) & 0 \\ \mathrm{Im}(\mathbf{a}_i) & \mathrm{Re}(\mathbf{a}_i) & 0 \end{bmatrix} \in \mathbb{R}^{(2q+1)\times 2}, \quad \mathbf{b}_i = \begin{bmatrix} \mathrm{Re}(b_i) \\ \mathrm{Im}(b_i) \end{bmatrix} \in \mathbb{R}^2,$$

for $i=1,\dots,p$, where $\mathbf{a}_i$ is the i-th row of the matrix $\mathbf{A}$, and b_i is the i-th entry of the vector $\mathbf{b}$. Substituting these definitions into (8.3) and setting $d_i=0$, $i=1,\dots,p$, yields the SOCP for solving the ℓ^∞ residual minimization problem,

$$\begin{aligned} &\text{minimize} \quad t \\ &\text{subject to} \quad \left\| \begin{bmatrix} \mathrm{Re}(\mathbf{a}_i) & -\mathrm{Im}(\mathbf{a}_i) & 0 \\ \mathrm{Im}(\mathbf{a}_i) & \mathrm{Re}(\mathbf{a}_i) & 0 \end{bmatrix} \mathbf{x} - \begin{bmatrix} \mathrm{Re}(b_i) \\ \mathrm{Im}(b_i) \end{bmatrix} \right\|_2 \le t, \quad i=1,\dots,p. \end{aligned}$$

8.2.2. *Primal-dual interior-point method.* The primal-dual interior-point algorithm solves (8.3) by minimizing the difference between the primary and the dual objective functions, known as the duality gap,

$$\eta\left(\mathbf{x},\mathbf{z},\mathbf{w}\right)=\mathbf{f}^{t}\mathbf{x}+\sum_{i=1}^{N}\left(\mathbf{b}_{i}^{t}\mathbf{z}_{i}+d_{i}w_{i}\right).$$

This gap is non-negative for feasible $\mathbf{x},\mathbf{z},\mathbf{w}$. Considering strictly feasible primal and dual problems (i.e., the inequalities in (8.3) and (8.4) are replaced by strict inequalities), we know that there exists solutions where the duality gap $\eta\left(\mathbf{x},\mathbf{z},\mathbf{w}\right)=0$. Such a solution achieves the optimum value (see e.g. [20]). While we provide an initial guess that is strictly feasible, we also need to enforce strict feasibility of the iterates. To this end, we define the barrier function $\phi\left(x,t\right)$,

$$\phi\left(\mathbf{u},t\right)=\begin{cases}-log\left(t^{2}-\left\Vert \mathbf{u}\right\Vert _{2}^{2}\right), & \left\Vert \mathbf{u}\right\Vert _{2}<t\\ \infty & otherwise\end{cases},$$

which approaches infinity as $\left\Vert \mathbf{u}\right\Vert _{2}^{2}\rightarrow t^{2}$, corresponding in the limit to a feasible (but not strictly feasible) solution of the problem.

Using the duality gap η and barrier functions for both of the primal and dual problems, we define the potential function

$$\varphi\left(\mathbf{x},\mathbf{z},\mathbf{w}\right)=\left(2N+\nu\sqrt{2N}\right)\log\eta\left(\mathbf{x},\mathbf{z},\mathbf{w}\right)+\sum_{i=1}^{N}\left(\phi\left(\mathbf{u}_{i},t_{i}\right)+\phi\left(\mathbf{z}_{i},w_{i}\right)\right)-2N\log N,$$

where $\nu\geq1$ is a parameter. This potential function satisfies

$$\eta\left(\mathbf{x},\mathbf{z},\mathbf{w}\right)\leq\exp\left(\varphi\left(\mathbf{x},\mathbf{z},\mathbf{w}\right)/\left(\nu\sqrt{2N}\right)\right),$$

for strictly feasible $\left(\mathbf{x},\mathbf{z},\mathbf{w}\right)$. Therefore, if $\varphi\rightarrow-\infty$ then $\eta\rightarrow0$ and the primal-dual algorithm converges. Furthermore, the strict feasibility of the initial guess and each of the iterates guarantees that the value of $\varphi\left(\mathbf{x},\mathbf{z},\mathbf{w}\right)$ decreases by some finite amount after each update (see [20]).

To minimize $\varphi\left(\mathbf{x},\mathbf{z},\mathbf{w}\right)$, we find the primal and dual search directions, $\delta\mathbf{x},\delta\mathbf{z}$ and $\delta\mathbf{w}$ by solving the linear system

$$\left[\begin{array}{cc}\mathbf{H}^{-1} & \overline{\mathbf{A}}\\ \overline{\mathbf{A}}^{t} & 0\end{array}\right]\left[\begin{array}{c}\delta\mathbf{Z}\\ \delta\mathbf{x}\end{array}\right]=\left[\begin{array}{c}-\mathbf{H}^{-1}\left(\rho\mathbf{Z}+\mathbf{g}\right)\\ 0\end{array}\right], \tag{8.5}$$

where $\rho=2N+\nu\sqrt{2N}$,

$$\mathbf{H}=\left[\begin{array}{ccc}\nabla^{2}\phi\left(\mathbf{u}_{1},t_{1}\right) & \dots & 0\\ \vdots & \ddots & \vdots\\ 0 & \dots & \nabla^{2}\phi\left(\mathbf{u}_{N},t_{N}\right)\end{array}\right],\quad\mathbf{g}=\left[\begin{array}{c}\nabla\phi\left(\mathbf{u}_{1},t_{1}\right)\\ \vdots\\ \nabla\phi\left(\mathbf{u}_{N},t_{N}\right)\end{array}\right],$$

and

$$\overline{\mathbf{A}}=\left[\begin{array}{c}\mathbf{A}_{1}\\ \mathbf{c}_{1}^{t}\\ \mathbf{A}_{2}\\ \mathbf{c}_{2}^{t}\\ \vdots\end{array}\right],\quad\mathbf{Z}=\left[\begin{array}{c}\mathbf{z}_{1}\\ w_{1}\\ \mathbf{z}_{2}\\ w_{2}\\ \vdots\end{array}\right],\quad\delta\mathbf{Z}=\left[\begin{array}{c}\delta\mathbf{z}_{1}\\ \delta w_{1}\\ \delta\mathbf{z}_{2}\\ \delta w_{2}\\ \vdots\end{array}\right],\quad\delta\mathbf{x}=\left[\begin{array}{c}\delta\mathbf{x}_{1}\\ \vdots\\ \delta\mathbf{x}_{n}\end{array}\right].$$

Finally, we state the algorithm. Given strictly feasible initial points $(\mathbf{x}, \mathbf{z}, \mathbf{w})$, a tolerance $\epsilon > 0$, and the parameter $\nu \geq 1$, we

(1) Solve equation (8.5) for the primal and dual search directions.
(2) Perform a plane search to find the (p, q) that minimize $\varphi(\mathbf{x} + p\delta\mathbf{x},\ \mathbf{z} + q\delta\mathbf{z},\ \mathbf{w} + q\delta\mathbf{w})$.
(3) Update $\mathbf{x} = \mathbf{x} + p\delta\mathbf{x}$, $\mathbf{z} = \mathbf{z} + q\delta\mathbf{z}$, and $\mathbf{w} = \mathbf{w} + q\delta\mathbf{w}$ as long as $\eta(\mathbf{x}, \mathbf{z}, \mathbf{w}) > \epsilon$.

We note that as η decreases in size the system of equations (8.5) becomes ill conditioned, which results in indeterminate search directions.

References

[1] C. Ahrens and G. Beylkin. Rotationally Invariant Quadratures for the Sphere. *Proceedings of the Royal Society A*, 465:3103–3125, 2009.

[2] G. Beylkin, C. Kurcz, and L. Monzón. Grids and transforms for band-limited functions in a disk. *Inverse Problems*, 23(5):2059–2088, 2007.

[3] G. Beylkin, C. Kurcz, and L. Monzón. Fast convolution with the free space Helmholtz Green's function. *J. Comp. Phys.*, 228(8):2770–2791, 2009.

[4] G. Beylkin and L. Monzón. On generalized Gaussian quadratures for exponentials and their applications. *Appl. Comput. Harmon. Anal.*, 12(3):332–373, 2002.

[5] G. Beylkin and K. Sandberg. Wave propagation using bases for bandlimited functions. *Wave Motion*, 41(3):263–291, 2005.

[6] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge, 2004.

[7] J. Bremer, Z. Gimbutas, and V. Rokhlin. A nonlinear optimization procedure for generalized Gaussian quadratures. *SIAM Journal of Scientific Computing*, 32(4):1761–1788, 2010.

[8] Q.-Y. Chen, D. Gottlieb, and J. S. Hesthaven. Spectral methods based on prolate spheroidal wave functions for hyperbolic PDEs. *SIAM J. Numer. Anal.*, 43(5):1912–1933, 2005.

[9] Yu Chen. Inner Product Quadratures. Technical report, Courant Institute, NYU, 2011. http://arxiv.org/abs/1205.0601.

[10] A. Glaser, X. Liu, and V. Rokhlin. A fast algorithm for the calculation of the roots of special functions. *SIAM Journal on Scientific Computing*, 29(4):1420–1438, 2007.

[11] G. Golub and J. Welsch. Calculation of gaussian quadrature rules. *Mathematics of Computation*, 23:221–230, 1969.

[12] I.S. Gradshteyn and I. M. Ryzhik. *Table of Integrals, Series, and Products.* Elsevier, 7th edition, 2007.

[13] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 1.21. http://cvxr.com/cvx, February 2011.

[14] F. A. Grünbaum. Differential operators commuting with convolution integral operators. *J. Math. Anal. Appl.*, 91(1):80–93, 1983.

[15] Y. Hua and T.K. Sarkar. Matrix pencil method and its performance. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 1988.

[16] Y. Hua and T.K. Sarkar. Matrix pencil method for estimating parameters of exponentially damped/undamped sinusoids in noise. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38(5):814–824, 1990.

[17] Y. Hua and T.K. Sarkar. On SVD for estimating generalized eigenvalues of singular matrix pencil in noise. *IEEE Transactions on Signal Processing*, 39(4):892–900, 1991.

[18] J.F. Kaiser and R.W. Schafer. On the Use of the I_0-Sinh Window for Spectrum Analysis. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(1):105–107, 1980.

[19] S.Y. Kung, K.S. Arun, and D.V. Bhaskar Rao. State-space and singular-value decomposition-based approximation methods for the harmonic retrieval problem. *Journal of the Optical Society of America*, 73(12):1799–1811, 1983.

[20] M.S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret. Applications of second-order cone programming. *Linear Algebra and its Applications*, 284:193–228, 1988.

[21] K. Sandberg and K.J. Wojciechowski. The EPS method: A new method for constructing pseudospectral derivative operators. *J. Comp. Phys.*, 230(15):5836–5863, 2011.

[22] D. Slepian. Prolate spheroidal wave functions, Fourier analysis and uncertainty V. The discrete case. *Bell System Tech. J.*, 57:1371–1430, 1978.

[23] D. Slepian and H. O. Pollak. Prolate spheroidal wave functions, Fourier analysis and uncertainty I. *Bell System Tech. J.*, 40:43–63, 1961.

[24] H. Xiao, V. Rokhlin, and N. Yarvin. Prolate spheroidal wavefunctions, quadrature and interpolation. *Inverse Problems*, 17(4):805–838, 2001.

Department of Applied Mathematics, University of Colorado, Boulder, CO 80309-0526, United States