Nonlinear approximations in tomography, quadrature construction, and multivariate reductions

by

Matthew Reynolds

B.S., University of Colorado, 2003.

M.S., University of Colorado, 2006.

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Applied Mathematics

2012

This thesis entitled:
Nonlinear approximations in tomography, quadrature construction, and multivariate reductions
written by Matthew Reynolds
has been approved for the Department of Applied Mathematics

Gregory Beylkin

Gunnar Martinsson

Date

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

Reynolds, Matthew (Ph.D., Applied Mathematics)

Nonlinear approximations in tomography, quadrature construction, and multivariate reductions

Thesis directed by Professor Gregory Beylkin

This thesis consists of contributions to three topics: algorithms for computing generalized Gaussian quadratures, tomographic imaging algorithms, and reduction algorithms. Our approach is based on using non-linear approximations of functions.

We develop a new algorithm for constructing generalized Gaussian quadratures for exponentials integrated against a non-sign-definite weight function. These quadratures integrate band-limited exponentials to a user-defined accuracy. We also introduce a method of computing quadrature weights via ℓ^∞ minimization.

Second, we develop a new imaging algorithm for X-ray tomography. This algorithm, polar quadrature inversion, uses rational approximations to approximate tomographic projections with a near optimal number of terms for a given accuracy. This rational signal model allows us to augment the measured data by extending the tomographic projection's domain in Fourier space. As the extended data from all the projections fill a disk in the Fourier domain, we use polar quadratures for band-limited exponentials and the unequally spaced fast Fourier transform to obtain our image. We demonstrate that the resulting images have significantly improved resolution without additional artifacts near sharp transitions.

Finally, we develop an extension of existing reduction algorithms for functions of one variable to functions of many variables. By reduction, we understand an approximation (to a user-supplied accuracy) of a linear combination of decaying exponentials by a representation of the same form but with a minimal number of terms. While for functions of one variable there is an underlying theory based on the analysis of functions of one complex variable, no such theory is available for the multivariate case. Our approach is a first step in the development of such theory. We demonstrate our algorithm on two examples of multivariate functions, a suboptimal linear combination of real-valued, decaying exponentials, and that of complex-valued, decaying exponentials.

Acknowledgements

First, I thank Gregory Beylkin, not only for all his guidance but also for the very interesting World War II history discussions over the years! I would like to thank the members of our research group, past and present: Lucas Monzón, Terry Haut, Chris Kurcz, Fernando Pérez, Dave Biagioni, Ryan Lewis, Cory Ahrens, and Kristian Sandberg. You have all helped me so much along the way and there is no way I could have done it without you! Also, I would like to thank Adam Norris and Keith Julien for their friendship and help. You guys were the first two friends I made in the department!

I am grateful for my fantastic family who supported me with love and strength over the years. In particular, I would like to thank my mom for being my inspiration, my twin sister Mandy for always challenging yet supporting me, and Heidy for all her hard, hard, work and love.

Finally, I would like to express my gratitude to all the amazing friends that I have made along the way. You guys keep me going. This is especially true for Mr. (soon to be Dr.) Douglas Baldwin. Your support, kindness, and affinity for long lunches have made this whole graduate school thing a great experience! Also, thank you for your help editing this thesis!

Contents

List of Tables

Table

List of Figures

Figure

Chapter 1

Introduction

In this thesis we apply nonlinear approximations to problems of computing quadratures for band-limited exponentials, reduction of multivariate linear combinations of exponentials, and tomographic reconstruction algorithms. The main contributions of this thesis are:

- A new algorithm for computing quadratures for band-limited functions. This method is an alternative to the method in [11].
- A new algorithm for computerized tomography (CT) that increases the resolution of the resulting images compared with the usual approaches. The improvement in image quality is based on using a rational signal model and accurate polar quadratures in the Fourier domain.
- A generalization of the reduction algorithm in [12] to multivariate functions. Reduction refers to a problem where a function represented via a linear combination of decaying exponentials (with an possibly excessive number of terms) is approximated, for a given accuracy, by a representation of the same form but with a minimal number of terms.

The results obtained in this thesis use nonlinear approximations with either band-limited or decaying exponentials. Such an approach is more efficient in approximating functions with singularities than the traditional approximation techniques.

1.1 Weighted quadratures for band-limited functions

We develop a new method for constructing quadratures for band-limited exponentials, $\left\{e^{ibx}\right\}_{|b|\leq c}$, integrated against a real-valued weight function w on the interval $|x| \leq 1$ [60]. The functions from this class are not necessarily periodic in the interval $x \in [-1, 1]$. The quadratures produced by the new algorithm are comparable to the quadratures introduced in [11] and, if the weight function $w = 1$, to those in [69].

The quadratures for band-limited exponentials differ from polynomial based quadratures in several respects. Instead of integrating polynomials in a finite dimensional subspace exactly, these quadratures, for a given accuracy $\epsilon > 0$, bandlimit $c > 0$, and weight function w, integrate functions in the infinite dimensional space

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

so that

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

We note that if we construct quadrature nodes $\{x_m\}_{m=1}^{M}$ and corresponding weights to integrate functions with bandlimit $2c$ with an accuracy of ϵ^2, then 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 [11].

The approach in [69] yields quadratures to integrate band-limited functions,

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

with the weight function $w(x) = 1$. Since the space $\mathcal{E}_c$ is dense in $\mathcal{B}_c$ and vice versa, the quadratures in [11] and [60] for $w = 1$, and the quadratures in [69], may be used interchangeably. We will discuss differences between algorithms in [69, 11, 60] in Chapter 3.

Since their introduction in [69] and [11], quadratures for band-limited exponentials have found applications in solving partial differential equations (see e.g. [15, 22, 62]). In particular, they allow us to discretize operators using their spectral representation while avoiding the spurious eigenvalues appearing in other discretizations [62]. It is a significant improvement since, otherwise, these spurious eigenvalues increase the norm of the matrices (representing differential operators) by an order of magnitude. Another application

of quadratures for band-limited exponentials (with a weight) yields a fast discrete Fourier transform in polar and spherical coordinates in the Fourier domain [8] (see also [5] for integration on the sphere). We use these quadratures to construct a new Fourier domain algorithm to invert the Radon transform in Chapter 4.

In contrast with quadratures constructed to integrate polynomials in a finite dimensional subspace, quadratures for band-limited exponentials allow trade-off between bandlimit and accuracy. Given a fixed number of quadrature nodes, we may integrate exponentials with a higher bandlimit at a lower accuracy or, similarly, integrate exponentials with a smaller bandlimit at a higher accuracy. This trade-off is useful in problems of signal processing where the collected data often has a low accuracy. Additionally, quadratures for band-limited functions integrate with significantly less node clustering near the endpoints of $x \in [-1, 1]$ when compared with the classical Legendre or Chebyshev quadrature nodes.

Along with the new method of calculating quadrature nodes presented in [60], we also introduce a new method of computing quadrature weights via ℓ^∞ minimization. Our method solves an overdetermined Vandermonde system instead of the square Vandermonde systems solved in [11]. This alternative weight calculation decreases the quadrature error over the interval of integration $[-1, 1]$ and, also, gives us a near equi-oscillatory behavior of the error. We note that the ℓ^∞ minimization method for calculating quadrature weights can be applied when using quadrature nodes obtained in [11] or [69].

Since we minimize the ℓ^∞ norm of a complex-valued Vandermonde systems, we choose to use a second order cone program (SOCP) solver to find quadrature weights. A review of the SOCP problem and an associated algorithm may be found in [60, Appendix], included in this thesis as Appendix A.8.2. For the calculations in [60], we solved SOCP's using the software package CVX for MATLAB™ [36]. To check the results of this package, we implemented the primal-dual algorithm [54].

1.2 Rational approximations in tomographic reconstruction problems

We present a new, fast, Fourier algorithm of inverting X-ray tomography data. This method increases the resolution in the resulting images by extrapolating the available data in the Fourier domain using approximations via decaying exponentials.

Computerized tomography (CT) uses X-rays passing through an object to generate a projection of an

object's density (recorded by a set of detectors). The experimental apparatus is then rotated and the experiment is repeated. The experiment is repeated until the apparatus has been rotated a full 180 degrees. We note that many scanning geometries are possible in these experiments, e.g. parallel or fan beam tomography. There are also 3D modalities of such experiments, for example, Cone Beam tomography. In this thesis we focus on the simplest case, 2D CT with parallel beam geometry.

The mathematical problem of recovering a function from its projections was introduced and solved by Radon in 1917 [59]. However, his results remained unknown outside the mathematical community, and the algorithm was rediscovered by Cormack [24], who used it to produce an image from CT projection data. The Radon transform was also rediscovered by Bracewell [17], who used it for radio astronomy. A good introduction to the topic of CT, along with more historical details, may be found in [25].

These techniques have been used extensively in other fields as well. The tau-p transform (also called slant-stack) used in geophysical signal processing is the Radon transform (see, for example, [70, Chapter 6]). Another form of the Radon transform is the Hough transform in image processing (see, for example, [27]). Also, in the field of electron microscopy, both Fourier inversion techniques [26] and direct summation [32] methods for inverting the Radon transform have found use.

Imaging algorithms for CT may be split into three categories:

- Methods based on implementing analytic formulas.
- Methods casting the imaging problem as that of solving a linear system with a large matrix.
- Methods that attempt to impose sparsity conditions on the image, i.e., compressive sensing/optimization techniques.

The set of methods that implement analytic formulas includes the traditional filtered back-projection (FBP) algorithm and some methods using the Fourier transform (see, for example [55] and [56]). The FBP algorithm has complexity $\mathcal{O}\left(N^3\right)$, where N is the number of samples per projection (assuming that the number of projections is the same order). Typically, the FBP algorithm is used in practice. The Fourier algorithms are faster, with a complexity of $\mathcal{O}\left(N^2 \log N\right)$, but involve interpolation in the Fourier domain. In the standard

realizations of methods based on implementing analytic formulas, only a limited accuracy is achieved without any explicit error control.

Algebraic reconstruction techniques (ART) cast the problem in a discrete setting. Discretizing an object (usually on an equally spaced grid), the projections are interpreted as obtained by applying a matrix to the discretized object. Thus, the problem is cast as that of solving a large linear system. For more information on ART see [43].

The third category of inversion algorithms for CT data are the methods that additional sparsity constraints, i.e., compressive sensing. These methods use an ART formulation and convex optimization methods. For an example of such an approach see [21].

As noted in the review article [57], many algorithms developed for CT over the years failed to produce significant improvements over the basic FBP algorithm. One reason behind this lack of improvement is that the objects of interest are piece-wise continuous and projections typically have discontinuous first derivatives. Hence, within the standard Nyquist signal model, we are always under-sampled.

Our approach to this problem is to introduce a different signal model in the classical problem of x-ray tomography. The signal model we use is based on near-optimal rational approximations of the form

$$g(x) = a_0 + 2\mathrm{Re}\sum_{m=1}^{M} \frac{w_m}{e^{-2\pi i x + \eta_m} - 1}, \quad x \in [0,1), \tag{1.1}$$

where $w_m \in \mathbb{C}$, $\eta_m \in \mathbb{C}$, $\mathrm{Re}(\eta_m) > 0$, and $a_0 \in \mathbb{R}$ is a constant [13]. The frequencies in (1.1) are not controlled by the number of terms M, but rather the size of $\mathrm{Re}(\eta_m)$. The closer $\mathrm{Re}(\eta_m)$ is to zero (the closer the pole is to the unit circle), the higher is the frequency generated by the corresponding term in (1.1). Compare this property with the standard signal model for periodic signals,

$$f(x) = \sum_{l=0}^{2N} f\left(\frac{l}{2N+1}\right) D_N\left(x - \frac{l}{2N+1}\right), \quad x \in [0,1),$$

where

$$D_N(x) = \frac{1}{2N+1}\sum_{|l|\le N} e^{2\pi i l x} = \frac{1}{2N+1}\frac{\sin(2N+1)\pi x}{\sin \pi x}$$

is the Dirichlet kernel. This model requires an increase in the number of terms to represent signals with high frequencies.

Using (1.1), we can construct rational approximations for each tomographic projection with a small (near optimal) number of terms to represent projections with accuracy below a user-selected threshold. This approximation allows us to augment the measured data, i.e., double the number of available samples in each projection or, equivalently, extend (double) the domain of the Fourier transform.

In [61] we develop a new, polar coordinate, Fourier domain algorithm which uses our nonlinear approximations of projection data (via decaying exponentials) in a natural way. We call this new algorithm polar quadrature inversion (PQI). The polar grids we use are nodes of quadratures for band-limited functions developed in [8]. To approximate the Fourier domain data at the desired quadrature nodes we approximate (for each projection) the Fourier transform in the radial direction by decaying exponentials. We use either the algorithm in [12] or the HSVD algorithm in [49] (also called the matrix pencil method [45, 46, 47]). Not only does this allow us to evaluate the Fourier transform of projections on an unequally spaced grid, but also to extrapolate in the Fourier domain. We note that increasing the bandlimit by a factor of two is equivalent to doubling the sampling rate in the rational approximations of projections.

We note that the PQI algorithm has features associated with all three types of CT inversion algorithms listed above. For example, PQI uses discretizations of analytic formulas. However, unlike FBP (or other Fourier methods), we control the error due to the careful selection of quadratures and the way we use rational approximations (1.1). This implies that we can reproduce results of an ART-type algorithm within a PQI-type algorithm. Also, by using a rational signal model, we introduce an element of compressed sensing (i.e., the rational signal model is not subject to the Nyquist constraint).

Using our new algorithm on projections of the Shepp-Logan phantom [64] we compare the results with the standard FBP algorithm. We demonstrate that our results have increased resolution without the introduction of additional artifacts into the image.

1.3 Multi-dimensional reduction

Let us first describe the reduction problem. In one variable, given a user-supplied accuracy $\epsilon > 0$ and a suboptimal linear combination of exponentials that represents a function f, i.e.

$$f(x) = \sum_{m=1}^{M_0} b_m e^{-\tau_m x},$$

where the number of terms M_0 is excessive, the reduction problem is to construct a new linear combination of exponentials,

$$g(x) = \sum_{m=1}^{M} w_m e^{-t_m x}, \;\; \mathcal{R}e(t_m) > 0, \tag{1.2}$$

such that

$$|f(x) - g(x)| \leq \epsilon$$

and $M < M_0$ on the interval $[0, 1]$ (see [12]). Depending on the function and/or our goals, we may measure the approximation error differently, e.g. using relative error.

In this thesis, we study a multivariate generalization of the reduction problem. Namely, given a function

$$F(x_1, \ldots, x_d) = \sum_{m=1}^{M_0} b_m e^{-\tau_{1,m} x_1} \ldots e^{-\tau_{d,m} x_d}, \tag{1.3}$$

where the number of terms M_0 is excessive, and $x_1, \ldots, x_d \in [0, 1]^d$, find a new representation,

$$G(x_1, \ldots, x_d) = \sum_{m=1}^{M} w_m e^{-t_{1,m} x_1} \ldots e^{-t_{d,m} x_d}, \tag{1.4}$$

such that $M < M_0$ and

$$|G(x_1, \ldots, x_d) - F(x_1, \ldots, x_d)| \leq \epsilon, \quad \text{for } x_1, \ldots, x_d \in [0, 1]^d$$

where ϵ is user supplied accuracy. Depending on the function F, we may be able to use relative error in place of absolute error.

The one variable reduction algorithm produces a new linear combination of exponentials (1.2) that has a (nearly) minimal number of complex nodes e^{-t_m} and complex weights w_m. Such approximations are useful for constructing fast algorithms as well as signal processing [10]. The one variable reduction algorithm

is based on the theory of Adamjan, Arov, and Krein (AAK theory) [2, 3, 4]. We present a brief overview of a particular version of AAK theory and its application to rational functions in Chapter 5.

One of the goals of this thesis is to extend the reduction algorithm of [12] to linear combinations of exponentials with many variables, in the form of (1.3). AAK theory does not have a multivariate extension since it strongly relies on properties of functions of one complex variable. On the other hand, the reduction problem may be extended by using the formulation above. The exponential form in (1.3) allows us to mimic the derivation of the single variable reduction algorithm. Specifically, the tensors in the multivariate derivation reduce to matrices, and we end up using the same steps (essentially) as the reduction algorithm for functions of one variable.

Let us remark that direct generalization of the approximation algorithm in 1D to multivariate functions does not make practical sense. To illustrate this point, let us consider again consider (1.3) on a domain normalized to $[0,1]^d$, where the coefficients b_m are complex valued, and $Re(\tau_{lm}) > 0$. It is clear generating $F(\frac{k_1}{2N}, \ldots, \frac{k_d}{2N})$, $k_1, \ldots, k_d \in \{0, \ldots, 2N\}$ is prohibitive as it requires evaluation at $(2N+1)^d$ points. On the other hand, we show that the reduction problem does not suffer from the "curse of dimensionality". We describe the multivariate reduction problem and our approach to solving it in Chapter 5.

One of the most promising applications of the reduction problem is to generate quadratures in dimension $d \geq 2$, especially in non-trivial geometries. We present an example of solving the reduction problem in 2D. These techniques may also be used in data analysis, but there are additional challenges with this application, such as problems with root-finding and noise. This research is ongoing.

1.4 Additional contributions

We stated the main contributions of the thesis in the beginning of this introduction. Additionally, some of the numerical techniques developed within these algorithms are of interest on their own, specifically,

- An extension of the method of rational function approximation in [13] to discrete Fourier transform data. While such an extension is essential to our new tomographic inversion algorithm, it is of interest by itself.

- The concept of using of different norms, i.e. ℓ^1 and ℓ^∞, to find weights of rational function approximations and band-limited quadratures.

- The generation of suboptimal linear combinations of exponentials via a Richardson extrapolation-like process. These new linear combinations have properties favorable for our other algorithms, such as moving nodes closer to the center of the unit disk.

1.5 Outline of the thesis

In Chapter 2 we first review Prony's method. Then we review, in greater detail, the method of approximation by exponentials of [12] and the matrix pencil method [45, 46, 47] (also called the HSVD [49]). The discussion then moves to the method of approximating functions via proper rational functions in [13]. We derive an extension of the methods in [13] to the discrete case where the input data are DFT coefficients. We then discuss methods of calculating weights of rational approximations using different norms. To conclude the chapter, we introduce a new method for generating sub-optimal linear combinations of exponentials based upon Richardson's extrapolation.

In Chapter 3 we review the methods of calculating quadratures for band-limited exponentials in [11] and [69]. We then review the our results in the paper [60]. These results include a new method for calculating band-limited quadratures for exponentials and new methods for calculating quadrature weights.

In Chapter 4 we review the classical CT problem, the FBP algorithm, Fourier algorithms (i.e., gridding) and ART. We then discuss the results in [61]. These include using rational functions to approximate tomographic projections and a new algorithm for the inversion of tomography data, the PQI algorithm.

We begin Chapter 5 with a review of the reduction algorithm in [12] for one variable on the finite interval, and then its extension to the interval $[0, \infty)$. Then, we briefly review of AAK before deriving an extension of the reduction algorithm to reduce functions of more than one variable.

In Chapter 6 we conclude the thesis, commenting on the new results of this body of work and future work.

Appendix A contains the paper [60], which we review in Chapter 3. Appendix B contains the paper

[61], which we review in Chapter 4.

Chapter 2

Approximation and reduction with exponentials and rational functions

In this thesis we extensively use linear combinations of exponentials to approximate functions. For example, from [12], given a real-valued, smooth function $f(x)$ on the interval $x \in [0,1]$ and a constant $\epsilon > 0$ we approximate $f(x)$,

$$\left| f(x) - \sum_{m=1}^{M} w_m e^{-\eta_m x} \right| \leq \epsilon \quad \forall x \in [0,1], \tag{2.1}$$

with a (near) optimal number of terms. We formulate this problem as a discrete problem: given a sequence of uniform samples of a function $f(x)$ on the interval $[0,1]$ and a constant $\epsilon' > 0$ we approximate the sequence such that

$$\left| f\left(\frac{n}{2N}\right) - \sum_{m=1}^{M} w_m \gamma_m^n \right| \leq \epsilon' \quad n = 0, \ldots, 2N,$$

where ϵ is slightly worse than ϵ'. For this formulation to produce the approximation (2.1) we must oversample $f(x)$.

Approximating sequences via a linear combination of exponentials, for $\epsilon = 0$, was originally studied by Gaspard de Prony circa 1795. His method, known as Prony's method, has numerous numerical problems. In place of Prony's method we use newer methods of approximation via linear combinations of exponentials: either the approximation and reduction algorithms of [12] or the matrix pencil method [45, 46, 47] (also called the HSVD [49]).

We use these approximations in two capacities: first, to approximate functions in the Fourier domain with the goal of obtaining rational function approximations of real-valued functions in the space domain (this approach works for functions either on the real line or the unit circle). Second, we construct quadratures for

band-limited functions via a formulation based on the matrix pencil method. We will return to quadratures for band-limited functions later in the thesis.

To build an approximation of a function via linear combinations of rational functions we use the method of [13]. Linear combinations of complex-valued, decaying exponentials, $\{e^{-\eta_m}\}_{m=1}^{M}$ where $\mathcal{R}e\,(\eta_m) > 0$, in the Fourier domain are converted to rational functions in the space domain analytically. Neither the matrix pencil method nor the method of [12] guarantee that the approximation of a sequence will contain only decaying exponentials, and we must deal with this problem in a manner that depends on the application. We note that rational approximations constructed in this manner are closely related to theory of Adamjan, Arov, and Krein (AAK), which we will look at later in the thesis.

Rational functions have the ability to approximate functions with a large bandlimit, e.g. functions with discontinuities in their derivatives, with a small number of terms relative to Fourier methods. The reason for this property is that the decay of the exponentials, $\{e^{-\eta_m}\}_{m=1}^{M}$, in the Fourier domain is related to the distance from the poles of our rational function to the real line (or the unit circle for the periodic case, see [13]). Thus, we can approximate functions with a large bandlimit via linear combinations of rational functions possessing only a few terms, the poles of which are close to the real line (respectively, the unit circle for the periodic case). Compare this property with Fourier methods, which require either large support in the Fourier domain or a large number of Fourier series coefficients to represent such functions. In Chapter 4 we use rationals to approximate tomographic projections, which have sharp boundaries.

We begin this chapter with some preliminaries, and then review the methods of approximation via linear combinations of exponentials that we use in this thesis: the approximation method in [12] and the matrix pencil method. Then, we divert to rational approximations, reviewing the method in [13]. Concluding the chapter, we present new results: first, we show that if a function is well approximated by rational functions, then the poles found via a discrete Fourier transform (DFT) of uniform samples are sufficient to approximate the function. Second, we discuss the use of different norms, mainly ℓ^1 and ℓ^2, for finding weights of approximations via exponentials and rational functions. To demonstrate the generalization of the approximation method in [13] to sampled functions, we provide a numerical example. This example also demonstrates the ℓ^1 method for calculating weights of rational functions. Finally, we present a new method of

approximating a linear combination of exponentials, where some of the exponentials have magnitude greater than one, with a (longer) sequence of only decaying exponentials.

2.1 Preliminaries

Given a function $f(x) \in L^1(\mathbb{R})$, we use the following definition of the Fourier transform,

$$\hat{f}(\xi) = \int_{-\infty}^{\infty} f(x)\, e^{-2\pi i x \xi} dx,$$

and the inverse Fourier transform,

$$f(x) = \int_{-\infty}^{\infty} \hat{f}(\xi)\, e^{2\pi i \xi x} d\xi.$$

Similarly, given a function $f(x) \in L^1([0, 2\pi))$ we define the Fourier series as

$$f(x) = \sum_{n=-\infty}^{\infty} a_n e^{2\pi i n x},$$

where the Fourier series coefficients are defined as

$$a_k = \int_0^1 f(x)\, e^{-2\pi i k x} dx.$$

We use the following definition for the DFT,

$$\widehat{f_k} = \frac{1}{N} \sum_{n=0}^{N-1} f_n e^{-2\pi i k n/N},$$

and the associated inverse discrete Fourier transform,

$$f_n = \sum_{n=0}^{N-1} \widehat{f_k} e^{2\pi i k n/N}.$$

2.2 Prony's method

Prony's method generates linear combination of exponentials for representing sequences. For this section we follow the discussion of Prony's method in [12]. Given $2N+1$ samples $h_k = h(k/2N),\ \forall k,\ 0 \leq k \leq 2N$, we represent the samples exactly with the linear combination of exponentials,

$$h_k = \sum_{n=1}^{\widetilde{N}} w_n \gamma_n^k. \tag{2.2}$$

The algorithm to determine coefficients w_m and nodes γ_m is as follows, given our sequence $h_k,\ k = 0, \ldots, 2N$ we construct the $(N+1) \times (N+1)$ singular Hankel matrix $\mathbf{H}$,

$$\mathbf{H} = \begin{pmatrix} h_0 & h_1 & \ldots & h_N \\ h_1 & \ddots & & \vdots \\ \vdots & & & \\ h_N & \ldots & & h_{2N} \end{pmatrix},$$

and find a vector in the null space such that

$$\mathbf{Hq} = 0,$$

where, without loss of generality, we solve for $\mathbf{q}$ in the form $\mathbf{q} = \Big(q_0, \ldots\ q_{\widetilde{N}-1}, -1, 0, \ldots, 0\Big)$ and $\widetilde{N} \leq N$. We note that if $\mathbf{H}$ were non-singular we would form a new, extended sequence $\{h_k\}_{k=0}^{2N+2}$ that includes two extra terms, a free parameter h_{2N+1}, and a parameter h_{2N+2} chosen such that $\mathbf{H}$ formed from the extended sequence is singular [12]. Using $\mathbf{q}$ we solve for the nodes $\{\gamma_n\}_{n=1}^{\widetilde{N}}$ via recurrence relation,

$$h_{k+\widetilde{N}} = \sum_{n=0}^{\widetilde{N}-1} h_{k+n} q_n, \quad k \geq 0$$

since, from our definition of $\mathbf{q}$,

$$(\mathbf{Hq})_k = \sum_{n=0}^{\widetilde{N}-1} h_{k+n} q_n - h_{k+\widetilde{N}} = 0.$$

The solution to this recurrence relation is

$$h_k = \sum_{n=1}^{\widetilde{N}} w_n \gamma_n^k \ \ \forall\, k = 0, \ldots, 2N, \tag{2.3}$$

where $\{\gamma_n\}_{n=1}^{\widetilde{N}}$ are roots of the polynomial $u\,(x) = \sum_{k=0}^{\widetilde{N}} q_k z^k$ (where, recall, $q_{\widetilde{N}} = -1$).

If some of $\{\gamma_n\}_{n=1}^{\widetilde{N}}$ are repeated roots, then the corresponding weights w_n are replaced by a polynomial $p_n(k)$ whose order is strictly less than the multiplicity of the root. However, for the purposes of our discussion we will assume that the all of the roots are unique. To find the weights $\{w_n\}_{n=1}^{\widetilde{N}}$ we use the roots $\{\gamma_n\}_{n=1}^{\widetilde{N}}$ to form the Vandermonde matrix $\mathbf{V}_{nk} = \gamma_n^k$, where $n = 1, \ldots, \widetilde{N}$ and $k = 0, \ldots, \widetilde{N} - 1$, and solve the Vandermonde system (2.3) for the weights w_n.

We would like to point out two difficulties with using Prony's method in practice. First, due to a large numerical null space it is difficult to find the vector $\mathbf{q}$. Second, the matrix $\mathbf{V}$ may be very poorly

conditioned. The method of approximation via exponentials in [12] remedies these problem by relaxing the condition of exact equality from (2.2) to,

$$\left| h_k - \sum_{n=1}^{\widetilde{N}} w_n \gamma_n^k \right| < \epsilon,$$

where $\epsilon > 0$ is an arbitrarily small, user supplied accuracy. This formulation allows representations with fewer nodes and weights since we only calculate an approximation of the entries in the sequence.

2.3 Approximation of functions by a sum of exponentials

We review the one-dimensional algorithm for approximation via linear combinations of exponentials in [12]. Given a user-selected accuracy $\epsilon > 0$ and $2L+1$ equally spaced samples of a complex-valued function $h(c\xi)$, $\xi \in [0,1]$, the algorithm yields an approximation

$$\left| h(c\frac{l}{2L}) - \sum_{m=1}^{M} w_m e^{-c\eta_m l} \right| < \epsilon, \;\; \mathcal{R}\mathrm{e}\,(\eta_m) > 0, \tag{2.4}$$

for $0 \le l \le 2L$, where the number of terms, M, is near minimal for the choices of c, L, and ϵ. In this formulation, the constant $c > 0$ scales the problem to the interval $[0,1]$. Provided that $h(c\xi)$ is sufficiently sampled, we define $\tau_m = 2L\eta_m$, and replace $l/(2L)$ with a continuous variable ξ in (2.4), and obtain

$$\left| h(c\xi) - \sum_{m=1}^{M} w_m e^{-c\tau_m \xi} \right| < \epsilon', \tag{2.5}$$

for $\xi \in [0,1]$. The new error ϵ' is only slightly worse than ϵ.

Algorithm 2.1 [14, Appendix A.1]

The algorithm to produce the approximation (2.4) has the following steps:

- Construct the $(L+1) \times (L+1)$ Hankel matrix $\mathbf{H}_{ll'} = h_{l+l'}$, where $h_{l+l'} = h(c\frac{l+l'}{2L})$, $0 \le l, l' \le L$.
- Find a vector $\mathbf{u}$ satisfying $\mathbf{H}\mathbf{u} = \sigma\overline{\mathbf{u}}$ with positive σ close to ϵ. A solution is guaranteed by Tagaki's factorization [44] and may be reduced to finding the singular value decomposition (SVD) of $\mathbf{H}$. Given singular values $\sigma_0 \ge \sigma_1 \ge \cdots \ge \sigma_M \ge \cdots \ge \sigma_L$, we choose M such that $\epsilon \approx \sigma_M/\sigma_0$ and the corresponding singular vector $\mathbf{u} = \{u_l\}_{l=0}^{L}$.

- Compute the roots γ_m of the polynomial

$$u(z) = \sum_{l=0}^{L} u_l z^l. \tag{2.6}$$

- The exponents η_m in equation (2.4) are defined by the roots γ_m via $\eta_m = -\log(\gamma_m)/c$, where log is the principal value of the logarithm. We also compute $\tau_m = 2L\eta_m$.

- Compute the weights w_m in (2.4) by solving the Vandermonde system,

$$h\left(\frac{cl}{2L}\right) = \sum_{m=1}^{M} w_m \gamma_m^l \tag{2.7}$$

  for the best fit in the ℓ^2 sense.

We only use the nodes that have corresponding weights with magnitude greater than the error bound ϵ. Typically, only M nodes satisfy this criterion. Sometimes we have a priori information that allows us to only solve for weights corresponding to a subset of the roots of (2.6). For example, some approximations should use only roots inside the unit circle. Such is the case when approximating Fourier series coefficients or samples of the Fourier transform.

An important feature of Algorithm 2.1 is that it detects the level of noise in the data. Using this feature, we can choose an approximation free from most of the noise that contaminates a signal. When the ratio σ_m/σ_0 reaches the level of noise in the signal, the rate of decay of the singular values of the matrix $\mathbf{H}$ changes significantly. The reason behind this behavior is that noise cannot be represented efficiently by exponentials. Thus, the benefit of adding additional terms in (2.4) becomes very small. We use this change in the rate of decay to select the singular value σ_M, and therefore the number of terms M, in (2.4). For more details see [13].

2.3.1 Numerical example

To demonstrate Algorithm 2.1 we approximate the function $\operatorname{sinc}(50\pi x)$ on the interval $[0, 1]$. We choose our L to be 128 and select our desired accuracy as $\epsilon = 10^{-8}$. The resulting approximation error is displayed in Figure 2.1.

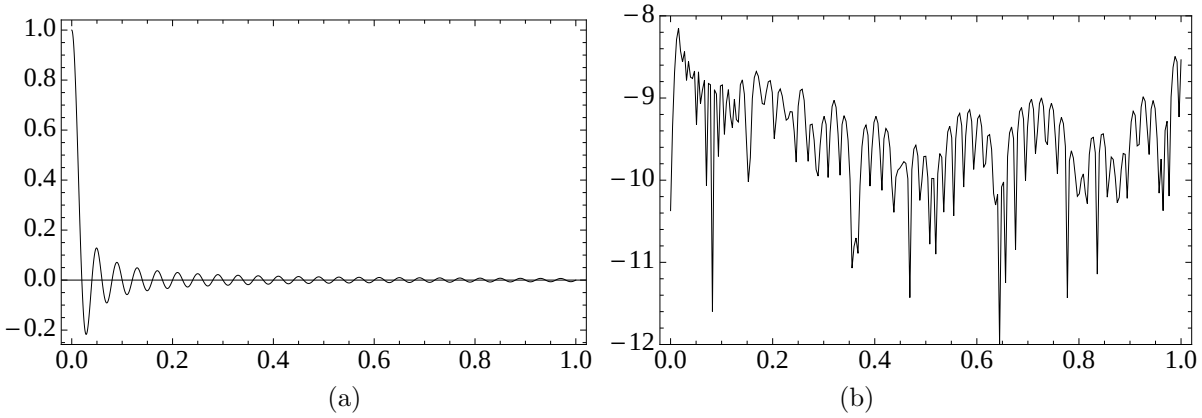


Figure 2.1: We display $\mathrm{sinc}\,(50\pi x)$ on $[0,1]$ (a) and the corresponding error of our approximation, in $\log_{10}$ scale, (b) using $\epsilon = 10^{-8}$ and 19 nodes and weights. We observe that the error is slightly worse towards the endpoints of the interval, but is still beneath our ϵ.

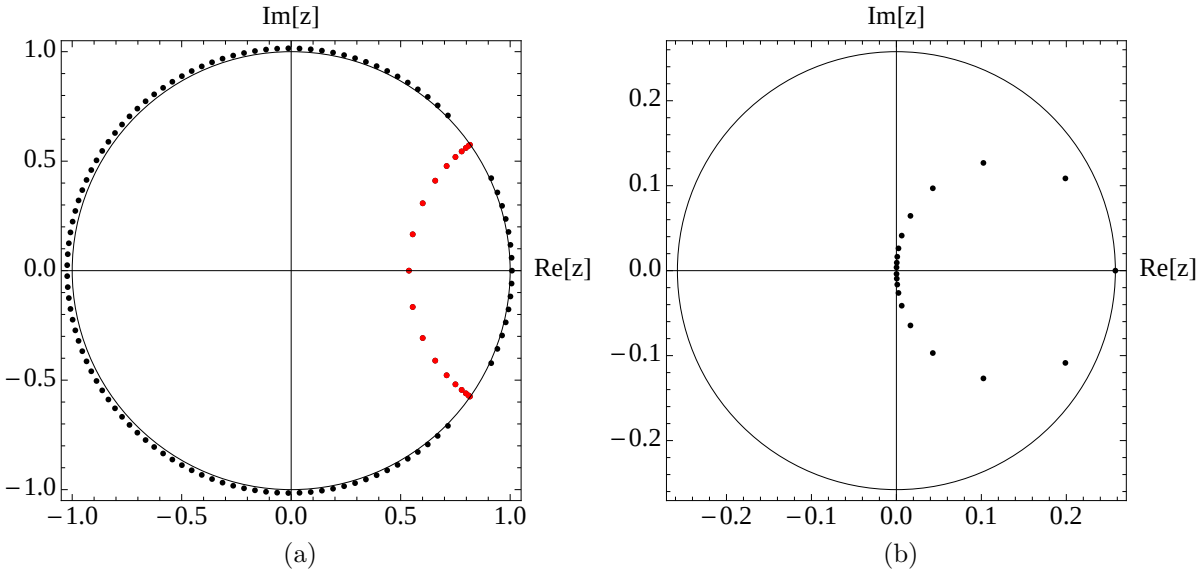


Figure 2.2: We display the nodes (a) and weights (b) used for approximating $\mathrm{sinc}\,(50\pi x)$ on $[0,1]$ (a) with the error bound $\epsilon = 10^{-8}$ and 19 nodes and weights. The the nodes marked in red (a) are all inside the unit circle, and these are the nodes we use for our approximation. The black points in (a) are roots of the con-eigenpolynomial outside the unit circle. The weights corresponding to these nodes are smaller than ϵ. We do not use them in our approximation.

2.4 Matrix Pencil Method

We point out alternatives to Algorithm 2.1 for finding (near) optimal approximations via linear combinations of exponentials. One such alternative is the matrix pencil method [45, 46, 47], also called the HSVD [49]. The advantage of this method is that we avoid the step of finding roots of the polynomial (2.6). However, we note that Algorithm 2.1 is faster (if properly implemented) since we need to find only a single singular vector of a structured (Hankel) matrix. The least squares matrix pencil method, reviewed in Section 2.4.2, requires computing of all singular vectors of $\mathbf{H}$ up to the index $M-1$ corresponding to $\sigma_{M-1}/\sigma_0 \approx \epsilon$.

We review two versions of the matrix pencil method. First, we review the standard matrix pencil method, meant to extract exponentials from a signal without noise. The second version is the least squares matrix pencil method. This method, instead of directly solving an eigenvalue problem as does the standard matrix pencil method, first takes the SVD and works with the resulting singular values and singular vectors. This method is well suited for extracting exponentials in the presence of noise. Our new method of calculating quadratures for band-limited exponentials, discussed in Chapter 3, is based on an argument using the least squares version of the matrix pencil method.

2.4.1 Standard matrix pencil method

The standard matrix pencil method is an alternative to Prony's method, i.e. given N samples $\{h_k\}_{k=0}^{N-1}$ we represent the samples exactly with the linear combination of exponentials,

$$h_k = \sum_{m=1}^{M} w_m \gamma_m^k, \tag{2.8}$$

where w_m, $\gamma_m \in \mathbb{C}$. We form two matrices from the data:

$$\mathbf{A} = \begin{bmatrix} h_0 & h_1 & \dots & h_{L-1} \\ h_1 & h_2 & \dots & h_L \\ \vdots & \vdots & & \vdots \\ h_{N-L-1} & h_{N-L} & \dots & h_{N-2} \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} h_1 & h_2 & \dots & h_L \\ h_2 & h_3 & \dots & h_{L+1} \\ \vdots & \vdots & & \vdots \\ h_{N-L} & h_{N-L+1} & \dots & h_{N-1} \end{bmatrix},$$

where $\mathbf{A}$, $\mathbf{B} \in \mathbb{C}^{(N-L)\times L}$. The parameter L is the so-called "pencil parameter." Its should be chosen such that $M \leq L \leq N - M$. The reason for this choice is explained later in this section. These two matrices may be represented as

$$\mathbf{A} = \mathbf{Z_1 W Z_2}, \quad \mathbf{B} = \mathbf{Z_1 W Z_0 Z_2},$$

where,

$$\mathbf{Z_1} = \begin{bmatrix} 1 & 1 & \dots & 1 \\ \gamma_1 & \gamma_2 & \dots & \gamma_M \\ \vdots & \vdots & & \vdots \\ \gamma_1^{(N-L-1)} & \gamma_2^{(N-L-1)} & \dots & \gamma_M^{(N-L-1)} \end{bmatrix}_{(N-L)\times M}, \quad \mathbf{Z_2} = \begin{bmatrix} 1 & \gamma_1 & \dots & \gamma_1^{(L-1)} \\ 1 & \gamma_2 & \dots & \gamma_2^{(L-1)} \\ \vdots & \vdots & & \vdots \\ 1 & \gamma_M & \dots & \gamma_M^{(L-1)} \end{bmatrix}_{M\times L},$$

$$\mathbf{W} = diag(w_1, \dots, w_M), \quad \mathbf{Z_0} = diag(\gamma_1 \dots, \gamma_M).$$

We now consider the matrix pencil

$$\mathbf{B} - \lambda \mathbf{A} = \mathbf{Z_1 W} \left\{ \mathbf{Z_0} - \lambda \mathbf{I} \right\} \mathbf{Z_2}.$$

When $\lambda \neq \gamma_m$ and $M \leq L \leq N - M$, the rank of this matrix pencil is M (since $\mathbf{Z_1}$ and $\mathbf{Z_2}$ are full rank when the nodes $\{\gamma_m\}_{m=1}^{M}$, $m = 1, \dots, M$, are distinct). However, if $\lambda = \gamma_m$ (and $M \leq L \leq N - M$) then the matrix $\mathbf{Z_0} - \lambda \mathbf{I}$ has a zero row, and thus the rank is $M - 1$.

Given that the only values of λ that reduce the rank are the nodes $\{\gamma_m\}_{m=1}^{M}$, $k = 1, \dots, M$, we find the values of λ such that the rank of

$$\mathbf{B} - \lambda \mathbf{A}$$

reduces from M to $M - 1$. This is the so-called "singular generalized eigenvalue problem" (singular since $\mathbf{A}$ and $\mathbf{B}$ have dimensions $(N - L) \times L$). To obtain these values of λ we solving the eigenvalue problem

$$\det \left(\mathbf{A}^{\dagger} \mathbf{B} - \lambda \mathbf{I} \right) = 0,$$

where $\mathbf{A}^{\dagger}$ is the pseudo-inverse of $\mathbf{A}$. To see why solving this eigenvalue problem gives us the nodes γ_k, we apply the pseudo-inverse $\mathbf{A}^{\dagger}$ to $\mathbf{B}$, yielding

$$\mathbf{A}^{\dagger} \mathbf{B} = \mathbf{Z_2}^{\dagger} diag \left(z_1, \dots, z_M \right) \mathbf{Z_2}.$$

Hence, the eigenvalues of $\mathbf{A}^{\dagger}\mathbf{B}$ are the M nodes γ_m, $m = 1, \ldots, M$.

The rank reduction property is the reason behind our bounds on the pencil parameter L. If $M > L$ then the rank of $\mathbf{Z_2}$ is L, and the rank of $\{\mathbf{Z_0} - \lambda\mathbf{I}\}\mathbf{Z_2}$ is also L regardless of the value of λ. Thus, we do not get a reduction in the rank of $\mathbf{B} - \lambda\mathbf{A}$ when $\lambda = \gamma_m$, and the method fails. Similarly, if $L > N - M$ then the matrix $\mathbf{Z_1}$ has a rank of $N - L$, and thus the rank of $\mathbf{Z_1}\mathbf{W}\{\mathbf{Z_0} - \lambda I\}$ is $N - L$ regardless of the values of λ and the method fails.

2.4.2 Least squares matrix pencil method

The least squares version of the matrix pencil method does not approximate the input sequence exactly. Instead, the data are approximated with a user-supplied accuracy ϵ. This variant of the matrix pencil method was created to approximate noisy signals of the form $g_k = h_k + n(k)$, $k = 0, \ldots, N-1$, where $\{h_k\}_{k=0}^{N-1}$ is a linear combination of exponentials (2.8) and $n(k)$ is additive noise. The formulation of the problem solved by the least squares matrix pencil method is the following, given a sequence $\{g_k\}_{k=0}^{N-1}$ and a user-supplied accuracy $\epsilon > 0$, approximate the sequence such that

$$\left| g_k - \sum_{m=1}^{M} w_m \gamma_m^k \right| < \epsilon, \quad k = 0, \ldots, N-1,$$

where M is a near-optimal number of terms.

The first step in the least squares matrix pencil method is to take the SVD of the data matrix $\mathbf{G}$:

$$\mathbf{G} = \begin{bmatrix} g_0 & g_1 & \cdots & g_L \\ g_1 & g_2 & \cdots & g_{L+1} \\ \vdots & \vdots & & \vdots \\ g_{N-L-1} & g_{N-L} & \cdots & g_{N-1} \end{bmatrix}_{(N-L)\times(L+1)},$$

$$\mathbf{G} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{*},$$

and select the singular value with index M such that $\sigma_{M-1}\kappa \approx \epsilon$, where ϵ is our desired error and κ is a constant dependent on the parameter L that we revisit later. We denote the truncated SVD as

$$\mathbf{G}_M = \mathbf{U}_M\mathbf{\Sigma}_M\mathbf{V}_M^{*}.$$

From the truncated SVD we construct the matrices $\widetilde{\mathbf{U}}_M$ and $\widehat{\mathbf{U}}_M$, where $\widetilde{\mathbf{U}}_M$ is the matrix formed from taking the first M columns of $\mathbf{U}$ and then removing the last row. Similarly, $\widehat{\mathbf{U}}_M$ is the formed by taking the first M columns of $\mathbf{U}$ and then removing the first row. Using these matrices we solve the the eigenvalue problem

$$\det\left(\widetilde{\mathbf{U}}_M^{\dagger}\widehat{\mathbf{U}}_M - \lambda I\right) = 0$$

for the nodes $\{\gamma_m\}_{m=1}^{M}$. For the case with no noise, the least squares matrix pencil method reduces to the standard matrix pencil method, since

$$\mathbf{A} = \widetilde{\mathbf{U}}\mathbf{\Sigma}\mathbf{V}^*,$$

$$\mathbf{B} = \widehat{\mathbf{U}}\mathbf{\Sigma}\mathbf{V}^*,$$

$$\mathbf{A}^{\dagger}\mathbf{B} = \left(\mathbf{\Sigma}\mathbf{V}^*\right)^{\dagger}\mathbf{U}^{\dagger}\mathbf{U}\left(\mathbf{\Sigma}\mathbf{V}^*\right),$$

The matrix $\mathbf{A}^{\dagger}\mathbf{B}$ is similar to $\mathbf{U}^{\dagger}\mathbf{U}$ and, thus, to find the nodes $\{\gamma_m\}_{m=1}^{M}$ we find the eigenvalues of $\mathbf{U}^{\dagger}\mathbf{U}$.

Let us discuss the constant κ associated with the approximation error. This constant is dependent on the parameter L, i.e. it changes depending on how rectangular $\mathbf{G}$ is. However, when we input an odd number of samples $h_k,\ k = 0, \ldots, 2N'$, and choose L such that $\mathbf{G} \in \mathbb{C}^{\left(N'+1\right)\times\left(N'+1\right)}$, the constant of proportionality is $\kappa = 1/\sigma_0$. When we use the least square matrix pencil method as the basis of our new method of calculating quadratures for band-limited exponentials, discussed in Chapter 3, or as an alternative to Algorithm 2.1, we use only square Hankel matrices.

2.5 Approximation via rational functions

In this thesis we use the rational approximations of functions (or data) presented in [13]. The basic idea behind this algorithm is that we approximate Fourier data, either a Fourier transform or a Fourier series, with decaying exponentials (see Algorithm 2.1). Since our approximations in the Fourier domain are linear combinations of decaying exponentials, we analytically transform back to the space domain, obtaining a rational function approximation of our function. For the purposes of this thesis we restrict ourselves to the approximation of Fourier series coefficients and, therefore, periodic rational functions.

Let us consider real valued, periodic rational functions of the form

$$g(x) = a_0 + 2\mathrm{Re}\sum_{m=1}^{M} \frac{w_m}{e^{-2\pi i x + \eta_m} - 1}, \quad x \in [0,1). \tag{2.9}$$

The coefficients $\{a_k\}_{k\in\mathbb{Z}}$ of the Fourier series of g in (2.9) are readily available,

$$a_k = \int_0^1 g(x)\, e^{-2\pi i k x} dx = \sum_{m=1}^{M} w_m e^{-\eta_m k}, \;\; a_{-k} = \overline{a}_k \;\; k \in \mathbb{N}, \tag{2.10}$$

and

$$a_0 = \int_0^1 g(x)\, dx,$$

which coincides with the constant term in (2.9), i.e., the proper rational terms in (2.9) do not contribute to a_0. We emphasize that all Fourier coefficients are fully described by $4M+1$ real parameters; given the complex-valued parameters η_m and w_m, $m = 1, \ldots M$, and a_0, we recover the function g completely. We note that the number of these parameters does not limit the frequency contents of the function g. Instead, the frequency is controlled by the distance of the nodes, $e^{-\eta_m}$, $m = 1, \ldots, M$, from the unit circle, i.e., the real part of the exponent, $\mathrm{Re}(\eta_m)$.

Given $2L+1$ Fourier series coefficients with positive index of a real-valued function $f(x)$, $\{a_k\}_{k=1}^{2L+1}$, the constant term a_0, and a user-supplied accuracy $\epsilon > 0$ we use the algorithm in Section (2.3) to approximate the sequence $\{a_k\}_{k=1}^{2L+1}$, obtaining

$$\left| a_k - \sum_{m=1}^{M} w_m e^{-\eta_m k} \right| < \epsilon$$

for $k = 1, \ldots, 2L+1$. Using the geometric series we then construct the periodic rational function (2.9).

2.5.1 Rational approximation of discrete data via the discrete Fourier transform

The authors of [13] demonstrate how to approximate periodic, real-valued functions with rational functions generated from the positive-index Fourier series coefficients of a function. However, in many applications (e.g. tomographic reconstruction problems) we do not have access to either the Fourier series coefficients (if we assume that the projection data are periodic) or the Fourier transform. In these problems we typically work with data samples in the space domain. We will show how to extend the ideas of [13] to these data via the DFT.

DFT coefficients are an approximation of the Fourier series coefficients of a function, but they not very accurate for functions sampled below the Nyquist rate. Assuming that we are under-sampled, we must use additional information to extract the Fourier series coefficients from DFT coefficients. In the following construction we assume that we may accurately approximate the function $g(x)$ on $[0,1)$ with a rational function of the form (2.9).

To find an expression for the DFT coefficients of N samples of g, $g_n = g(n/N)$, $n = 0, \ldots, N-1$ (where $N \geq 4M+1$), we substitute the representation (2.10) into the Fourier series of g. For any integer $N > 0$, summing the geometric series, we have

$$\begin{aligned} g(x) - a_0 &= \sum_{k \geq 1} \left(a_k e^{2\pi i k x} + \overline{a}_k e^{-2\pi i k x} \right) \\ &= 2\mathcal{R}e \sum_{n \geq 0} \sum_{j=0}^{N-1} \sum_{m=1}^{M} w_m e^{-\eta_m (j+Nn)} e^{2\pi i (j+Nn) x} - 2\mathcal{R}e \sum_{m=1}^{M} w_m \\ &= 2\mathcal{R}e \sum_{j=0}^{N-1} \sum_{m=1}^{M} \frac{w_m}{1 - e^{-\eta_m N} e^{2\pi i N x}} e^{-\eta_m j} e^{2\pi i j x} - 2\mathcal{R}e \sum_{m=1}^{M} w_m. \end{aligned} \tag{2.11}$$

Sampling $g(x)$, we obtain

$$g\left(\frac{n}{N}\right) = a_0 - 2\mathcal{R}e \sum_{m=1}^{M} w_m + 2\mathcal{R}e \sum_{j=0}^{N-1} \sum_{m=1}^{M} \frac{w_m}{1 - e^{-\eta_m N}} e^{-\eta_m j} e^{2\pi i j n / N}, \tag{2.12}$$

where $n = 0, \ldots, N-1$. Computing the DFT of $g\left(\frac{n}{N}\right)$, $n = 0, \ldots, N-1$,

$$\hat{g}_j = \frac{1}{N} \sum_{n=0}^{N-1} g\left(\frac{n}{N}\right) e^{-2\pi i n j / N}, \quad j = 0, \ldots, N-1,$$

we obtain

$$\hat{g}_j = \sum_{m=1}^{M} \frac{w_m}{1 - e^{-\eta_m N}} e^{-\eta_m j} + \sum_{m=1}^{M} \frac{\overline{w}_m}{1 - e^{-\overline{\eta}_m N}} e^{-\overline{\eta}_m (N-j)}, \quad j = 1, \ldots, N-1, \tag{2.13}$$

and

$$\widehat{g}_0 = a_0 - 2\mathcal{R}e \sum_{m=1}^{M} w_m + 2\mathcal{R}e \sum_{m=1}^{M} \frac{w_m}{1 - e^{-\eta_m N}} = a_0 + 2\mathcal{R}e \sum_{m=1}^{M} w_m \left(\frac{1}{1 - e^{-\eta_m N}} - 1 \right).$$

Note that the DFT coefficients and the Fourier coefficients (2.10) of g share the same nodes $e^{-\eta_m}$, $m = 1, \ldots, M$. On the other hand, the DFT coefficients also contain (what we call) companion nodes $e^{\overline{\eta}_m}$, $m = 1, \ldots, M$ which lie outside the unit disk. These companion nodes appear due to aliasing, i.e., folding of Fourier series coefficients corresponding to high frequencies onto low frequencies. We note that if the sampling

rate is above the Nyquist rate, then the DFT coefficients $\hat{g}_j$, $j = 1, \ldots, N/2 - 1$ accurately approximate the Fourier series coefficients with the same indices and the companion nodes may be ignored. However, when we are approximating functions with singularities, we have to deal with the potential influence of the companion nodes. We may use either Algorithm 2.1 or the matrix pencil method (described in Section 2.4.2) to find nodes given DFT coefficients as input. To find weights for the linear combination approximating DFT data we may either use the last step of Algorithm 2.1, i.e. solve a Vandermonde system on the Fourier domain side, or use one of the alternative methods described in the next section of this chapter.

Let us make an observation about approximating sampled data with periodic rational functions. Assume we are given N equally spaced samples of g, $g_n = g\,(n/N)$, $n = 0, \ldots, N-1$ so that $N \geq 4M + 1$. In principle, this information should be sufficient to find a_0, η_m and w_m. In the presence of noise, however, the function g needs to be over-sampled, i.e., $N = \nu\,(4M + 1)$, where $\nu > 1$ is the oversampling factor. We note that this sampling requirement is different than the traditional oversampling required for the Shannon-Nyquist signal model. Typically, the amount of sampling required for the rational signal model, both with and without noise, is less than the rate dictated by the Nyquist criterion.

To compute these nonlinear approximations we take the DFT of our input data g_n, $n = 0, \ldots, N-1$, and apply Algorithm 2.1 to find the nodes. However, Algorithm 2.1 requires an odd number of input samples in order to form the Hankel matrix $\mathbf{H}$. Therefore, we have two approaches to using the DFT coefficients $\hat{g}_j$, $j = 0, \ldots, N-1$. In the first approach, we use the DFT coefficients $\hat{g}_j$, $j = 1, \ldots, N-1$ as input into Algorithm 2.1. This forces the total number of samples N to be even. The resulting $N/2 \times N/2$ Hankel matrix $\mathbf{H}$ we construct in Algorithm 2.1 is equivalent to a self-adjoint Toeplitz matrix (for this reason we call this approach Hankel-Toeplitz). This may be seen by multiplying $\mathbf{H}$ by the matrix $\mathbf{J}$ with entries $\left\{\delta_{i,N/2-j-1}\right\}_{i,j=0,\ldots,N/2-1}$, leading to a construction similar to that in [11]. In this case the roots (except those on the unit circle) indeed come in pairs, $\gamma_m^{out} = 1/\overline{\gamma_m^{in}}$. However, some of the nodes may lie on the unit circle which creates problems with the extension.

In the second approach, we only use half of the DFT coefficients. Specifically, we use $\hat{g}_j$, $j = 1, \ldots, 2\widetilde{N} + 1$, where $\widetilde{N}$ is defined as $\widetilde{N} = \lfloor N/4 + 1/2 \rfloor - 1$, where $\lfloor\,\rfloor$ denotes the floor function. The

$\left(\widetilde{N}+1\right) \times \left(\widetilde{N}+1\right)$ Hankel matrix $\mathbf{H}$ used in Algorithm 2.1 is then defined as

$$\mathbf{H}_{ll'} = \widehat{g}_{l+l'+1}, \;\; l, l' = 0, \ldots, \widetilde{N}.$$

In this construction we do not expect the nodes to come in pairs as we do in the Hankel-Toeplitz approach. In fact, if g is sampled at a rate above the Nyquist rate this approach should only output nodes that are strictly inside the unit circle. However, we use this approach on problems that have discontinuities in their derivatives. Hence, the decay in the Fourier domain may not be sufficient to ensure that the DFT coefficients represent the Fourier series coefficients well. This may lead some of the nodes being output by Algorithm 2.1 to be outside of the unit circle. The existence of these nodes is incompatible with our rational model (2.9), and we are forced to discard them, creating problems with our rational approximation. To alleviate some of these problems we explore methods of calculating weights alternative to the last step of Algorithm 2.1.

2.5.2 Calculating weights for rational approximations

Since Algorithm 2.1 produces a near optimal number of terms, in the case where we only use half the DFT coefficients to form the Hankel matrix $\mathbf{H}$, it may sometimes introduce a growing exponential with a small weight. Similarly, when using all the DFT coefficients to form the Hankel matrix $\mathbf{H}$, unless we are sampled at a rate beyond the Nyquist rate, there may be exponentials with significant weights that lie on the unit circle. Roots either on the unit circle or just outside the unit circle (both typically have small weights) prevent us from constructing the rational function (2.9). We discard these spurious nodes at the expense of introducing an additional error. The following approach aims to isolate such errors to the vicinity of singularities (responsible for a slow decay of the Fourier coefficients and causing this effect).

In our experience, when using all of the DFT coefficients to form $\mathbf{H}$, the locations of nodes on the unit circle do not necessarily correspond to singularities. This will cause problems with the method we propose. Thus, we restrict our discussion on calculating weights with ℓ^1 minimization to the case where we only use half of the DFT coefficients.

Let us discuss why nodes outside the unit circle may appear in our approximations. As mentioned in Section 2.5.1, the exponential representation of the DFT coefficients (2.13) differs from that of the Fourier

coefficients (2.10) in that it has a second sum,

$$\sum_{m=1}^{M} \frac{\overline{w}_m}{1-e^{-\overline{\eta}_m N}} e^{-\overline{\eta}_m (N-j)}, \quad \mathcal{R}\mathrm{e}\,(\eta_m) > 0. \tag{2.14}$$

Since, in the construction of the Hankel matrix $\mathbf{H}$, we use only the DFT coefficients with indices corresponding to positive frequencies, we have $j \leq N/2-1$ in (2.14). Consequently, the sum in (2.14) typically has a small contribution within this index range, although, as a function of j, it contains growing terms, $e^{\overline{\eta}_m j}$. Since the form (2.4) does not account for these terms explicitly, their contribution may result in roots γ_m just outside the unit disk. Also, such an effect may be caused by the presence of noise.

Since the proximity of a node to the unit circle controls the frequency contribution of that node, the impact of removing nodes just outside the unit disk should be localized to neighborhoods of singularities. However, when we remove nodes from just outside the unit circle and use the ℓ^2 norm to calculate the weights via (2.7), the error spreads out to a large neighborhood of the singularity as illustrated in Figure 2.5 (a). To remedy this situation, we calculate weights by minimizing the ℓ^1 norm of the the residual with respect to the original spatial data. We choose to minimize the ℓ^1 norm due to its well known sparsity properties (see, e.g., [63]). The effect of using the ℓ^1 norm to calculate weights is illustrated in the numerical example in Section 2.5.3.

Let us describe the details of calculating weights using the ℓ^1 minimization method. Once we obtain the exponents $\{\eta_m\}_{m=1}^{M}$ via Algorithm 2.1 and remove those with $\mathcal{R}e\,(\eta_m) > 0$, we proceed to compute weights in the space domain. Specifically, using the fact that the function is real we extend the approximation of positive frequencies to negative frequencies and analytically sum the Fourier series to obtain a rational function of the form (2.9). Denoting the grid at which we measure the projection data as $\{x_j\}_{j=0}^{N-1}$, we discretize (2.9) and obtain a Cauchy-like system to solve for the weights w_m.

$$2\mathrm{Re} \sum_{m=1}^{M} w_m \left(\frac{1}{e^{-2\pi i n/N+\eta_m}-1} - \frac{1}{1-e^{-\eta_m N}} + 1 \right) = g\left(\frac{n}{N}\right) - \widehat{g}_0 = g_n - \widehat{g}_0, \tag{2.15}$$

where $n = 0, \ldots, N-1$. Let us denote the $N \times 2M$ matrix of this system as $\mathbf{C}$ and the right hand side, $\{g_n - \widehat{g}_0\}_{n=0}^{N-1}$, as $\mathbf{g}$. We note that the contribution of terms $1/\left(1-e^{-\eta_m N}\right) - 1$ is minor since $e^{-\eta_m N}$ is typically small.

We first solve

$$\mathbf{Cw} = \mathbf{g}, \quad \text{with} \quad \operatorname{argmin}_{\mathbf{w}} \|\mathbf{Cw} - \mathbf{g}\|_2 , \tag{2.16}$$

where $\mathbf{g}$ is the vector containing the (shifted) data samples. We then verify if the residual is within the error tolerance. If it is not (usually, this is the case for only a few projections), we then solve

$$\mathbf{Cw} = \mathbf{g}, \quad \text{with} \quad \operatorname{argmin}_{\mathbf{w}} \|\mathbf{Cw} - \mathbf{g}\|_1 . \tag{2.17}$$

In this approach we rely on an observation that if solving (2.16) satisfies the error tolerance, then the difference in using ℓ^1 or ℓ^2 norms is insignificant.

For solving (2.16) we use the SVD while for (2.17) we use convex optimization. For the latter, since $\mathbf{C}$ is complex valued, we solve via a second order cone program implemented in CVX [36]. An alternative approach for solving (2.17) relies on the iteratively re-weighted least squares (IRLS) algorithm (see, e.g. [63]).

We illustrate the difference between using the ℓ^2 norm and the ℓ^1 norm, when we discard some nodes outside the unit circle, in an example. In Figure 2.5 (a), using the ℓ^2 norm, we show that the error spreads away from the singularities of the function in Figure 2.3. This should be compared with Figure 2.5 (b) where, using (2.17), the error is significantly reduced away from the sharp transitions of the function shown in Figure 2.3.

2.5.3 Numerical example

We test our extension of the approximation by periodic rational functions in [13] to use on sampled data. The function we chose to use was also an example in [13],

$$f(x) = \begin{cases} \left(2e^{4\pi x} - 1 - e^{\pi}\right), & 0 \le x < 1/4 \\ -\sin\left(\frac{4}{3}\pi x - \frac{\pi}{3}\right), & \frac{1}{4} \le x < 1 \end{cases}, \tag{2.18}$$

shown in Figure 2.3. We performed two experiments on this function. First, we sampled the function $f(x)$ with $N = 512$, and, for $\epsilon = 10^{-3}$, used the method of rational approximation for sampled data in Section 2.5.1 to approximate. The resulting error of this approximation is shown in Figure 2.4 (a). Second, we sampled the

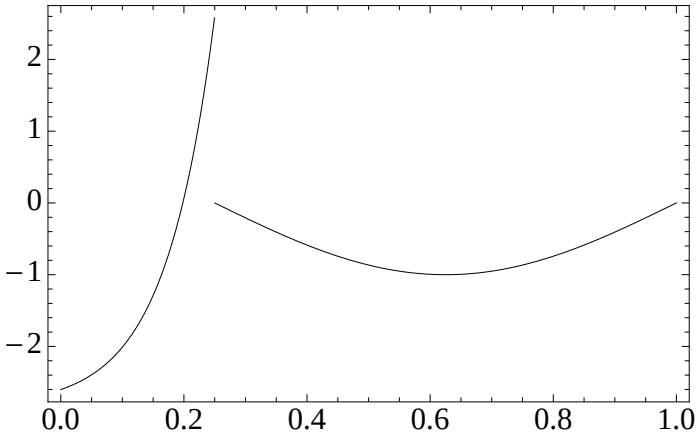


Figure 2.3: A plot of $f(x)$, (2.18). Note that this function has two discontinuities, one at 0 and the other at $1/4$.

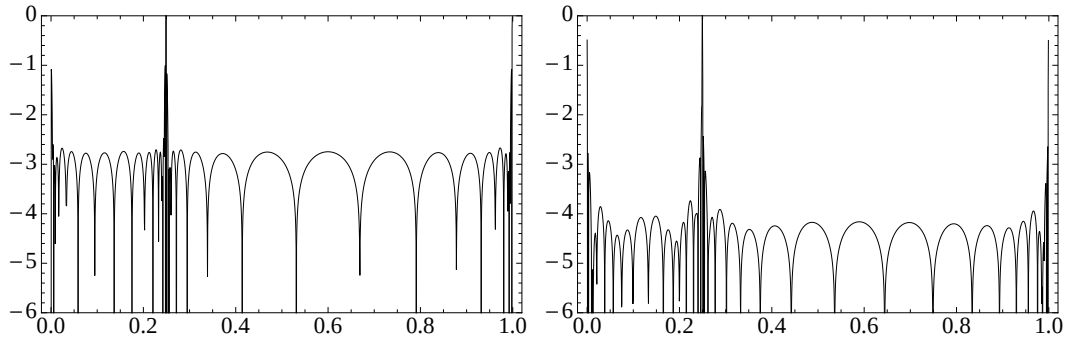


Figure 2.4: Approximation errors for the function $f(x)$ (2.18), shown in Figure 2.3. Shown in (a) is the approximation error for $f(x)$, sampled at a rate of $N = 512$, for $\sigma_M/\sigma_0 = 10^{-3}$, where $M = 10$. The number of rational functions matched the index, with 10 nodes inside the unit circle. In (b) we have the approximation error for $f(x)$, sampled with $N = 1024$, for $\sigma_M/\sigma_0 = 10^{-4.5}$, where $M = 15$. For this experiment the number of nodes inside the unit circle did not match the index. There were only 13 nodes inside the unit circle. All weights were calculated using the ℓ^1 minimization method.

function $f(x)$ with $N = 1024$, and, for $\epsilon = 10^{-4.5}$, used the method of rational approximation for sampled data in Section 2.5.1 to approximate. The resulting error of this approximation is shown in Figure 2.4 (b).

We observe that the number of nodes inside the unit circle matched the index M for the case of 512 samples. However, for the case of 1024 samples, where we asked for a smaller error, there was a mismatch. The number of nodes inside the unit circle was two less than the index. The reason for this is that the two missing nodes were located just outside the unit circle. Since we drop these nodes, we use the ℓ^1 minimization method to calculate weights. The result is that we get an error close to the value of ϵ even though we are missing two nodes with significant weight. We compare the error of using the ℓ^1 minimization method for calculating weights with the ℓ^2 method, both in the case of missing nodes, in Figure 2.5.

2.6 Generating sub-optimal approximations

There is another method for dealing with approximations, generated by the method in [12] or the matrix pencil method, requiring exponentials with magnitude greater than one. This method is limited in its applicability by the distance between the nodes with magnitude greater than one and the unit circle. If the spurious nodes outside the unit circle lie a very small distance from the unit circle, the method will work well. If the nodes are too far from the unit circle, this method will fail. We explain this mode of failure after

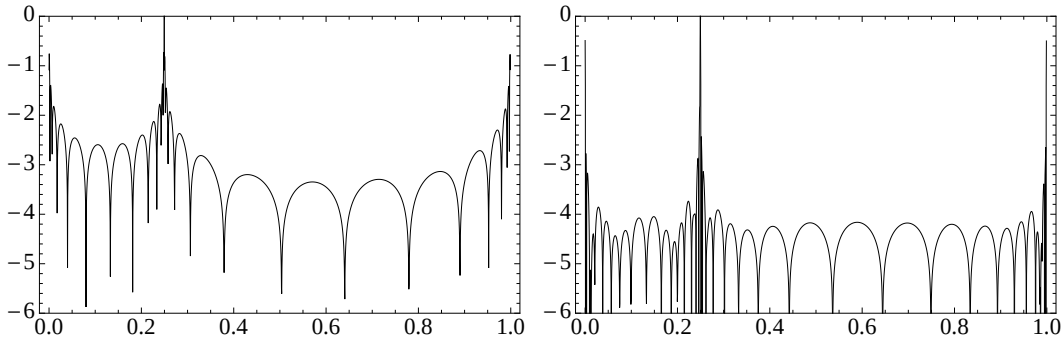


Figure 2.5: A comparison of the approximation errors using the ℓ^2 minimization method for calculating weights (a), and the ℓ^1 method (b) for our example $f(x)$ sampled at a rate of $N = 1024$. The user-supplied error in these examples is $\sigma_M/\sigma_0 = 10^{-4.5}$, where $M = 15$. Clearly, (a) illustrates the damage done to the error by the ℓ^2 method for calculating weights in the case of the missing nodes. The ℓ^1 minimization method does a much better job of compensating for missing nodes in (b).

we describe the method.

The exponential approximations generated by the matrix pencil method or the approximation method in [12] occasionally require exponentials with magnitude greater than one due to the property of generating approximations with a near-optimal number of terms. Specifically, given a user-specified accuracy $\epsilon > 0$ both algorithms approximate the data with the prescribed accuracy via linear combinations of exponentials that are as short as possible. There is no restriction on the location of these exponentials with respect to the unit circle. The only restriction is on the length of the approximation.

Instead of removing nodes outside the unit circle and then compensating with the weight calculation (see Section 2.5.2), it makes to sense to replace the near-optimal linear combination of exponentials with a suboptimal linear combination (i.e. with more terms) whose terms possess the traits we desire (e.g. inside the unit circle). To generate this suboptimal linear combination of exponentials we use an idea similar to Richardson extrapolation. The idea is to generate a new, suboptimal, approximation by multiplying the spurious exponentials outside the unit circle by a linear combination of real-valued decaying exponentials. We choose the weights and exponents of this linear combination of exponentials to control the error via Taylor series error. To demonstrate this method, we consider a linear combination of exponentials,

$$f(x) = \sum_{m=1}^{M} w_m e^{-\eta_m x}, \tag{2.19}$$

where $x \in [0,1]$ and $\mathcal{R}e(\eta_m) > 0$, except one of the η_m's where $\mathcal{R}e(\eta_m) \leq 0$. We denote the index of this node as m_0, where $\eta_{m_0} = -\delta + i\beta$, $\delta > 0$ and $\mathrm{Im}(\beta) = 0$. It is without a loss of generality that we assume only one of the exponentials is outside the unit circle, since the case where more than one node lies outside the unit circle can be dealt with by either applying the method to each spurious node individually, or applying the correction for the node farthest outside the unit circle to all the nodes outside the unit circle. We separate the function (2.19) into two parts

$$f(x) = \sum_{m=1}^{M} w_m e^{-\eta_m x} + w_{m_0} e^{-\eta_{m_0} x} = \widetilde{f}(x) + w_{m_0} e^{-\eta_{m_0} x}, \tag{2.20}$$

and then multiply to obtain our new approximation, $g(x) = \widetilde{f}(x) + w_{m_0} e^{-\eta_{m_0} x} \left(2e^{-2\delta x} - e^{-4\delta x}\right)$. The approximation $g(x)$ had one more exponential than $f(x)$, and the error of the approximation is described

using the Taylor series of the exponential,

$$\begin{aligned} 2e^{-2\delta x} - e^{-4\delta x} &= 2 \cdot \left(1 - 2\delta x + \frac{4\delta^2 x^2}{2!} + \dots\right) - \left(1 - 4\delta x + \frac{16\epsilon\delta^2 x^2}{2!} + \dots\right) \\ &= 1 + \mathcal{O}\left(\delta^2 x^2\right). \end{aligned} \tag{2.21}$$

Thus, the error $|g(x) - f(x)|$ is controlled by the distance we move the nodes outside the unit circle. We may choose to use higher-order approximations but at the price of adding more terms to our new linear combinations of exponentials.

Let us make a few observations about this method of generating suboptimal approximations. First, note that we are limited in the amount we can move exponentials. If, to move all the nodes inside the unit circle, we use $\delta > 1$, then increasing the number of terms will hurt our error. Therefore, in order for this method to work well, the nodes we move inside must be very close to the unit circle. We also note that we are free to move the nodes by amounts different than δ, depending on how much we want to move the exponentials in (2.19).

When we use this technique to generate a suboptimal linear combination of exponentials, the final step in the process is to apply the reduction algorithm in [12], but with a slight modification. The modification is to use the algorithm for functions defined on the interval $[0, \infty)$. This version of the reduction algorithm is reviewed in Chapter 5. Using this modification of the reduction algorithm guarantees the reduced linear combination of exponentials to only have terms with exponentials inside the unit circle. The reduction algorithm, however, must be used carefully. The conditioning of the involved matrices depends upon the distance between the nodes being reduced. Since this method of moving exponentials into the unit circle generates nodes very close to one another, we must exercise caution.

Chapter 3

Band-limited functions and quadratures

One of the contributions of this thesis is a new algorithm for computing quadratures for band-limited 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 $[-1, 1]$. The currently available algorithms for generating this type of quadratures may be found in [11] and [69]. The new algorithm for computing quadratures of band-limited exponentials yields quadrature nodes as eigenvalues of a matrix. We also introduce a new method for calculating quadrature weights using ℓ^∞ minimization. Both new results of this chapter were published in [60]

To provide background on this topic, we first review band-limited functions and their restriction to the finite interval $[-1, 1]$. We also review a basis for band-limited functions, the prolate spheroidal wave functions. Our focus then shifts to exploring the methods of [11] and [69]. Both papers develop algorithms (via different approaches) to calculate quadratures for band-limited functions. Finally, we review the results of our paper, [60], included with this thesis as Appendix A.

3.1 Prolate spheroidal wave functions

In practice, signals are not only of finite duration, but are also band-limited. In other words, these signals have a beginning and an end, while transmitted with a finite frequency content. However, it is well known that a function cannot be have finite support in both the time domain and the frequency domain. How can this be reconciled?

To explore this phenomena, we consider as an example Gaussians and their Fourier transform. Defining

the Fourier transform as

$$\hat{f}(\omega) = \int_{-\infty}^{\infty} f(x)\, e^{-2\pi i \omega x} dx,$$

we see that the Fourier transform of the Gaussian $f(x) = e^{-\alpha x^2}$ $(\alpha > 0)$ is the Gaussian $\hat{f}(\omega) = \sqrt{\pi/\alpha}\, e^{-(\pi\omega)^2/\alpha}$. Neither $f(x)$ or $\hat{f}(\omega)$ have compact support. However, both $f(x)$ or $\hat{f}(\omega)$ are rapidly decaying. We note that the rate of decay in both domains is controlled by the parameter $\alpha > 0$. We have chosen a function that isn't compactly supported in both domains, but is approximately so. Given any $\epsilon > 0$, there exists intervals in both the space domain and the frequency domain such that, outside of these intervals, $|f(x)| < \epsilon$ and $\left|\hat{f}(\omega)\right| < \epsilon$.

A set of functions that represents both band-limited and space-limited signals are the prolate spheroidal wave functions (PSWFs) introduced by Slepian, Landau, and Pollack [67, 51, 52]. These functions have their support within the interval $[-1, 1]$ in the space domain and $[-c, c]$ in the Fourier domain. Thus, they emulate the near-band-limited and space-limited behavior of Gaussians. To define these functions, we define the space-limiting operator as

$$F_c\phi(\omega) = \int_{-1}^{1} \phi(x)\, e^{-ic\omega x} dx.$$

Combining this operator with its adjoint yields the space-limiting and band-limiting operator $Q_c = \frac{c}{2\pi} F_c^* F_c$. The PSWFs are the eigenfunctions of the operator Q_c,

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

We note that they are also the eigenfunctions of F_c,

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

where

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

Slepian and Pollak [67] made the observation that ψ_j are also the eigenfunctions of the differential operator

$$L_c\psi_j \equiv \left(-(1-x^2)\frac{d^2}{dx^2} + 2x\frac{d}{dx} + c^2x^2\right)\psi_j(x) = \eta_j\psi_j(x), \tag{3.1}$$

i.e., they coincide with the classical PSWFs of mathematical physics. In (3.1), the eigenvalues η_j form a strictly $\alpha > 0$, positive sequence.

3.2 Band-limited quadratures

Unlike the classical Gaussian quadratures for polynomials which integrate exactly a subspace of polynomials up to a fixed degree, Gaussian type quadratures for exponentials, introduced in [11], 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, for a given $\epsilon > 0$, all exponentials with $|b| \leq c$ are integrated with at least that accuracy. Specifically, for a given accuracy ϵ, bandlimit c, and weight function w, the Gaussian-type quadratures in [11] are designed to integrate functions in the linear space

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

so that

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

We note that quadratures of this type are not unique for a given bandlimit c and accuracy ϵ.

An alternative approach in [69] yields quadratures to integrate band-limited functions in

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

with the weight function $w(x) = 1$. This approach is based on explicitly constructing and using the PSWFs, a basis in $\mathcal{B}_c$. The PSWFs form a Chebyshev system (defined below), 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 [11] for $w = 1$ and [69] may be used interchangeably. We note that the method in [11] allows us to construct quadratures for a weight function that does not have to be positive (see e.g. [9, Section 5]). Similarly, the method for computing quadratures for band-limited exponentials we introduce in [60] also allows us to construct quadratures for integrals against an non-sign-definite weight function.

3.3 Quadratures for band-limited functions via trigonometric moments

Let us briefly summarize the method in [11] for generating quadratures to integrate the family of exponentials $\left\{e^{ibx}\right\}_{|b|\leq c}$ with a real-valued weight function w. First, we compute the trigonometric moments

$$u_n = \int_{-1}^{1} e^{icxn/N} w\left(x\right) dx, \quad -N \leq n \leq N, \tag{3.2}$$

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

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

is over-sampled. We then arrange the trigonometric moments $\{u_n\}_{n=-N}^{N}$ as the entries of a self-adjoint Toeplitz matrix $\mathbf{T} = \{u_{n-n'}\}_{0\leq n,n'\leq N}$. An important note is 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.

Computing the eigenvector $\mathbf{q}^{(s)} = [q_0, \ldots, q_N]^t$ 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 $\{\gamma_j\}_{j=1}^{N}$, $\gamma_j \neq 0$, it is shown in [11, Theorem 4.1] that there exist weights $\{w_j\}_{j=1}^{N}$ 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

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

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, $\{\gamma_j\}_{j=1}^{N}$, 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 $\mathbf{T}\mathbf{x}_0 = \mathbf{e}_0$, where $\mathbf{e}_0 = [1, 0, \ldots 0]^t$, uses the Wiener-Levinson algorithm of complexity $\mathcal{O}\left(N^2\right)$ with a small constant which is sufficiently fast (on a desktop computer) 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, $j = 1, \ldots, N$ 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. However, no extra precision is needed when using these quadratures.

If the weight function $w = 1$, then the eigenpolynomial $q^{(s)}(z)$ is a discrete prolate spheroidal wave function (DPSWF) (see [66]) 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 [69] obtained by a different approach that uses the PSWFs.

If we construct quadrature nodes $\{x_m\}_{m=1}^{M}$ and corresponding weights to integrate functions with bandlimit $2c$ with an accuracy of ϵ^2, then 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.

3.4 Quadratures for band-limited functions via PSWFs

In [69] quadratures are constructed using the PSWFs, which form a basis for band-limited functions. The approach closely follows the classical method of obtaining Gaussian quadratures for polynomials. To understand this method of constructing quadratures, and how it relates to polynomials, we define the Chebyshev system. A family of $n + 1$ real-valued functions, $\{u_0, \ldots, u_n\}$, defined on an interval $I = [a, b]$, is a Chebyshev system if any nontrivial, linear combination of functions in the family

$$u(t) = \sum_{j=0}^{n} \alpha_j u_j(t),$$

has at most n zeros on the interval I. This property is a generalization of the set of polynomials $\left\{1, t, t^2, \ldots, t^n\right\}$.

Since the PSWFs form a Chebyshev system, the approach for computing quadratures in [69] first finds ψ_j by solving (3.1) and then computes the M nodes as zeros of ψ_M, $\psi_M(x_j) = 0$, $j = 1, \ldots M$. It is observed in [69] that the accuracy of quadratures may be improved by optimizing the positions of nodes and the values of weights further. A Newton-type optimization in 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 [38]). 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 [33]. This fact that the PSWFs satisfy the second order differential equation in (3.1) implies that their zeros may be found without ever explicitly computing the functions themselves. We note that the DPSWFs (see the previous section) also satisfy a second order differential equation and, hence, the algorithm in [33] is applicable in that case as well.

3.5 Review of "On generalized Gaussian quadratures for band-limited exponentials" [60]

In this section we summarize the paper "On generalized Gaussian quadratures for band-limited exponentials" by Reynolds, Beylkin, and Monzón [60]. We have included a copy of [60] in Appendix A.

In [60] we present a new method of constructing generalized Gaussian quadratures for exponentials integrated against a non-sign-definite weight function. The capabilities of this new method are similar to the method in [11]. We derive this method by finding quadrature nodes that accurately discretize inner products of band-limited functions (although the method works for non-positive-definite weight functions, as shown in an example in [60]). The algorithm for computing these quadratures is based on the matrix pencil method, reviewed in Chapter 2.

We also present a new method for calculating quadrature weights in [61]. This method is based on minimizing the ℓ^∞ residual of a Vandermonde system that is formed using the quadrature nodes found in our calculations. We minimize the ℓ^∞ norm of the residual of this Vandermonde system via a second order cone program (SOCP). The tests on this method of calculating weights were performed on the quadrature nodes generated via our new algorithm, and quadrature nodes provided in [69] and [11]. Resulting quadrature errors are slightly more accurate than those output by standard methods of calculating quadrature weights. In addition, the error behavior of quadratures with weights calculated in this manner is closer to equioscillatory, especially in the case of [69] (see [60, Section 6] for plots of these error behaviors).

3.5.1 Computing quadrature nodes as eigenvalues

We derive a method of calculating quadrature nodes using inner products. The details may be found in [60, Section 3]. The use of inner products in our derivation leads to an algorithm that calculates the quadrature nodes via an eigenvalue problem. We first show how this approach translates to generating standard polynomial quadratures.

Considering 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 two matrices, $\mathbf{A}$, $\mathbf{B} \in \mathbb{R}^{M\times M}$,

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

and

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

The quadrature nodes we seek are $\{x_m\}_{m=1}^{M}$ and the weights $\{w_m\}_{m=1}^{M}$. To solve for the nodes we denote the non-singular matrix $\mathbf{\Phi} = \{\phi_l(x_m)\}_{\substack{l=0,\ldots,M-1\\ m=1,\ldots,M}}$, which allows us to express $\mathbf{A}$ and $\mathbf{B}$ as $\mathbf{A} = \mathbf{\Phi W \Phi}^t$ and $\mathbf{B} = \mathbf{\Phi X W \Phi}^t$, where $\mathbf{W} = \mathrm{diag}\,(w_1 \ldots w_M)$ and $\mathbf{X} = \mathrm{diag}\,(x_1 \ldots x_M)$ are diagonal matrices and $\mathbf{M}^t$ denotes the transpose of the matrix $\mathbf{M}$. Computing

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

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 extend this idea to find quadratures for exponentials with bandlimit c. The quadratures we find are not exact, but they are accurate up to a user-specified accuracy $\epsilon > 0$, i.e.

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

To solve this problem, we consider

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

discretized as,

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

We compute quadrature nodes $\{x_m\}_{m=1}^{M}$ and weights $\{w_m\}_{m=1}^{M}$, with $M < N$, that discretize $\mathbf{G}$,

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

such that

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

We show in [60, Section 3] that, defining $\mathbf{A}$ and $\mathbf{B}$ as

$$\mathbf{A} = \{\mathbf{Q}_{nn'}\}_{\substack{n=0,\ldots,N-1 \\ n'=0,\ldots,N}}, \;\; \mathbf{B} = \{\mathbf{Q}_{nn'}\}_{\substack{n=1,\ldots,N \\ n'=0,\ldots,N}}, \tag{3.3}$$

allows us, formally, to solve for the quadrature nodes as the eigenvalues of the matrix $\mathbf{A}^{\dagger}\mathbf{B}$, where $\mathbf{M}^{\dagger}$ denotes the pseudoinverse of the matrix $\mathbf{M}$. However, due to the poor conditioning of $\mathbf{G}$ we use the Singular Value Decomposition (SVD) as an intermediate step. We truncate the SVD of $\mathbf{G}$ at the index $M-1$ such that $\sigma_{M-1}/\sigma_0 \approx \epsilon$,

$$\mathbf{G}_M = \mathbf{U}_M \mathbf{\Sigma}_M \mathbf{V}_M^*,$$

where $\Sigma_M = \mathrm{diag}\,(\sigma_0, \sigma_1, \ldots \sigma_{M-1})$, and $\mathbf{U}_M$ and $\mathbf{V}_M$ are the submatrices of the unitary matrices $\mathbf{U}$ and $\mathbf{V}$ containing the first M singular vectors of $\mathbf{G}$. From (3.3) we write the corresponding matrices $\mathbf{A}_M$ and $\mathbf{B}_M$ as

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

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

$$\widetilde{\mathbf{U}}_M = \{\mathbf{U}_{\tilde{n}m}\}_{\substack{\tilde{n}=0,\ldots,N-1 \\ m=0,\ldots,M-1}}, \;\; \widehat{\mathbf{U}}_M = \{\mathbf{U}_{\hat{n}m}\}_{\substack{\hat{n}=1,\ldots,N \\ m=0,\ldots,M-1}}. \tag{3.4}$$

The quadrature nodes are the non-zero eigenvalues of $\mathbf{A}_M^{\dagger}\,\mathbf{B}_M$. However, since $\widetilde{\mathbf{U}}_M^{\dagger}\widehat{\mathbf{U}}_M$ and $\mathbf{A}_M^{\dagger}\,\mathbf{B}_M$ have the same non-zero eigenvalues,

$$\mathbf{A}_M^{\dagger}\,\mathbf{B}_M = (\mathbf{\Sigma}_M \mathbf{V}_M^*)^{\dagger}\, \widetilde{\mathbf{U}}_M^{\dagger} \widehat{\mathbf{U}}_M \mathbf{\Sigma}_M \mathbf{V}_M^*,$$

we calculate the quadrature nodes as the eigenvalues of the $M \times M$ matrix $\widetilde{\mathbf{U}}_M^{\dagger}\widehat{\mathbf{U}}_M$.

We summarize the algorithm from [60] for calculating quadrature nodes for band-limited exponentials given a weight function $w(x)$, bandlimit c, and accuracy ϵ.

Algorithm 3.1

(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 over-sampled.

(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-1} such that σ_{M-1}/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 (3.4).

(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 quadrature nodes x_m, $m = 1, \ldots, M$.

3.5.2 Calculating quadrature weights

We shift our focus to methods of calculating quadrature weights corresponding to the quadrature nodes calculated in the last section. This material is contained in [60, Section 4]. In particular, we concentrate on two methods of calculating quadrature nodes, one using ℓ^2 minimization and the other using ℓ^∞ minimization. The system we solve using these methods is the over-determined system $\mathbf{V}\mathbf{w} = \mathbf{u}$, where $\mathbf{V} \in \mathbb{C}^{(2N+1)\times M}$ is a Vandermonde matrix defined as $\mathbf{V}_{nm} = e^{icx_m n/N}$, $m = 1 \ldots M$ and $n = -N, \ldots, N$, $\mathbf{u} = \{u_n\}_{n=-N}^{N}$ is the vector of trigonometric moments

$$u_n = u\left(\frac{n}{N}\right) = \int_{-1}^{1} e^{icx\frac{n}{N}} w(x)\,dx, \tag{3.5}$$

and $\mathbf{w} = \{w_m\}_{m=1}^{M}$ is the vector of quadrature weights.

To solve the ℓ^2 minimization problem, $\min_{\mathbf{w}} \|\mathbf{V}\mathbf{w} - \mathbf{u}\|_2$, we avoid using the original over-sampled trigonometric moments (3.5) by forming the normal equations, $\mathbf{V}^*\mathbf{V}\mathbf{w} = \mathbf{V}^*\mathbf{u}$, explicitly. This form of the normal equations allows us to solve for weights by inverting a $M \times M$ matrix instead of calculating the SVD of $\mathbf{V}$.

To solve the ℓ^∞ minimization problem, $\min_{\mathbf{w}} \|\mathbf{V}\mathbf{w} - \mathbf{u}\|_\infty$, we cannot avoid using the original over-sampled trigonometric moments (3.5). Furthermore, since the matrix $\mathbf{V}$ has complex entries, we resort to convex optimization to solve the ℓ^∞ minimization problem. Our approach is to write $\min_{\mathbf{w}} \|\mathbf{V}\mathbf{w} - \mathbf{u}\|_\infty$ as a SOCP (see the appendix of [60]), and apply an appropriate solver.

For the examples in [60] we used CVX, a convex optimization package for MATLAB™, to calculate weights. We checked the results generated by CVX against our own implementation of the primal-dual algorithm for solving SOCP's (see [54]). The results from our algorithm and CVX matched. For more on SOCP's see the appendix of [60].

3.5.3 Numerical examples and comparisons with quadratures in [69] and [11]

We conclude our review of [60] with an overview of the numerical examples and quadrature comparisons in the paper. The first of two examples we examine is an example from antenna design. We compute a quadrature for the following integral,

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

where c is the bandlimit and I_0 is the modified Bessel function of order zero. For a bandlimit of $c = 10\pi$ and singular value σ_{21}, $\sigma_{21}/\sigma_0 \approx 1.2 \cdot 10^{-15}$, the resulting quadrature has 22 nodes. The ℓ^∞ minimization method produces weights that result in a quadrature with maximum absolute error $\epsilon = 1.21 \cdot 10^{-14}$. For more detail on this example see Section A.5.1.

The second example in [60] displays the ability of our algorithm to work with non-sign-definite weight functions. For the weight function

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

$x \in [-1, 1]$, we calculate quadrature nodes and weights for bandlimit $c = 5\pi$ and singular value ratio $\sigma_{13}/\sigma_0 = 5.0 \cdot 10^{-14}$. The resulting quadrature has 14 nodes and the maximum error is $6.68 \cdot 10^{-14}$. For more detail on this example see Section A.5.2.

To compare the quadratures computed in [60] with the quadratures in [69] and [11] we generate similar tables as were in the other two papers and directly compare the results. For a sampling of bandlimits from

$c = 20$ to $c = 4000$, our method generated quadratures slightly more accurate with weights calculated via ℓ^2 minimization. Weights calculated via ℓ^∞ minimization were more accurate for the bandlimits between $c = 20$ to $c = 1000$. The ℓ^∞ minimization algorithm, implemented in CVX, could not handle the matrices necessary for the calculations $c = 2000$ or $c = 4000$. For more information on these comparisons see Section A.6.

We also observed improvement in error generated by quadrature nodes from [69] and [11] when using our ℓ^∞ minimization method for calculating weights. Quadratures for bandlimit $c = 50$, with 24 nodes, improved in accuracy from $1.2 \cdot 10^{-7}$ in [11] to $7.8 \cdot 10^{-8}$ using nodes from [11] and weights calculated via ℓ^∞ minimization. Similarly, the accuracy of $8.3 \cdot 10^{-8}$ in [69] was improved to $5.3 \cdot 10^{-8}$ using nodes from [69] and weights calculated via ℓ^∞ minimization. For more detailed comparisons see Section A.6.

Chapter 4

Rational approximations for tomographic reconstructions

4.1 Introduction

Before we review the results of [61], we revisit the classical X-ray tomography problem. While we describe the problem for the simplest geometry, the so-called parallel beam geometry, our results in [61] generalize to other scanning geometries, e.g. fan beam tomography. Then we briefly review some of the standard methods for the inversion of the Radon transform, including, filtered back-projection (FBP), Fourier methods, and algebraic reconstruction tomography (ART). Finally, we review [61], which is attached in this thesis as Appendix B.

4.2 The Radon transform

We consider the classical problem of X-ray tomography, the inversion of the Radon transform. Given a function u defined on the plane, the Radon transform is defined as the integral over the lines $\{\mathbf{x} : \ s = \boldsymbol{\nu} \cdot \mathbf{x}\}$ parametrized by $s \in \mathbb{R}$ and $\boldsymbol{\nu} = (\cos\theta, \sin\theta)$ a unit vector,

$$v(s, \boldsymbol{\nu}) = (Ru)(s, \boldsymbol{\nu}) = (R_\theta u)(s) = \int_{\mathbb{R}^2} u(\mathbf{x}) \delta(s - \boldsymbol{\nu} \cdot \mathbf{x}) \, d\mathbf{x}. \tag{4.1}$$

Note that the points $(s, \boldsymbol{\nu})$ and $(-s, -\boldsymbol{\nu})$ define the same line. We consider $s \in [-1, 1]$ since the object of interest has compact support.

4.2.1 Inversion of the Radon transform

The inversion of the Radon transform may be accomplished via the FBP algorithm, which may be written in operator form as

$$I = R^* K R,$$

where the back-projection operator (the dual transform) is defined as

$$\left(R^* v\right)(\mathbf{x}) = \int_{\|\boldsymbol{\nu}\|=1} v\left(s, \boldsymbol{\nu}\right) |_{s=\boldsymbol{\nu}\cdot\mathbf{x}} \, d\boldsymbol{\nu}$$

and the convolution operator (filter) as

$$\left(Kf\right)(s) = \frac{1}{2} \int_{\mathbb{R}} |\rho| \, \hat{f}\left(\rho\right) e^{2\pi i \rho s} d\rho.$$

4.3 Fourier slice theorem and Fourier reconstruction algorithms

Fourier reconstruction algorithms build an object's two dimensional Fourier transform from the Fourier transforms of the tomographic projections. These samples, due to the geometry of the parallel-beam geometry, are initially collected on a polar grid. The data are then interpolated to a cartesian grid and inverted via the two-dimensional DFT. An alternative to this approach is to leave the interpolation for the unequally spaced fast Fourier transform (USFFT) [28, 6, 53], see e.g. [56]. However, to use the USFFT in this manner, both quadratures and interpolation methods must be chosen carefully, which has not been the case in this field. This was a part of our motivation for the new algorithm described in [61].

To gather information about the two dimensional Fourier transform from tomographic projections we need to relate the Fourier transform of the projections to the Fourier transform of the object. Towards this end, we use the Fourier slice theorem,

$$\left(Fu\right)(\rho\boldsymbol{\nu}) = \int_{\mathbb{R}^2} u(\mathbf{x}) e^{-2\pi i \rho \boldsymbol{\nu}\cdot\mathbf{x}} d\mathbf{x} = \int_{\mathbb{R}} \left(Ru\right)(s, \boldsymbol{\nu}) \, e^{-2\pi i \rho s} ds. \tag{4.2}$$

We derive the Fourier slice theorem directly from our definition of the Radon transform (4.1). Plugging (4.1)

into the right-hand side of (4.2) and evaluating the integral yields

$$\begin{aligned}\int_{\mathbb{R}} (Ru)(s,\boldsymbol{\nu})\, e^{-2\pi i\rho s} ds &= \int_{\mathbb{R}}\int_{\mathbb{R}^2} u(\mathbf{x})\delta(s-\boldsymbol{\nu}\cdot\mathbf{x})\, e^{-2\pi i\rho s} d\mathbf{x} ds \\ &= \int_{\mathbb{R}^2} u(\mathbf{x}) \left(\int_{\mathbb{R}} \delta(s-\boldsymbol{\nu}\cdot\mathbf{x})\, e^{-2\pi i\rho s} ds \right) d\mathbf{x} \\ &= \int_{\mathbb{R}^2} u(\mathbf{x}) e^{-2\pi i\rho \boldsymbol{\nu}\cdot\mathbf{x}} d\mathbf{x}.\end{aligned}$$

Using the Fourier slice theorem, we outline the standard Fourier reconstruction algorithm as used in the field. Let us denote the tomographic projection data as $\{g_n(k)\}_{n=0}^{N-1}$, where $k = 0, \ldots, N_\theta - 1$ are indices of the angles used in the tomographic experiment (where $\theta_k = \pi k/N_\theta$), and N is the number of samples in each projection (where $x_n,\ n = 0, \ldots N-1$, such that $x_n \in [-1, 1]$).

Algorithm 4.0

(1) Compute the FFT of the projection data $\{g_n(k)\}_{n=0}^{N-1}$ for each $k = 0, \ldots, N_\theta - 1$.

(2) Using only the DFT coefficients up to the index of the Nyquist frequency, apply a window such that the DFT coefficients decay to a value close to zero.

(3) The data from Step 2 naturally lie on a polar grid. Interpolate these data to a cartesian grid.

(4) Apply the two dimensional DFT to recover the image.

Steps (3) and (4) of Algorithm 4.0 may be replaced by one step if we use the USFFT in place of two dimensional DFT's and interpolation. Both of Algorithm 4.0 and the alternative using the USFFT have a complexity of $\mathcal{O}\left(N^2 \log N\right)$, where we assume that $N \approx N_\theta$.

While this algorithm produces a low-accuracy result, it cannot be improved further without connecting this computation with the evaluation of integrals appearing in the formulation of the Radon transform. In our new algorithm, this connection is established by quadratures for the disk in the Fourier domain.

4.4 Filtered back projection

4.4.1 Filtered back projection algorithm

The FBP algorithm is still the most common algorithm used in industrial scanners [57]. The reason for its wide use is its simplicity and robustness in the presence of noise. The complexity of the FBP algorithm is $\mathcal{O}\left(N^3\right)$ (assuming that $N_\theta \approx N$), which is fast enough for most industrial CT machines.

We derive the FBP algorithm from the Fourier slice theorem and the Fourier reconstruction algorithm. Given the definition of the inverse Fourier transform

$$u\left(\mathbf{x}\right) = \int_0^{2\pi} \int_0^{\infty} \widehat{u}(\mathbf{x}) e^{2\pi i \rho \boldsymbol{\nu}\cdot\mathbf{x}} \left|\rho\right| d\rho d\theta,$$

we rewrite the the integral over ρ such that $\rho \in (-\infty, \infty)$,

$$u\left(\mathbf{x}\right) = \frac{1}{2} \int_{|\boldsymbol{\nu}|=1} \int_{-\infty}^{\infty} \widehat{u}(\mathbf{x}) e^{2\pi i \rho \boldsymbol{\nu}\cdot\mathbf{x}} \left|\rho\right| d\rho d\boldsymbol{\nu}. \tag{4.3}$$

Note that in this definition we integrate twice over the Fourier domain, hence the division by $1/2$. We replace $\widehat{u}\left(\mathbf{x}\right)$ in (4.3) using the Fourier slice theorem,

$$u\left(\mathbf{x}\right) = \frac{1}{2} \int_{|\boldsymbol{\nu}|=1} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \left(Ru\right)\left(s, \boldsymbol{\nu}\right) e^{2\pi i \rho (\boldsymbol{\nu}\cdot\mathbf{x} - s)} ds \left|\rho\right| d\rho d\boldsymbol{\nu}. \tag{4.4}$$

Rearranging the terms of (4.4) yields,

$$u\left(\mathbf{x}\right) = \int_{|\boldsymbol{\nu}|=1} \left(\frac{1}{2} \int_{-\infty}^{\infty} \left(\int_{-\infty}^{\infty} \left(Ru\right)\left(s, \boldsymbol{\nu}\right) e^{-2\pi i \rho s} ds \right) e^{2\pi i \rho s'} \left|\rho\right| d\rho \right) |_{s'=\boldsymbol{\nu}\cdot\mathbf{x}} \, d\boldsymbol{\nu},$$

or, as described in Section (4.2), $u\left(\mathbf{x}\right) = \left(R^* K\, R\, u\right)\left(\mathbf{x}\right)$.

4.5 Algebraic reconstruction tomography

We briefly review the tomographic reconstruction method known as Algebraic Reconstruction Tomography, or ART. The idea behind this approach to the CT problem is to set up the inversion of the Radon transform as a discrete, linear algebra problem. The solution of which yields an image.

To set up the ART problem we assume that the image is represented in a basis,

$$u\left(\mathbf{x}\right) = \sum_{l=1}^{L} c_l \chi_l\left(\mathbf{x}\right), \tag{4.5}$$

where $\{\chi_l\}_{l=1}^{L}$ are basis functions and c_l are real-valued coefficients. A common approach to setting up the ART problem is to choose l as a flattened index of pixels. Then χ_l are then the indicator functions

$$\chi_l = \begin{cases} 1 & \mathbf{x} \in l^{\text{th}} \text{ pixel} \\ \\ 0 & \text{otherwise} \end{cases}.$$

Given this approach to the ART problem the linear system is

$$\mathbf{Ac} = \mathbf{p},$$

where $\mathbf{A}$ is the matrix containing the information about which line in the sampled Radon transform goes through the support of which basis function χ_l, $\mathbf{c}$ is the vector of coefficients in (4.5), and $\mathbf{p}$ is a vector containing all the projections, placed one on top of another. The matrix $\mathbf{A}$ is quite large and dense. For example, assuming we have 256 projections, each with 256 samples, and L is also 256, the size of $\mathbf{A}$ is 65536×65536. Due to their size, these linear systems are typically solved in an iterative manner.

4.6 Review of "Rational approximations for tomographic reconstructions" [61]

In this section we summarize the paper "Rational approximations for tomographic reconstructions" by Reynolds, Beylkin, and Monzón [61]. We have included a copy of [61] in Appendix B.

In [61] we introduce a rational signal model in the CT problem with the goal of improving image resolution. The projection data collected in CT measurements, assuming that the object of interest is described by functions with jump discontinuities, contain at worst the same type of discontinuities. We choose linear combinations of rational functions to describe these projections since, as described in [13], they are able to represent functions with jump discontinuities with a small (near optimal) number of terms for a given accuracy threshold. As a result, our rational function model allows us to augment our data, i.e. double the number of samples per projection. This doubling is equivalent to extending the domain of the Fourier transform by a factor of two.

Using this signal model, we introduce a new Fourier domain algorithm, the polar quadrature inversion algorithm, or PQI. This algorithm uses nonlinear approximations (sums of exponentials) of DFT coefficients from each projection to extend data in the Fourier domain by a factor of two. To solve the imaging problem, the nonlinear approximations for each projection are evaluated on a quadrature grid designed for a bandlimit twice as large as the bandlimit of the original data (see [8]). Given data at the appropriate quadrature nodes the last step in the algorithm is to evaluate the inverse Fourier transform using the USFFT [28, 6, 53].

We compare the output of the PQI algorithm with the output of the standard FBP algorithm. Also, we compare the output of the PQI algorithm with the output of the FBP algorithm using augmented projections (projections whose sampling has been increased by a factor of two) as input. These numerical experiments demonstrate the improved resolution of reconstructed images produced by both the PQI algorithm and the FBP algorithm with augmented projections. Importantly, this increase in resolution comes without additional artifacts near sharp transitions in the image.

4.6.1 A rational model for signals

In Section 2 of [61] we demonstrate how use DFT coefficients of a sampled function to form the rational function g,

$$g(x) = a_0 + 2\mathrm{Re}\sum_{m=1}^{M} \frac{w_m}{e^{-2\pi i x + \eta_m} - 1}, \quad x \in [0, 1), \tag{4.6}$$

where $w_m \in \mathbb{C}$, $\eta_m \in \mathbb{C}$, $\mathrm{Re}(\eta_m) > 0$, and $a_0 \in \mathbb{R}$ is a constant. In [13] rational functions of the form (4.6) we computed using Fourier series coefficients. However, in CT problems the data are samples of tomographic projections, not Fourier series coefficients.

To extend our rational function theory to work with DFT coefficients we first note the functional form of the Fourier series coefficients of (4.6). The Fourier series coefficients are decaying exponentials in the form of

$$a_j = \sum_{m=1}^{M} w_m e^{-\eta_m j}, \ j = 1, 2, \ldots.$$

We show, both in Chapter 2, Section 5.1 of this thesis and Section 2 of [61] that the DFT of a sequence of

N samples of g, $\{g\,(n/N)\}_{n=0}^{N-1}$, is

$$\hat{g}_j = \sum_{m=1}^{M} \frac{w_m}{1-e^{-\eta_m N}} e^{-\eta_m j} + \sum_{m=1}^{M} \frac{\overline{w}_m}{1-e^{-\overline{\eta}_m N}} e^{-\overline{\eta}_m (N-j)}, \quad j = 1, \ldots, N-1. \tag{4.7}$$

Hence, the DFT coefficients and the Fourier series coefficients use the same nodes. We use this to our advantage when constructing the PQI algorithm.

4.6.2 Preliminary considerations

Section 3 in [61] reviews the necessary tools for constructing the PQI algorithm. The beginning of this section is background information on the CT problem, reviewed earlier in this Chapter. Since the PQI algorithm uses quadratures on the disk $D = \{\mathbf{p} \mid \|\mathbf{p}\| \leq 1\}$ in the Fourier domain for inversion, we review quadratures for band-limited exponentials on the disk [8]. These quadratures integrate exponentials with a bandlimit c with a user-selected accuracy ϵ. To obtain these quadratures, we discretize the integral

$$f\,(\mathbf{x}) = c^2 \int_D \hat{f}(c\mathbf{p}) e^{ic\mathbf{p}\cdot\mathbf{x}} d\mathbf{p} = \frac{c^2}{2} \int_{-\pi}^{\pi} \int_{-1}^{1} \hat{f}\,(c\rho, \theta)\, e^{ic\rho(x_1 \cos\theta + x_2 \sin\theta)}\, |\rho|\, d\rho d\theta, \tag{4.8}$$

where $\mathbf{x} = (x_1, x_2)$, $x_1, x_2 \in [-1, 1]$. We demonstrate in Section 3 of [61] that the number of angular modes required to represent the Fourier transform $\widehat{f}$ is proportional to the bandlimit c, justifying the use of the standard trapezoidal rule for discretizing the integral with respect to the angular variable. In the radial direction, the quadratures used integrate band-limited exponentials against the weight $|\rho|$. For the derivation of the quadratures used in the radial direction, see either [11] or [60]. Applying this quadrature to (4.8) yields

$$\left| f\,(\mathbf{x}) - \frac{\pi c^2}{N_\theta} \sum_{\nu=-N_\rho}^{N_\rho} \sum_{k=0}^{N_\theta - 1} w_\nu \hat{f}\,(c\rho_\nu, \theta_k)\, e^{ic\rho_\nu (x_1 \cos\theta_k + x_2 \sin\theta_k)} \right| \leq \epsilon, \quad x_1, x_2 \in [-1, 1] \tag{4.9}$$

where ρ_ν, $|\rho_\nu| < 1$, are (unequally spaced) quadrature nodes on the diameters, w_ν are corresponding quadrature weights, and $\theta_k = 2\pi k / N_\theta$. The number of samples in angle, N_θ, and the number of quadrature nodes in the radial direction both depend on the bandlimit c and the desired accuracy ϵ. The sum in (4.9) is evaluated on a $N_{image} \times N_{image}$ cartesian grid using USFFT at the cost of $\mathcal{O}\left(c^2 \log c\right)$ since N_{image}, N_ρ and N_θ are all proportional to c.

Also in Section 3 of [61] there is a review of the method of approximation via exponentials of [12]. We reviewed this algorithm in Chapter 2 of the thesis (see Algorithm 2.1), and do not repeat it here.

4.6.3 Calculating weights for tomography problems

The approach to calculating weights for periodic rational functions using ℓ^1 minimization, reviewed in this thesis in Chapter 2, Section 5.2, is the topic of Section 4 of [61]. This approach to calculating weights for rational functions is useful in tomography problems due to jumps present in the derivatives of tomographic projections. Such discontinuities cause problems for our rational approximations. These difficulties arise from the use of the DFT. If the decay of the Fourier series coefficients is algebraic, then the DFT coefficients will decay slowly. The decay will not be sufficient unless we use a very large sampling rate. Thus, when we apply Algorithm 2.1 to DFT coefficients to find nodes, some of the nodes may appear outside the unit circle.

Such nodes outside the unit circle are not consistent with our rational signal model (4.6), and must be discarded. The absence of these nodes causes errors in our approximations of tomographic projections. These errors express themselves near the sharp boundaries of the projections, since they typically contain higher frequency information. Therefore, the error due to these missing nodes should be localized when we compute our approximations. But this is not the case.

Since the standard method of calculating weights in Algorithm 2.1 uses ℓ^2 minimization, the reconstruction error, instead of being localized to the vicinity of sharp transitions, is spread out over the image. For examples of this phenomenon see Chapter 2 Figures 2.3-2.5 or, in Appendix B, Section B.4.

To mitigate the effects of missing nodes, we introduce in Section B.4 of [61] a method of calculating weights in the space domain using a Cauchy-like system. We also introduced this system in 2.5.2,

$$2\mathrm{Re}\sum_{m=1}^{M} w_m \left(\frac{1}{e^{-2\pi i n/N+\eta_m}-1} - \frac{1}{1-e^{-\eta_m N}} + 1 \right) = g\left(\frac{n}{N}\right) - \widehat{g}_0 = g_n - \widehat{g}_0, \tag{4.10}$$

where $n = 0, \ldots, N-1$. We denote the $N \times 2M$ matrix of this system as $\mathbf{C}$ and the right hand side $\{g_n - \widehat{g}_0\}_{n=0}^{N-1}$ as $\mathbf{g}$. Given this system we solve

$$\mathbf{Cw} = \mathbf{g}, \quad \text{with} \quad \mathrm{argmin}_{\mathbf{w}} \left\| \mathbf{Cw} - \mathbf{g} \right\|_2 . \tag{4.11}$$

We then verify if the residual is within the error tolerance our error tolerance defined in our application of Algorithm 2.1. If it is not (typically only for a small number of projections), we then solve

$$\mathbf{Cw} = \mathbf{g}, \quad \text{with} \quad \mathrm{argmin}_{\mathbf{w}} \left\| \mathbf{Cw} - \mathbf{g} \right\|_1 . \tag{4.12}$$

We restate that in the case where the solution of (4.11) satisfies the error tolerance, then the difference in (4.11) and (4.12) is negligible.

4.6.4 Polar quadrature inversion

Using the tools developed and reviewed thus far we review the polar quadrature inversion, or PQI, algorithm. This algorithm is also presented in Section B.5.

Algorithm 4.1

Give an accuracy $\epsilon > 0$ and a set of tomographic projections $\{g_n(k)\}_{n=0}^{N-1}$, $k = 0, \ldots, N_\theta - 1$,

(1) Compute the FFT of the projection data $\{g_n(k)\}_{n=0}^{N-1}$ for individual projection $k = 0, \ldots, N_\theta - 1$,

$$\widehat{g}_j(k) = \sum_{n=0}^{N-1} g_n(k) e^{-2\pi i j n/N}.$$

(2) Using subsets of the DFT coefficients $\widehat{g}_j(k)$, $j = 1, \ldots, 2\widetilde{N} + 1$ (where $\widetilde{N} = \lfloor N/4 + 1/2 \rfloor - 1$ and $\lfloor\ \rfloor$ denotes the floor function) construct its exponential approximation via Algorithm 2.1 from Chapter 2,

$$\left| \widehat{g}_j(k) - \sum_{m=1}^{M_k} w_{mk} e^{-c\eta_{mk} j} \right| \leq \epsilon, \quad j = 1, \ldots, \widetilde{N}, \quad \mathcal{R}e(\eta_{mk}) > 0. \tag{4.13}$$

We use only nodes inside the unit circle in (4.13). The weights for each exponential approximation k are calculated by (4.11). If necessary for some of the projections, i.e. if the ℓ^2 residual is not beneath our threshold, we recalculate those weights using (4.12).

(3) Form the continuous version of (4.13),

$$\widehat{g}^{ext}(k, \rho) = \sum_{m=1}^{M} w_{mk} e^{-c\tau_{mk}\rho}, \quad \rho \geq 0, \quad \mathcal{R}e(\tau_{mk}) > 0, \tag{4.14}$$

where $\tau_{mk} = \widetilde{N}\eta_{mk}$. Window the continuous function (4.14),

$$\widehat{g}^{bl}(k, \rho) = \widehat{g}^{ext}(k, \rho) W(\rho), \tag{4.15}$$

and extend the definition of (4.15) to include negative values of ρ via

$$\widehat{g}^{bl}(k, -\rho) = \overline{\widehat{g}^{bl}(k, \rho)}.$$

Evaluate the windowed approximations $\widehat{g}^{bl}(k,\rho)$ at the quadratures nodes ρ_ν, $\nu = -N_\rho, \ldots, N_\rho$, corresponding to the new bandlimit $c_{new} = 2c$ (note that this step includes extrapolation) for each $k = 0, \ldots, N_\theta - 1$.

(4) Obtain the image by computing the sum in (4.9) via the USFFT.

The operation count of our current implementation of the PQI algorithm is as follows: Step 1 of the algorithm requires $\mathcal{O}(N_\theta N \log N)$ operations. Step 2 currently requires $\mathcal{O}\left(N_\theta N^3\right)$ for finding the nodes. Finding the weights requires $\mathcal{O}\left(N_\theta N M^2\right)$ operations when using (4.11), and a similar operation count to solve (4.12) but with a larger constant. The remaining steps require $\mathcal{O}(N^2 \log N)$ operations.

However, there are methods of making this method faster. The operation count for finding the nodes in Step 2 may be reduced to $\mathcal{O}\left(N_\theta N M^2\right)$, where M is the number of terms in the exponential approximation (this may be achieved using randomized projections, see the review article [40]). Also, Step 2 is trivially parallelizable as the PQI algorithm treats each projection separately. Also of note, only a few projections require using (4.12), so the large constant associated with solving (4.12) is not an issue.

4.6.5 Numerical examples on the Shepp-Logan phantom

To conclude our review of [61] we look at the numerical experiments in the paper. All the experiments use test data consisting of 512 projections of the Shepp-Logan phantom [64], with 512 samples per projection. In our experiments with both the PQI and FBP algorithms we use the radial Hann window,

$$W(\rho) = \sin^2\left(\frac{\pi}{2}\rho - \frac{\pi}{2}\right), \quad \rho \in [0,1].$$

We perform two types of experiments: noiseless reconstruction and reconstruction after adding Gaussian noise to each projection.

Our noiseless experiments consist of three approaches to image reconstruction : forming the image with the standard FBP algorithm, then with the FBP algorithm using augmented projections (with twice the samples) as input, and finally with the PQI algorithm. Both the PQI algorithm and the FBP algorithm using augmented projections produced higher resolution results than the results from the standard FBP algorithm (see Section B.6.1). Zooming in on small areas of the reconstructions shows in detail the same

result: the PQI algorithm and the FBP algorithm using augmented projections produce higher resolution images without introducing additional artifacts (see Section B.6.2).

To test if the PQI algorithm is stable under the addition of noise we add a small amount of Gaussian white noise to the projections. This noise has zero mean and standard deviation of $2.5 * 10^{-4}$. This noise level is of the same order as the smallest features captured by the projections. The results of these experiments are shown in Section B.6.3. The reconstruction errors are the same as in Section B.6.1, except for the addition of a small amount of speckle.

Chapter 5

Generalization of reduction algorithms for functions of several variables

5.1 Introduction

We begin this chapter with a review of the reduction algorithm in [12]. Considering that one of the new results of the thesis is a generalization of this algorithm to higher dimensions, and that the derivation of the higher-dimensional generalization leans heavily on the derivation of the one-dimensional reduction algorithm, we see it fit to review not only the one-dimensional reduction algorithm but also its derivation.

The reduction algorithm for functions of a single variable, and the approximation by exponentials algorithm presented in Chapter 2, were both derived in [12] for a good reason; they are intimately related. The derivation of the reduction algorithm essentially the same as the derivation of the algorithm for generating approximations via linear combination of exponentials. The difference between the use of these algorithms is that the input data for the approximation algorithm is a set of samples of a function, while the starting point of the reduction algorithm is a suboptimal linear combination of exponentials.

Using the derivation of the reduction algorithm for functions a single variable as a guide, we extend the reduction algorithm to reduce separated representations of multivariate functions. Preliminary results have been reported in [7] for the case of real-valued coefficients and non-negative exponents. We further extend these results to include the case of complex-valued coefficients and exponents.

5.2 Preliminaries

5.2.1 Reduction algorithm for linear combinations of exponentials in one variable

Recall from Chapter 2 that for functions of one variable we consider the following approximation problem: given the accuracy $\epsilon > 0$, for a smooth function $f(x)$ find the minimal (or nearly minimal) number of complex weights w_m and complex nodes e^{t_m} such that

$$\left| f(x) - \sum_{m=1}^{M} w_m e^{t_m x} \right| \leq \epsilon \quad \forall x \in [0,1]. \tag{5.1}$$

In [12] problem (5.1) is formulated as a discrete problem: given $2N+1$ values of $f(x)$ on a uniform grid in $[0,1]$ and a target accuracy $\epsilon > 0$, we find the minimal number M of complex weights w_m and complex nodes γ_m such that

$$\left| f(\frac{k}{2N}) - \sum_{m=1}^{M} w_m \gamma_m^k \right| \leq \epsilon \quad \forall k,\, 0 \leq k \leq 2N. \tag{5.2}$$

The sampling rate $2N$ must be chosen as to oversample $f(x)$ and guarantee that the function can be accurately reconstructed from its samples. The nodes and weights in (5.2) depend on ϵ and N and, once obtained, the continuous approximation (5.1) is defined using (5.1) where the exponents are set to

$$t_m = 2N \log \gamma_m.$$

The input to the reduction algorithm is not a sampled function, but a suboptimal linear combination of exponentials that represents the function f, i.e.

$$f(x) = \sum_{m=1}^{M_0} b_m e^{-\tau_m x}, \tag{5.3}$$

where the number of terms, M_0, is excessive. Given a user-supplied accuracy $\epsilon > 0$, we construct a new function that is also a linear combination of exponentials,

$$g(x) = \sum_{m=1}^{M} w_m e^{-t_m x}, \tag{5.4}$$

such that

$$|f(x) - g(x)| \leq \epsilon, \quad x \in [0,1],$$

and where $M < M_0$. This new linear combination of exponentials has a minimal number of complex weights w_m and complex nodes e^{-t_m} for the accuracy $\epsilon > 0$. Depending on the function and/or problem under

consideration, we may measure the approximation error in (5.1) in a different way, e.g., we may use relative error. Following Algorithm 2.1, with an appropriate sampling rate $N \gg M_0$, we construct the Hankel matrix $(\mathbf{Hq})_{nn'} = h_{n+n'}$, where

$$h_n = f\left(\frac{n}{2N}\right) = \sum_{m=1}^{M_0} b_m e^{-\frac{\tau_m}{2N} n} = \sum_{m=1}^{M_0} b_m r_m^n. \tag{5.5}$$

and we denote $r_m = e^{-\tau_m/(2N)}$. We factor the Hankel matrix as

$$\mathbf{H} = \mathbf{VBV}^t,$$

where $\mathbf{V}$ is the $(N+1)\times M_0$ Vandermonde matrix $\mathbf{V}_{km} = r_m^k$ and $\mathbf{B} = \operatorname{diag}(b_1, \ldots, b_m)$. The matrix $\mathbf{H}$ has a null space of dimension $N+1-M_0$. The reduction algorithm achieves the same goal as Algorithm 2.1, i.e., it generates (near) optimal linear combinations of exponentials to approximate sequences of samples. However, there are two important differences between the algorithms: first, the input to the reduction algorithm is already represented as a (suboptimal) linear combination of exponentials. Second, the representation of the input data as a linear combination of exponentials allows us to solve the con-eigenvalue problem for $\mathbf{H}$ with an auxiliary matrix of size $M_0 \times M_0$ by eliminating the null space of $\mathbf{H}$ from consideration. The reduction algorithm eliminates the need to sample the input function f.

We review the derivation of the reduction method, taking as a point of departure (5.5). As in Chapter 2, we solve the con-eigenvalue problem

$$\sum_{n'=0}^{N} h_{n+n'} u_{n'} = \sigma \overline{u}_n, \quad n = 0, \ldots, N, \tag{5.6}$$

for $\sigma > 0$ and $\mathbf{u} = (u_0, \ldots, u_N) \neq 0$. Using (5.5) in (5.6), our equation reduces to

$$\sum_{m=1}^{M_0} b_m r_m^n u(r_m) = \sigma \overline{u}_n, \tag{5.7}$$

where $u(z) = \sum_{k=0}^{N} u_k z^k$. We multiply each side of (5.7) by z^n and sum over the index n to get

$$\sum_{m=1}^{M_0} b_m \frac{1-(r_m z)^{N+1}}{1-r_m z} u(r_m) = \sigma \overline{u(\overline{z})}. \tag{5.8}$$

Then, given the polar decomposition $b_m = \rho_m e^{i\theta_m}$, we define $c_m = \sqrt{\rho_m} e^{i\theta_m/2}$. We make the substitution $z = \overline{r}_k$ in (5.9) and multiply by $\overline{c}_k$ to get

$$\sum_{m=1}^{M_0} \overline{c}_k \frac{1-(r_m \overline{r}_k)^{N+1}}{1-r_m \overline{r}_k} c_m^2 u(r_m) = \sigma \overline{c_k u(r_k)}. \tag{5.9}$$

Defining $v_k = c_k u(r_k)$, we may write (5.9) as

$$\mathbf{A}\mathbf{v} = \sigma\overline{\mathbf{v}}, \tag{5.10}$$

where $\mathbf{A}$ is the auxiliary $M_0 \times M_0$ matrix

$$\mathbf{A}_{km} = \overline{c}_k \frac{1-(r_m\overline{r}_k)^{N+1}}{1-r_m\overline{r}_k} c_m. \tag{5.11}$$

The $M_0 \times M_0$ matrix $\mathbf{A}$ is typically much smaller than the $(N+1)\times(N+1)$ Hankel matrix $\mathbf{H}$ used in Algorithm 2.1. We use (5.8) to form the rational function

$$\overline{u(\overline{z})} = \frac{1}{\sigma}\sum_{m=1}^{M_0} b_m \frac{1-(r_m z)^{N+1}}{1-r_m z} v_m. \tag{5.12}$$

and solve for M_0 roots. To determine which roots we use in (5.4), we calculate the weights for all M_0 nodes and then drop the nodes corresponding to weights less than ϵ.

To find the weights of our reduced exponential sum, we solve the Vandermonde system in the least squares sense,

$$h_n = \sum_{m=1}^{M_0} w_m \gamma_m^n, \quad n = 0, \ldots, 2N$$

Since we solve a reduction problem and therefore have access to f (5.3), we proceed to form the normal equations explicitly. Let $\mathbf{h}$ denote the vector with entries h_n, $n = 0, \ldots, 2N$, $\mathbf{w}$ denote the vector with entries w_m, $m = 1, \ldots, M_0$, and $\mathbf{V}$ denote the Vandermonde matrix with entries $\mathbf{V}_{nm} = \gamma_m^n$, $m = 1, \ldots, M_0$ and $n = 0, \ldots, 2N$. Forming the normal equations $\mathbf{V}^*\mathbf{V}\mathbf{w} = \mathbf{V}^*\mathbf{h}$, we see that

$$\mathbf{V}^*\mathbf{h} = \sum_{m=1}^{M_0} b_m \sum_{n=0}^{2N} (r_m\overline{\gamma}_s)^n = \sum_{m=1}^{M_0} \frac{1-(r_m\overline{\gamma}_s)^{2N+1}}{1-(r_m\overline{\gamma}_s)} b_m.$$

and

$$\mathbf{V}^*\mathbf{V} = \sum_{m=1}^{M_0} w_m \sum_{n=0}^{2N} (\gamma_m\overline{\gamma}_s)^n = \sum_{m=1}^{M_0} \frac{1-(\gamma_m\overline{\gamma}_s)^{2N+1}}{1-(\gamma_m\overline{\gamma}_s)} w_m.$$

Thus, we arrive at the system for the weights w_m,

$$\sum_{m=1}^{M_0} \frac{1-(\alpha_m\overline{\gamma}_s)^{2N+1}}{1-(\alpha_m\overline{\gamma}_s)} b_m = \sum_{m=1}^{M_0} \frac{1-(\gamma_m\overline{\gamma}_s)^{2N+1}}{1-(\gamma_m\overline{\gamma}_s)} w_m. \tag{5.13}$$

Finding weights in this manner allows us to deal with a $M_0 \times M_0$ system of equations instead of that of size $(N+1)\times M_0$.

5.2.2 Extension of the reduction algorithm to functions with domain $[0, \infty)$

We may extend the reduction algorithm to consider the following problem: given a suboptimal linear combination of exponentials f, i.e.

$$f(x) = \sum_{m=1}^{M_0} b_m e^{-\tau_m x}, \tag{5.14}$$

(where M_0 is excessive) and a user-supplied accuracy $\epsilon > 0$, construct a new linear combination of exponentials,

$$g(x) = \sum_{m=1}^{M} w_m e^{-t_m x},$$

such that

$$|f(x) - g(x)| \leq \epsilon, \;\; x \in [0, \infty)$$

and where $M < M_0$. We study this problem in the discrete sense, i.e. we define

$$h_n = \sum_{m=1}^{M_0} b_m r^n, \tag{5.15}$$

where $r_m = e^{-\tau_m/(2N)}$ and N is chosen to sufficiently sample the function $f(x)$. The first step in the reduction algorithm for functions on the domain $[0, \infty)$ is to solve the con-eigenvalue problem using a infinite Hankel matrix, i.e.

$$\sum_{n'=0}^{\infty} h_{n+n'} u_{n'} = \sigma \overline{u}_n, \quad n = 0,\, 1,\, 2, \ldots. \tag{5.16}$$

Following the same steps as the derivation in Section 5.2.1, except summing over all the terms in the geometric series (instead of just summing over $n = 0, \ldots 2N$), we get

$$\sum_{m=1}^{M_0} b_m \frac{1}{1 - r_m z} u(r_m) = \sigma \overline{u(\overline{z})}. \tag{5.17}$$

Using the polar decomposition, the substitution $z = \overline{r}^k$, and multiplying by $\overline{c}_k$ yields

$$\sum_{m=1}^{M_0} \overline{c}_k \frac{1}{1 - r_m \overline{r}_k} c_m^2 u(r_m) = \sigma \overline{c_k u(r_k)}. \tag{5.18}$$

Finally, defining $\mathbf{v}$ as the vector with entries $v_k = c_k u(r_k)$ we may write (5.18) as

$$\mathbf{A}\mathbf{v} = \sigma \overline{\mathbf{v}}, \tag{5.19}$$

where $\mathbf{A}$ is the auxiliary $M_0 \times M_0$ matrix

$$\mathbf{A}_{km} = \overline{c}_k \frac{1}{1 - r_m \overline{r}_k} c_m. \tag{5.20}$$

We have again reduced our problem to an auxiliary $M_0 \times M_0$ con-eigenvalue problem. We use the con-eigenpair $(\sigma, \mathbf{v})$ and (5.17) to represent the con-eigenpolynomial,

$$\overline{u\left(\overline{z}\right)} = \frac{1}{\sigma} \sum_{m=1}^{M_0} b_m \frac{1}{1 - r_m z} v_m. \tag{5.21}$$

and solve for the roots $\{\gamma_m\}_{m=1}^{M_0}$, a subset of which contains the nodes we keep after our reduction. We choose nodes the same way as the case for the finite interval, by calculating the weights w_m, $m = 1, \ldots, M_0$ and discarding all the weights such that $|w_m| < \epsilon$.

To solve for weights in the reduction algorithm for functions on $[0, \infty)$ we solve the infinite Vandermonde system,

$$h_n = \sum_{m=1}^{M_0} w_m \gamma_m^n, \quad n = 0,\, 1,\, 2,\, \ldots$$

Since this is an semi-infinite Vandermonde system, we have no choice but forming the normal equations to solve for weights. We denote $\mathbf{h}$ as the (infinitely long) vector with entries h_n, $n = 0,\, 1,\, 2,\, \ldots,$ $\mathbf{w}$ as the vector with entries w_m, $m = 1, \ldots, M_0$, and $\mathbf{V}$ as the (infinite) Vandermonde matrix with entries $V_{nm} = \gamma_m^n$, $m = 1, \ldots, M_0$ and $n = 0,\, 1,\, 2,\, \ldots$. Forming the normal equations $\mathbf{V}^*\mathbf{V}\mathbf{w} = \mathbf{V}^*\mathbf{h}$, we get

$$\mathbf{V}^*\mathbf{h} = \sum_{m=1}^{M_0} b_m \sum_{n=0}^{\infty} r_m^n \overline{\gamma}_s^n = \sum_{m=1}^{M_0} \frac{1}{1 - (r_m \overline{\gamma}_s)} b_m.$$

and

$$\mathbf{V}^*\mathbf{V} = \sum_{m=1}^{M_0} w_m \sum_{n=0}^{\infty} (\gamma_m \overline{\gamma}_s)^n = \sum_{m=1}^{M_0} \frac{1}{1 - (\gamma_m \overline{\gamma}_s)} w_m.$$

Thus, we have reduced the infinite Vandermonde system for finding weights to the finite system

$$\sum_{m=1}^{M_0} \frac{1}{1 - (\alpha_m \overline{\gamma}_s)} b_m = \sum_{m=1}^{M_0} \frac{1}{1 - (\gamma_m \overline{\gamma}_s)} w_m. \tag{5.22}$$

5.2.3 Reduction algorithm

We review the steps of the reduction algorithms. Given a set of nodes, $\{r_m\}_{m=1}^{M_0}$ and weights, $\{b_m\}_{m=1}^{M_0}$, where the number of terms is excessive, along with a sampling rate $2N$ and a user-specified accuracy $\epsilon > 0$,

Algorithm 5.1

- form the matrix $\mathbf{A}$, using (5.11) for reduction over $[0,1]$ or (5.20) for reduction over the interval $[0,\infty)$.
- Solve the con-eigenvalue problem $\mathbf{A}\mathbf{v} = \sigma\overline{\mathbf{v}}$, (5.10) or (5.19), and choose a con-eigenvalue value σ_M such that $\sigma_M/\sigma_0 \approx \epsilon$.
- Using the con-eigenvector $\mathbf{v}$ corresponding to σ_M construct the rational representation of the con-eigenpolynomial $\overline{u\,(\overline{z})}$ using (5.12) for reduction over $[0,1]$ or (5.21) for reduction over $[0,\infty)$.
- Find the roots γ_m, $m = 1, \ldots M_0$ of the con-eigenpolynomial (5.12) for reduction over $[0,1]$ or (5.19) for reduction over $[0,\infty)$. A subset of these roots are the new nodes $\{\gamma_m\}_{m=1}^{M}$.
- Form the normal equations (5.13) for reduction over $[0,1]$ or (5.22) for reduction over $[0,\infty)$. Solve for the weights $\{w_m\}_{m=1}^{M_0}$. Discard all the weights such that $|w_m| < \epsilon$. If necessary, recalculate the weights using only the remaining nodes.

Remark 5.1 Let us remark on a method for solving the con-eigenvalue problem $\mathbf{A}\mathbf{v} = \sigma\overline{\mathbf{v}}$. We may use the augmented system

$$\begin{bmatrix} 0 & \overline{\mathbf{A}} \\ \mathbf{A} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ \overline{\mathbf{v}} \end{bmatrix} = \sigma \begin{bmatrix} \mathbf{v} \\ \overline{\mathbf{v}} \end{bmatrix},$$

changing the problem from a con-eigenvalue problem to a standard eigenvalue problem. The eigenvalues of this problem come in $\pm\sigma$ pairs. Indeed, we assuming that we have a solution of the eigenvalue problem

$$\begin{bmatrix} 0 & \overline{\mathbf{A}} \\ \mathbf{A} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} = \sigma \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}. \tag{5.23}$$

it is easy to see that $-\sigma$ with the corresponding eigenvector $\begin{bmatrix} \mathbf{x} \\ -\mathbf{y} \end{bmatrix}$ are also solutions. Hence, the eigenvalues of (5.23) always appear in $\pm\sigma$ pairs with corresponding eigenvectors $\begin{bmatrix} \mathbf{x} \\ \pm\mathbf{y} \end{bmatrix}$. For the solution of (5.42),

we use positive σ and, since $\begin{bmatrix} \overline{\mathbf{y}} \\ \overline{\mathbf{x}} \end{bmatrix}$ is also an eigenvector for the eigenvalue σ, we define the solution as $\mathbf{v} = \mathbf{x} + \overline{\mathbf{y}}$ unless $\mathbf{x} = -\overline{\mathbf{y}}$. If $\mathbf{x} = -\overline{\mathbf{y}}$, then we set $\mathbf{v} = i\mathbf{x}$.

Remark 5.2 Solving (5.12) or (5.21) is not a well-conditioned numerical procedure. Due to cancellations, solving for roots $\{\gamma_m\}_{m=1}^{M_0}$ in this manner can cause the loss of half of the digits of precision (see [41, Section 2.3] for more discussion on this issue). This loss of precision can be avoided by solving the con-eigenvalue problem using the method in [42].

5.2.4 The theory of Adamjan, Arov, and Krein

Let us briefly discuss a particular version of the theory of Adamjan, Arov, and Krein (AAK), which originally appeared in [2, 3, 4]. This theory was developed for approximating bounded functions on the unit circle by meromorphic functions with an optimal number of poles in the unit disk for a given L^∞ error. Specifically, we will review this theory, examine its shortcomings in applications to practical problems, and look at the special case of AAK applied to rational functions. More recent treatments of the AAK method may be found in [71] and [58]. This discussion follows the appendix in [41].

Let H^∞ denote the Hardy space of bounded analytic functions. Also denote

$$H_N^\infty = \left\{ \frac{g(z)}{(z-\eta_1)\dots(z-\eta_k)} \mid |\eta_j| < 1,\, k \le N,\, g \in H^\infty \right\}. \tag{5.24}$$

Suppose we have a function $f \in L^\infty([0, 2\pi))$, s.t.

$$f = \sum_{n=-\infty}^{\infty} f_n z^{-n}.$$

We consider the semi-infinite Hankel matrix associated with f and its SVD,

$$\mathbf{H} = \begin{bmatrix} f_1 & f_2 & f_3 & \dots \\ f_2 & f_3 & & \\ f_3 & & \ddots & \\ \vdots & & & \end{bmatrix}, \quad \mathbf{H}\mathbf{v}{=}\sigma_N\mathbf{w}, \;\; \mathbf{H}^*\mathbf{w}{=}\sigma_N\mathbf{v}, \tag{5.25}$$

where

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \end{bmatrix}, \quad \mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \end{bmatrix}.$$

Defining

$$v(z) = \sum_{j=1}^{\infty} v_j z^{j-1}, \quad w(z) = \sum_{j=1}^{\infty} w_j z^{-j},$$

the AAK theorem states that for $r(z) = f(z) - \sigma_N \frac{w(z)}{v(z)}$,

$$\|f - r\|_\infty = \inf_{g \in H_N^\infty} \|f - g\| = \sigma_N.$$

Algorithms based on this theory use truncated (finite) Hankel matrices to generate near-optimal rational functions, with poles η_k, $k = 1, \ldots N$. The truncation of these Hankel matrices may be problematic if the decay of the Fourier series coefficients is slow. Another potential source of error is the computation of residues. We compute N residues corresponding to the poles. However, these residues may not be sufficient to represent the analytic function $g(z)$ in (5.24).

If we apply AAK theory to a rational function (which may be approximated by another rational function that has fewer poles) we avoid the errors traditionally associated with applications of AAK theory. Specifically, this implementation allows us to work with matrices of a reasonable size and avoid truncation errors that appear when applying AAK to practical problems. We consider rational functions of the form

$$f(z) = f_0 + \sum_{m=1}^{M} \frac{\alpha_m z^{-1}}{1 - \gamma_m z^{-1}} + \sum_{m=1}^{M} \frac{\overline{\alpha}_m z}{1 - \overline{\gamma}_m z},$$

where α_m and γ_m are complex-valued and $0 < |\gamma_m| < 1$. For functions of this form the Fourier series coefficients are

$$f_n = \sum_{m=1}^{M} \alpha_m \gamma_m^{n-1}, \; n = 1, 2, \ldots.$$

Plugging these Fourier series coefficients into (5.25) yields

$$\begin{aligned} \sum_{j=1}^{\infty} \left(\sum_{m=1}^{M} \alpha_m \gamma_m^{i+j-2} \right) v_j &= \sum_{m=1}^{M} \alpha_m \gamma_m^{i-1} \sum_{j=1}^{\infty} \gamma_m^{j-1} v_j \\ &= \sum_{m=1}^{M} \alpha_m \gamma_m^{i-1} v(\gamma_m) \\ &= \alpha w_i, \; i = 1, 2, \ldots. \end{aligned}$$

Multiplying both sides by z^{i-1} and summing yields

$$\sum_{m=1}^{M} \frac{\alpha_m}{1-\gamma_m z} v(\gamma_m) = \sigma z^{-1} w\left(z^{-1}\right). \tag{5.26}$$

Similarly, we have

$$\sum_{m=1}^{M} \frac{\overline{\alpha}_m}{1-\overline{\gamma}_m z} \overline{\gamma}_m^{-1} w\left(\overline{\gamma}_m^{-1}\right) = \sigma v(z). \tag{5.27}$$

We evaluate $z = \overline{\gamma}_n$ in (5.26) and $z = \gamma_n$ in (5.27),

$$\sum_{m=1}^{M} \frac{\alpha_m}{1-\gamma_m \overline{\gamma}_n} v(\gamma_m) = \sigma \overline{\gamma}_n^{-1} w\left(\overline{\gamma}_n^{-1}\right)$$

$$\sum_{m=1}^{M} \frac{\overline{\alpha}_m}{1-\overline{\gamma}_m \gamma_n} \overline{\gamma}_m^{-1} w\left(\overline{\gamma}_m^{-1}\right) = \sigma v(\gamma_n),$$

and symmetrize both equations,

$$\sum_{m=1}^{M} \frac{\alpha_m^{\frac{1}{2}} \overline{\alpha}_n^{\frac{1}{2}}}{1-\gamma_m \overline{\gamma}_n} \alpha_m^{\frac{1}{2}} v(\gamma_m) = \sigma \overline{\alpha}_n^{\frac{1}{2}} \overline{\gamma}_n^{-1} w\left(\overline{\gamma}_n^{-1}\right)$$

$$\sum_{m=1}^{M} \frac{\overline{\alpha}_m^{\frac{1}{2}} \alpha_n^{\frac{1}{2}}}{1-\overline{\gamma}_m \gamma_n} \overline{\alpha}_m^{\frac{1}{2}} \overline{\gamma}_m^{-1} w\left(\overline{\gamma}_m^{-1}\right) = \sigma \alpha_n^{\frac{1}{2}} v(\gamma_n).$$

We define the vectors $\mathbf{p}$ and $\mathbf{q}$ with the entries $p_m = \alpha_m^{\frac{1}{2}} v(\gamma_m)$ and $q_m = \overline{\alpha}_m^{\frac{1}{2}} \overline{\gamma}_m^{-1} w\left(\overline{\gamma}_m^{-1}\right)$, and the positive definite matrix $\mathbf{C}$ with entries

$$C_{mn} = \frac{\alpha_m^{\frac{1}{2}} \overline{\alpha}_n^{\frac{1}{2}}}{1-\gamma_m \overline{\gamma}_n}.$$

The SVD equations in (5.25) may now be written as

$$\begin{cases} \mathbf{C}\mathbf{p} = \sigma \mathbf{q} \\ \mathbf{C}\overline{\mathbf{q}} = \sigma \overline{\mathbf{p}}. \end{cases}$$

These equations reduce further to a finite con-eigenvalue problem. Defining $\mathbf{x} = \mathbf{p} + \overline{\mathbf{q}}$, we observe that

$$\mathbf{C}\mathbf{x} = \sigma \overline{\mathbf{x}}.$$

If $\mathbf{x} = 0$ we observe that $i\mathbf{p} = \overline{i\mathbf{q}}$ and

$$\mathbf{C}(i\mathbf{p}) = \sigma \overline{i\mathbf{p}}.$$

5.3 Extension of the reduction algorithm from [12] to higher dimensions

Let us extend Algorithm 5.1 to functions of many variables. There are two such extensions presented in this chapter. First, we extend the reduction algorithm for functions of a single variable on a finite interval to that of functions of many variables on a finite domain. Then, we modify the reduction algorithm for functions of many variables to include the case of an semi-infinite domain.

We first formulate the reduction problem for functions of many variables. Given a function $F(\mathbf{x}) = F(x_1, \ldots, x_d)$,

$$F(x_1, \ldots, x_d) = \sum_{m=1}^{M_0} b_m e^{-\tau_{1,m} x_1} \ldots e^{-\tau_{d,m} x_d}, \tag{5.28}$$

on $[0,1]^d$, $d > 1$, where the coefficients b_m are complex valued, and $Re(\tau_{l,m}) > 0$, the discrete version of this problem may be formulated as that of reducing the number of terms in

$$F(\frac{k_1}{2N}, \ldots, \frac{k_d}{2N}) = \sum_{m=1}^{M_0} b_m e^{-\tau_{1m} \frac{k_1}{2N}} \ldots e^{-\tau_{dm} \frac{k_d}{2N}}, \tag{5.29}$$

where $0 \leq k_1, \ldots k_d \leq 2N$, and N is large enough to oversample F. For functions on a uniform grid in $[0,1]^d$, the number of points required in this formulation is $(2N+1)^d$ which is prohibitive for large dimension d. Thus, we replace the problem (5.29) with a reduction problem. Let us denote $e^{-\tau_{j,m}/(2N)} = \alpha_{j,m}$, $j = 1, \ldots, d$ and $m = 1, \ldots, M_0$. Using this notation we rewrite (5.29) as

$$F(k_1, \ldots k_d) = \sum_{m=1}^{M_0} b_m \alpha_{1,m}^{k_1} \ldots \alpha_{d,m}^{k_d}. \tag{5.30}$$

We state the multivariate reduction problem as follows. Given (5.30), where the number of terms is excessive, and a user-specified accuracy $\epsilon > 0$, find a new representation of the same form but with fewer terms,

$$G(k_1, \ldots k_d) = \sum_{m=1}^{M} w_m \gamma_{1,m}^{k_1} \ldots \gamma_{d,m}^{k_d}, \tag{5.31}$$

where $M < M_0$ and

$$|G(k_1, \ldots, k_d) - F(k_1, \ldots, k_d)| \leq \epsilon, \quad \text{for } 0 \leq k_1, \ldots k_d \leq 2N.$$

For simplicity, we assume that the range of all indices is the same, $0 \leq k_1, \ldots k_d \leq 2N$.

5.3.1 Finding nodes $\gamma_{1,m} \dots \gamma_{d,m}$

We consider the multidimensional Hankel operator $\mathbf{H}$ defined by

$$h_{k_1,\dots k_d} = \sum_{m=1}^{M_0} b_m \alpha_{1,m}^{k_1} \dots \alpha_{d,m}^{k_d}, \tag{5.32}$$

where $0 \le k_1, \dots k_d \le 2N$. As in Algorithm 5.1 we solve a con-eigenvalue problem,

$$\mathbf{H}u = \sigma \overline{u},$$

i.e.,

$$\sum_{0 \le l_1,\dots,l_d \le N} h_{k_1+l_1,\dots,k_d+l_d} u_{l_1,l_2,\dots,l_d} = \sigma \bar{u}_{k_1,\dots,k_d}. \tag{5.33}$$

Assuming that there exist solutions of the form

$$u_{l_1,\dots,l_d} = \sum_{m=1}^{M_0} a_m \, \overline{\alpha}_{1,m}^{l_1} \, \dots \, \overline{\alpha}_{d,m}^{l_d}, \tag{5.34}$$

our first step is to find the coefficients a_m. Substituting (5.32) into (5.33), we obtain

$$\sum_{m=1}^{M_0} b_m \sum_{0 \le l_1,\dots,l_d \le N} \alpha_{1,m}^{k_1+l_1} \, \dots \, \alpha_{d,m}^{k_d+l_d} \, u_{l_1,\dots,l_d} = \sigma \bar{u}_{k_1,\dots,k_d}. \tag{5.35}$$

Let us define a multivariate polynomial

$$u(x_1, \dots, x_d) = \sum_{0 \le l_1,\dots,l_d \le N} u_{l_1,\dots,l_d} x_1^{l_1} \dots x_d^{l_d},$$

so that

$$\sum_{m=1}^{M_0} b_m \alpha_{1,m}^{k_1} \, \dots \, \alpha_{d,m}^{k_d} \, u(\alpha_{1,m}, \dots, \alpha_{d,m}) = \sigma \bar{u}_{k_1,\dots,k_d}. \tag{5.36}$$

We denote

$$q_m = u(\alpha_{1,m}, \dots, \alpha_{d,m}).$$

Multiplying equations in (5.36) by $x_1^{k_1} \cdot \dots \cdot x_d^{k_d}$ and summing over indices, we obtain

$$\sum_{m=1}^{M_0} b_m \frac{1-(\alpha_{1,m}x_1)^{N+1}}{1-\alpha_{1,m}x_1} \dots \frac{1-(\alpha_{d,m}x_d)^{N+1}}{1-\alpha_{d,m}x_d} q_m = \sigma \overline{u(\overline{x}_1, \dots, \overline{x}_d)}. \tag{5.37}$$

Evaluating (5.37) at $x_1 = \overline{\alpha}_{1,n}$, $x_2 = \overline{\alpha}_{2,n}$,..., $x_d = \overline{\alpha}_{d,n}$, we obtain

$$\sum_{m=1}^{M_0} b_m \frac{1-(\alpha_{1,m}\overline{\alpha}_{1,n})^{N+1}}{1-\alpha_{1,m}\overline{\alpha}_{1,n}} \dots \frac{1-(\alpha_{d,m}\overline{\alpha}_{d,n})^{N+1}}{1-\alpha_{d,m}\overline{\alpha}_{d,n}} q_m = \sigma \overline{q_n} \tag{5.38}$$

Let us introduce matrix $\mathbf{F}$ with entries

$$F_{nm} = \frac{1-(\alpha_{1,m}\overline{\alpha}_{1,n})^{N+1}}{1-\alpha_{1,m}\overline{\alpha}_{1,n}} \cdots \frac{1-(\alpha_{d,m}\overline{\alpha}_{d,n})^{N+1}}{1-\alpha_{d,m}\overline{\alpha}_{d,n}}. \tag{5.39}$$

Each component,

$$\frac{1-(\alpha_{j,m}\overline{\alpha}_{j,n})^{N+1}}{1-\alpha_{j,m}\overline{\alpha}_{j,n}},$$

may be shown to be a self-adjoint, positive definite matrix. The Kronecker product of these components, the matrix $\mathbf{F}$, is therefore also positive definite (see e.g. [44, Theorem 7.5.3]). We have

$$\sum_{m=1}^{M_0} F_{nm} b_m q_m = \sigma \overline{q}_n. \tag{5.40}$$

Let us write $b_m = \rho_m e^{i\theta_m}$ and set $c_m = \rho_m^{\frac{1}{2}} e^{\frac{i\theta_m}{2}}$. Multiplying (5.40) by $\overline{c}_n$ yields,

$$\sum_{m=1}^{M_0} \overline{c}_n F_{nm} c_m (c_m q_m) = \sigma \overline{c}_n \overline{q}_n. \tag{5.41}$$

Introducing the $M_0 \times M_0$ positive definite matrix $\mathbf{M}$ with entries $M_{nm} = \overline{c}_n F_{nm} c_m$ and the vector $\mathbf{p}$ with entries $p_m = c_m q_m$, we write (5.41) as the con-eigenvalue problem

$$\mathbf{M}\mathbf{p} = \sigma \overline{\mathbf{p}}. \tag{5.42}$$

Since $\mathbf{M}$ is a positive define matrix, there exist M_0 positive con-eigenvalues and corresponding linearly independent con-eigenvectors solving (5.42) (see [44, Theorem 4.6.11] or [12]). Our approach to solving (5.42) is to define an extended matrix and set up the eigenvalue problem

$$\begin{bmatrix} 0 & \overline{\mathbf{M}} \\ \mathbf{M} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{p} \\ \overline{\mathbf{p}} \end{bmatrix} = \sigma \begin{bmatrix} \mathbf{p} \\ \overline{\mathbf{p}} \end{bmatrix}. \tag{5.43}$$

For details on solving this problem, see Remark 5.1.

Solving the eigenvalue problem (5.43) yields the solution of (5.42), $(\sigma, \mathbf{p})$ and, therefore, $q_m = u(\alpha_{1,m}, \ldots, \alpha_{d,m})$. From (5.37), we obtain a separated rational form for the multivariate eigenpolynomial

$$u(x_1, \ldots, x_d) = \frac{1}{\sigma} \sum_{m=1}^{M_0} \overline{b}_m \frac{1-(\overline{\alpha}_{1,m}x_1)^{N+1}}{1-\overline{\alpha}_{1,m}x_1} \cdots \frac{1-(\overline{\alpha}_{d,m}x_d)^{N+1}}{1-\overline{\alpha}_{d,m}x_d} \overline{q}_m, \tag{5.44}$$

which makes it possible to compute its values and the values of its derivatives.

By analogy with the one-dimensional theory, we expect the nodes $\gamma_{j,m}$ $j = 1, \ldots, d$ in (5.31) to lie on the hypersurface $u(x_1, \ldots, x_d) = 0$. Unlike the one-dimensional case, this is not a discrete set. Instead, we may look for a common solution to a set of d eigenpolynomials corresponding to con-eigenvalues that are less or equal to σ. Labeling the con-eigenvalues

$$\sigma_0 \geq \sigma_1 \geq \cdots \geq \sigma_M \geq \sigma_{M+1} \geq \cdots \geq \sigma_{M_0},$$

where M is index such that $\sigma_M/\sigma_0 < \epsilon$, we solve the following nonlinear system of equations,

$$u_l(x_1, \ldots, x_d) = 0, \quad l = M,\, M+1, \ldots,\, M+d-1,$$

where $u_l(x_1, \ldots, x_d)$ is the eigenpolynomial corresponding to the con-eigenvalue σ_l in (5.54).

5.3.2 Finding weights

For the same reason that the approximation problem is impractical for large dimensions d, we cannot use data on a multidimensional grid to solve for the weights in our reduction. Therefore, similar to the reduction for functions of one variable, we form the normal equations explicitly and solve a much smaller auxiliary system of equations. Once we have found the nodes $\gamma_{j,m}$ $j = 1, 2, \ldots, d$, we form the system to find the weights w_m,

$$\sum_{m=1}^{M_0} w_m \gamma_{1,m}^{k_1} \ldots \gamma_{d,m}^{k_d} = \sum_{m=1}^{M_0} b_m \alpha_{1,m}^{k_1} \ldots \alpha_{d,m}^{k_d}, \quad 0 \leq k_1, \ldots, k_d \leq 2N. \tag{5.45}$$

Forming the normal system of linear equations by applying the adjoint matrix to both sides of equation (5.45), we obtain the $M_0 \times M_0$ system of linear equations for w_m,

$$\sum_{m=1}^{M_0} w_m \frac{1 - \left(\gamma_{1,m}\overline{\gamma}_{1,n}\right)^{2N+1}}{1 - \gamma_{1,m}\overline{\gamma}_{1,n}} \ldots \frac{1 - \left(\gamma_{d,m}\overline{\gamma}_{d,n}\right)^{2N+1}}{1 - \gamma_{d,m}\overline{\gamma}_{d,n}} = \sum_{m=1}^{M_0} b_m \frac{1 - \left(\alpha_{1,m}\overline{\gamma}_{1,n}\right)^{2N+1}}{1 - \alpha_{1,m}\overline{\gamma}_{1,n}} \ldots \frac{1 - \left(\alpha_{d,m}\overline{\gamma}_{d,n}\right)^{2N+1}}{1 - \alpha_{d,m}\overline{\gamma}_{d,n}}. \tag{5.46}$$

After we solve for the weights w_m, $m = 1, \ldots, M_0$ we remove the the indices m from consideration such that $|w_m| < \sigma_M/\sigma_0$.

We note that when solving (5.45), we solve for the weights of all the roots of (5.44), $\gamma_{1,m} \ldots \gamma_{d,m}$, $m = 1, \ldots, M_0$. In practice this is not the case. We typically have a subset of the roots $\gamma_{1,m} \ldots \gamma_{d,m}$, $m = 1, \ldots, M_0$, since some of the roots have locations outside out area of interest, e.g. we would throw out

the index $\widetilde{m}$ where $\left|\gamma_{\widetilde{d},\widetilde{m}}\right| > 1$ for some $\widetilde{d} \in \{1, \dots d\}$. After we have removed terms with small weights, we repeat the calculation in (5.46) with only the remaining nodes to improve the accuracy of our approximation.

Let us also mention that in the reduction problem for multivariate functions the number of nodes used in our approximation (after thresholding the weights), i.e. (5.48), does not match the index of the con-eigenvalue.

5.4 Infinite case

In an important variant of our approach, we let indices $0 \leq k_1, \dots k_d$ extend to infinity. Namely, given

$$F(k_1, \dots k_d) = \sum_{m=1}^{M_0} b_m \alpha_{1,m}^{k_1} \dots \alpha_{d,m}^{k_d}, \tag{5.47}$$

we find a new representation of the same form but with fewer terms,

$$G(k_1, \dots k_d) = \sum_{m=1}^{M} w_m \gamma_{1,m}^{k_1} \dots \gamma_{d,m}^{k_d}, \tag{5.48}$$

where $M < M_0$ and

$$|G(k_1, \dots, k_d) - F(k_1, \dots, k_d)| \leq \epsilon, \quad \text{for } 0 \leq k_1, \dots k_d.$$

To derive this variant of our approach we will use both the derivation of the reduction algorithm for functions of a single variable on $[0, \infty)$ and the derivation of the reduction algorithm for multivariate functions on a finite interval.

5.4.1 Finding nodes

To find the nodes in the infinite case we repeat many of the same steps as in the reduction of multivariate functions on a finite interval. The difference is that we sum over all the terms in the resulting geometric series instead of just $2N+1$. The logic of extending to the infinite case for multivariable functions is the same as the extension of the reduction algorithm for functions of one variable to $[0, \infty)$.

The Hankel operator $\mathbf{H}$ is now defined by a semi-infinite tensor

$$h_{k_1, \dots k_d} = \sum_{m=1}^{M_0} b_m \alpha_{1,m}^{k_1} \dots \alpha_{d,m}^{k_d}. \quad 0 \leq k_1, \dots k_d \tag{5.49}$$

We solve

$$\mathbf{H}u = \sigma\overline{u},$$

i.e.,

$$\sum_{0\leq l_1,\ldots,l_d} h_{k_1+l_1,\ldots,k_d+l_d} u_{l_1,\ldots,l_d} = \sigma\bar{u}_{k_1,\ldots,k_d}, \quad 0 \leq k_1, \ldots k_d \tag{5.50}$$

Following the same steps as the derivation in Section 5.3, except summing over all the terms in the geometric series, yields

$$\sum_{m=1}^{M_0} b_m \frac{1}{1-\alpha_{1,m}x_1} \cdots \frac{1}{1-\alpha_{d,m}x_d} q_m = \sigma\overline{u(\overline{x}_1, \ldots, \overline{x}_d)}. \tag{5.51}$$

We evaluate at $x_1 = \overline{\alpha}_{1,n}$, $x_2 = \overline{\alpha}_{2,n}$,..., $x_d = \overline{\alpha}_{d,n}$, to obtain

$$\sum_{m=1}^{M_0} b_m \frac{1}{1-\alpha_{1,m}\overline{\alpha}_{1,n}} \cdots \frac{1}{1-\alpha_{d,m}\overline{\alpha}_{d,n}} q_m = \sigma\overline{q_n}, \tag{5.52}$$

and introduce the $\mathbf{F}$ matrix for multivariate reduction over an infinite interval

$$F_{nm} = \frac{1}{1-\alpha_{1,m}\overline{\alpha}_{1,n}} \cdots \frac{1}{1-\alpha_{d,m}\overline{\alpha}_{d,n}}. \tag{5.53}$$

From here the multivariate, infinite reduction algorithm proceeds with the same steps as the multivariate, finite reduction algorithm. We use (5.53) in equations (5.40)-(5.42), and solve (5.43) to get the vector $\mathbf{q}$, which we use to construct the separated representation

$$u(x_1, \ldots, x_d) = \frac{1}{\sigma} \sum_{m=1}^{M_0} \overline{b}_m \frac{1}{1-\overline{\alpha}_{1,m}x_1} \cdots \frac{1}{1-\overline{\alpha}_{d,m}x_d} \overline{q}_m. \tag{5.54}$$

Again, we expect the nodes $\gamma_{j,m}$ $j = 1, 2, \ldots, d$ to lie on the hypersurface $u(x_1, x_2, \ldots, x_d) = 0$. We look for a common solution to a set of d eigenpolynomials corresponding to con-eigenvalues that are less or equal to σ.

5.4.2 Finding weights

As we have shown multiple times in this chapter we solve for the weights w_m by forming the normal equations explicitly and solving a much smaller auxiliary system of equations. Given the nodes $\gamma_{j,m}$ $j =$

$1, 2, \ldots, d$, we form the system,

$$\sum_{m=1}^{M_0} w_m \gamma_{1,m}^{k_1} \ldots \gamma_{d,m}^{k_d} = \sum_{m=1}^{M_0} b_m \alpha_{1,m}^{k_1} \ldots \alpha_{d,m}^{k_d}, \quad 0 \leq k_1, \ldots, k_d \tag{5.55}$$

We then form the normal system of linear equations by applying the adjoint matrix to both sides of equation (5.55), and sum the resulting geometric series yielding,

$$\sum_{m=1}^{M_0} w_m \frac{1}{1 - \gamma_{1,m}\overline{\gamma}_{1,s}} \cdots \frac{1}{1 - \gamma_{d,m}\overline{\gamma}_{d,s}} = \sum_{m=1}^{M_0} b_m \frac{1}{1 - \alpha_{1,m}\overline{\gamma}_{1,s}} \cdots \frac{1}{1 - \alpha_{d,m}\overline{\gamma}_{d,s}}. \tag{5.56}$$

We solve for weights w_m, $m = 1, \ldots, M_0$ and remove from consideration the indices m such that $|w_m| < \epsilon$. We may then solve for the weights once more using only the nodes that survive our thresholding process.

5.5 Algorithm

We summarize our reduction algorithm for many variables. Given a set of complex-valued nodes $\alpha_{1,m}^{k_1}, \ldots, \alpha_{d,m}^{k_d}$ and weights b_m, $m = 1, \ldots M_0$, that form the suboptimal representation of a function (5.30), along with a sampling rate $2N$ and a user-specified accuracy $\epsilon > 0$,

Algorithm 5.2

- Form the extended matrix from (5.43). For the finite case on the grid $0 \leq k_1, \ldots k_d \leq 2N$ form the matrix $\mathbf{F}$ using (5.39). Otherwise, for the infinite case, form the matrix $\mathbf{F}$ using (5.53). Using the definition of $\mathbf{F}$, form the matrix $\mathbf{M}$ in (5.42).

- Solve the eigenvalue problem (5.43). The solution to this eigenvalue problem yields the solution to the con-eigenvalue problem (5.42), i.e. $(\sigma, \mathbf{p})$.

- Using the con-eigenpair $(\sigma, \mathbf{p})$, select d con-eigenvalues and their corresponding con-eigenvectors, $(\sigma_{m_1}, \mathbf{p}_{m_1}), \ldots, (\sigma_{m_d}, \mathbf{p}_{m_d})$. Using these form d rational functions $u_1(x_1, \ldots, x_d), \ldots, u_d(x_1, \ldots, x_d)$ using (5.44) (for the finite case) or (5.54) (for the infinite case). Set all the rational functions equal

to 0 and solve the system of equations

$$\begin{cases} u_1(x_1, \ldots, x_d) &= 0 \\ u_2(x_1, \ldots, x_d) &= 0 \\ \vdots & \vdots \quad \vdots \\ u_d(x_1, \ldots, x_d) &= 0 \end{cases},$$

for the roots $\nu_{1,m}, \ldots, \nu_{d,m}$, $m = 1, \ldots, M_0$.

- Solve for the weights of the new representation (5.31). First, calculate the weights for the roots $\gamma_{1,m}, \ldots, \gamma_{d,m}$, $m = 1, \ldots, M_0$ using (5.46) for the finite case or (5.56) for the infinite case. Threshold away all the nodes whose corresponding weights are less than ϵ. If necessary, after the thresholding step, calculate the weights again using either (5.46) or (5.56) to refine the weights.

5.6 Numerical examples

5.6.1 Reducing real-valued nodes and weights

For our first example we use the multivariate reduction algorithm to reduce a linear combination of real-valued nodes and weights in the form (5.30). Our example has two variables, x_1 and x_2, 33 weights b_m, and 33 sets of nodes α_{1m}, α_{2m}. We show the distribution of nodes in Figure 5.2. The values of the weights were randomly chosen in the interval $[0, 1]$.

The resulting nodes from two reduction experiments are shown in Figure 5.2. The first result shows two curves whose intersections are the nodes we use in our approximation. These curves are the intersections of con-eigenpolynomials $u_m(x_1, x_2)$ with the xy-plane, i.e. $u_m(x_1, x_2) = 0$.

In Table 5.1 we display numbers of terms, errors, and normalized of our reduction experiments on a function composed of real-valued nodes and weights. While we observe a decrease in our error when we increase the number of terms, we note that the errors do not match the normalized singular values. However, the rate of decrease in the error and the normalized singular values is about the same in terms of digits.

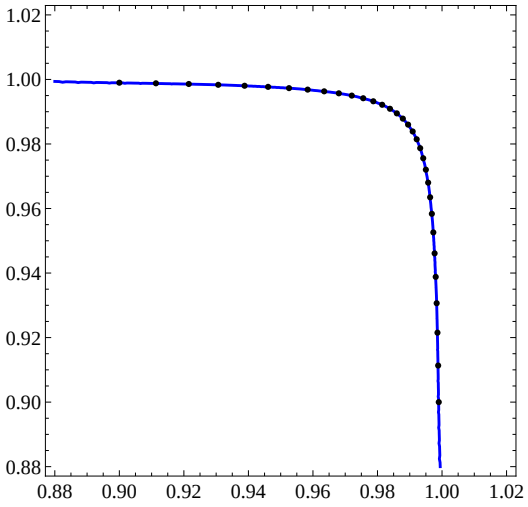


Figure 5.1: Positions of the real-valued nodes we use as input to our reduction algorithm. The blue curve is the intersection of the con-eigenpolynomial with the xy-plane, i.e. $u(x_1, x_2) = 0$, corresponding to $\sigma_{28}/\sigma_0 = 2.95 \cdot 10^{-11}$.

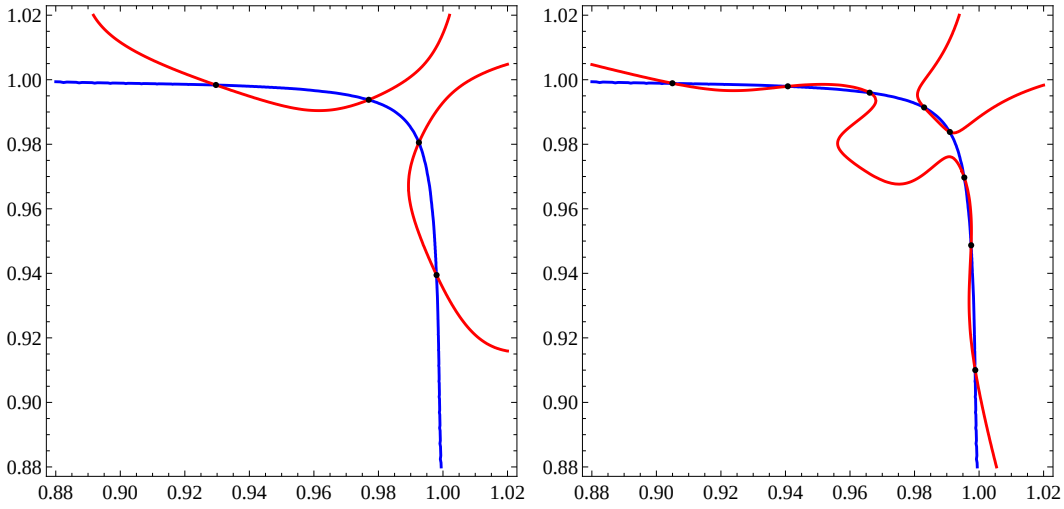


Figure 5.2: Intersections of con-eigenpolynomials curves on the xy-plane. These intersections are the output nodes of our reduction. (a) shows the intersection of eigenpolynomials corresponding to $\sigma_8/\sigma_0 = 2.90 \cdot 10^{-3}$ and $\sigma_{28}/\sigma_0 = 2.95 \cdot 10^{-11}$ in the xy-plane. (b) shows the intersection of eigenpolynomials corresponding to $\sigma_{16}/\sigma_0 = 2.34 \cdot 10^{-6}$ and $\sigma_{28}/\sigma_0 = 2.95 \cdot 10^{-11}$ in the xy-plane.

Number of Terms	Error	Normalized Singular Value
4	$8.80 \cdot 10^{-2}$	$2.90 \cdot 10^{-3}$
5	$1.72 \cdot 10^{-2}$	$5.13 \cdot 10^{-4}$
6	$3.11 \cdot 10^{-3}$	$7.41 \cdot 10^{-5}$
7	$1.70 \cdot 10^{-3}$	$1.28 \cdot 10^{-5}$
8	$9.72 \cdot 10^{-4}$	$2.34 \cdot 10^{-6}$
9	$2.68 \cdot 10^{-4}$	$3.41 \cdot 10^{-7}$
10	$6.24 \cdot 10^{-5}$	$5.64 \cdot 10^{-8}$

Table 5.1: Table of reduction errors for the real-valued experiment. We note that the errors do not follow the normalized singular values. However, the error does decrease by a similar amount each time we add an additional node.

5.6.2 Reducing complex valued nodes and weights

We also generate an example constructed with complex-valued nodes and weights. We use values of the function

$$g(x_1, x_2) = sinc(\pi x_1)\, sinc(\pi x_2)$$

approximated (with accuracy $1.45 \cdot 10^{-16}$ using one-dimensional algorithm) by the tensor product

$$g(\frac{10k_1}{52}, \frac{10k_2}{52}) = \left(\sum_{m=1}^{16} b_m \alpha_m^{k_1}\right)\left(\sum_{m=1}^{16} b_m \alpha_m^{k_2}\right), \quad k_1, k_2 = 0, 1, \ldots 52.$$

Hence, our input data to the reduction algorithm is a function with 256 terms. Applying multivariate reduction (two variables) over a finite domain with the con-eigenvalue $3.86 \cdot 10^{-6}$, and obtain the reduced approximation

$$f(k_1, k_2) = \sum_{m=1}^{101} w_m \gamma_{1,m}^{k_1} \gamma_{2,m}^{k_2}$$

with 101 terms (see Figure 5.3). We show the results of another reduction experiment in Figure 5.4, this time our resulting approximation has only 68 terms, but the maximum error is larger (see Table 5.2). We note that the distribution of nodes is similar to our result with 101 terms, but there are no terms in our reduced sum with real-valued nodes. Table 5.2 shows the results of our experiments on complex-valued weights and nodes in terms of number of terms, error, and normalized singular value. As in the real-valued example we see a decrease in the error as we add more terms.

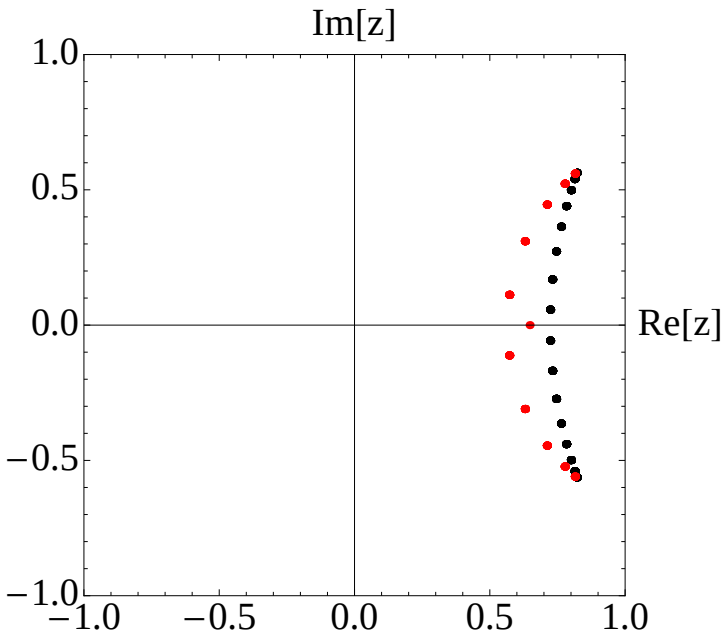


Figure 5.3: Positions of reduced nodes $\gamma_{1,m}$ (in red) and original nodes $\alpha_{1,m}$ (in black). Due to symmetry, there is an identical picture for $\gamma_{2,m}$ and $\alpha_{2,m}$.

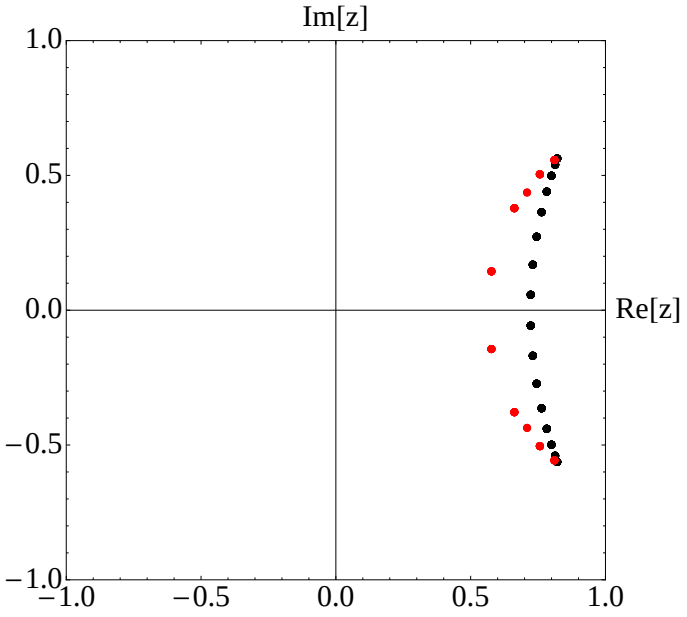


Figure 5.4: Positions of reduced nodes $\gamma_{1,m}$ (in red) and original nodes $\alpha_{1,m}$ (in black). Due to symmetry, there is an identical picture for $\gamma_{2,m}$ and $\alpha_{2,m}$.

Number of Terms	Error	Normalized singular Value
29	$9.44 \cdot 10^{-3}$	$1.10 \cdot 10^{-3}$
43	$2.12 \cdot 10^{-3}$	$7.94 \cdot 10^{-5}$
68	$9.86 \cdot 10^{-5}$	$9.04 \cdot 10^{-6}$
82	$1.77 \cdot 10^{-5}$	$2.97 \cdot 10^{-6}$
101	$1.97 \cdot 10^{-6}$	$5.52 \cdot 10^{-7}$

Table 5.2: Table of reduction errors for the complex-valued experiment. The errors do not follow the normalized singular values, but decrease as we add more nodes.

Chapter 6

Review of results and future work

In this thesis we have presented three contributions:

- A new algorithm for computing quadratures for band-limited functions.
- A new imaging algorithm for computerized tomography that increases image resolution.
- A generalization of the reduction algorithm in [12] to multivariate functions.

We briefly revisit these contributions and discuss future work on the topic.

6.1 Generalized Gaussian quadratures for band-limited exponentials

In [60], reviewed in Chapter 3 and included as Appendix A, we derive a new algorithm for constructing generalized Gaussian quadratures for exponentials integrated against a non-sign-definite weight function. Given a bandlimit $c > 0$ and an accuracy $\epsilon > 0$ by the user, this algorithm computes quadratures that integrate band-limited exponentials with bandlimit less than c and accuracy ϵ. Also in [60] we introduced a method of computing quadrature weights using ℓ^∞ minimization. To perform this calculation we use convex optimization.

Future work on this algorithm should improve its computational cost. To speed-up the computations, algorithms should take advantage of the structure of the involved matrices. The first step in the algorithm is to take the singular value decomposition of a Toeplitz matrix. Currently our implementation does not use this structure. Furthermore, the step that solves an eigenvalue problem can also be improved, since the involved matrices also have a special structure.

Another possible improvement may be an addition of an optimization step after the nodes and weights have been calculated. Ideally, this step would optimize both the nodes and the weights in the ℓ^∞ sense simultaneously. However, since the ℓ^∞ norm is not differentiable, this has proven to be a challenge.

6.2 Rational approximations for tomographic reconstructions

In [60], reviewed in Chapter 4 and included as Appendix B, we derive a new imaging algorithm for X-ray tomography, polar quadrature inversion (PQI). This algorithm increases resolution in the resulting images by extrapolating the available data in the Fourier domain using approximations via decaying exponentials. These extrapolated data fill a disk in Fourier space. To produce an image, the PQI algorithm uses polar quadratures for band-limited exponentials and the unequally spaced Fast fourier transform.

A feature not demonstrated in the PQI algorithm is our ability to interpolate in angular variable in the Fourier domain. By adding non-linear approximations in the angular variable, the PQI algorithm may require fewer projections. In medical terms, such an algorithm would significantly reduce the required dosage of X-rays.

There also needs to be a reasonable alternative to using the discrete Fourier transform to find nodes for exponential approximations. While we have presented a work-around for the case of exponentials with nodes just outside the unit circle, it would be better not to generate spurious nodes in the first place. One approach to this problem is to generate a suboptimal approximation via decaying exponentials. This approximation would not have an optimal number of terms, but would consist only of decaying exponentials. Then the reduction algorithm (over the interval $[0, \infty)$) would be used to reduce the length of the approximation. An example of a method for generating suboptimal approximations is the Richardson-like algorithm for moving nodes inside the unit circle, presented in Chapter 2 of this thesis. However, this method needs improvement, as the proximity requirement for nodes outside the unit circle is too restrictive.

6.3 Reduction algorithms for functions of several variables

In Chapter 5 we introduced a generalization of the reduction algorithm in [12] to functions of many variables. To derive the multivariate algorithm, we could not use the analysis of functions of one complex

variable. Instead, our approach was to derive the algorithm by mimicking the derivation of the reduction algorithm for functions of a single variable. Two examples showed that this method does indeed allow us to reduce linear combinations of multivariate exponentials, for both real- and complex-valued exponentials.

There are many possibilities for future work on this topic. One possible application is in computing quadratures for exponentials in higher dimensions, where we first compute suboptimal quadratures (there are many ways of doing this). Then we would apply our reduction algorithm to the suboptimal quadratures, arriving at near-optimal quadratures.

Another research topic regarding these algorithms is determining better methods of solving the root-finding problem. Whether solving for reduced quadrature nodes or for decaying exponentials, these methods require an ill-conditioned root-finding step. A great start towards solving these issues would be to generalize the work of [42] to multivariate problems.

Bibliography

[1] M. Abramowitz and I. A. Stegun. Handbook of mathematical functions. Dover Publications, 9 edition, 1970.

[2] V. M. Adamjan, D. Z. Arov, and M. G. Kreĭn. Infinite Hankel matrices and generalized Carathéodory-Fejér and I. Schur problems. Funkcional. Anal. i Priložen., 2(4):1–17, 1968.

[3] V. M. Adamjan, D. Z. Arov, and M. G. Kreĭn. Infinite Hankel matrices and generalized problems of Carathéodory-Fejér and F. Riesz. Funkcional. Anal. i Priložen., 2(1):1–19, 1968.

[4] V. M. Adamjan, D. Z. Arov, and M. G. Kreĭn. Analytic properties of the Schmidt pairs of a Hankel operator and the generalized Schur-Takagi problem. Math. USSR Sbornik, 15(1):34–75, 1971.

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

[6] G. Beylkin. On the fast Fourier transform of functions with singularities. Appl. Comput. Harmon. Anal., 2(4):363–381, 1995.

[7] G. Beylkin. On approximation of functions by exponential sums. Presentation, International Conference on Nonlinear Waves, Integrable Systems, and Applications, 2005.

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

[9] 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.

[10] G. Beylkin, R.D. Lewis, and L. Monzón. On the Design of Highly Accurate and Efficient IIR and FIR Filters. IEEE Trans. Signal Process., 60(8):4045–4054, 2012. http://dx.doi.org/10.1109/TSP.2012.2197397.

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

[12] G. Beylkin and L. Monzón. On approximation of functions by exponential sums. Appl. Comput. Harmon. Anal., 19(1):17–48, 2005.

[13] G. Beylkin and L. Monzón. Nonlinear inversion of a band-limited Fourier transform. Appl. Comput. Harmon. Anal., 27(3):351–366, 2009.

[14] G. Beylkin and L. Monzón. Approximation of functions by exponential sums revisited. Appl. Comput. Harmon. Anal., 28(2):131–149, 2010.

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

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

[17] R.N. Bracewell. Strip integration in astronomy. Aust. J. Phys., 9:198–217, 1956.

[18] 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.

[19] E. J. Candès and L. Demanet. The curvelet representation of wave propagators is optimally sparse. Comm. Pure Appl. Math., 58(11):1472–1528, 2005.

[20] E. J. Candès and D. L. Donoho. New tight frames of curvelets and optimal representations of objects with piecewise C^2 singularities. Comm. Pure Appl. Math., 57(2):219–266, 2004.

[21] E.J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. Information Theory, IEEE Transactions on, 52(2):489 – 509, feb. 2006.

[22] 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.

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

[24] A.M. Cormack. Representation of a function by its line integrals, with some radiological applications. J. Appl. Phys., 34:2722–2727, 1963.

[25] S. Deans. The Radon transform and some of its applications. Krieger Publishing Company, 1993.

[26] D.J. DeRosier and A. Klug. Reconstruction of three dimensional structures from electron micrographs. Nature, 217:130–134, 1968.

[27] R. Duda and P. Hart. Use of the hough transformation to detect lines and curves in pictures. Communications of the ACM, 15(1):11–15, 1972.

[28] A. Dutt and V. Rokhlin. Fast Fourier transforms for nonequispaced data. SIAM J. Sci. Comput., 14(6):1368–1393, 1993.

[29] S. Engelberg and E. Tadmor. Recovery of edges from spectral data with noise—a new perspective. SIAM J. Numer. Anal., 46(5):2620–2635, 2008.

[30] A. Gelb and D. Cates. Segmentation of images from Fourier spectral data. Commun. Comput. Phys., 5(2-4):326–349, 2009.

[31] A. Gelb and E. Tadmor. Detection of edges in spectral data. Appl. Comput. Harmon. Anal., 7(1):101–135, 1999.

[32] P. Gilbert. The reconstruction of a three-dimensional structure from projections and its applications to electron microscopy II. Direct methods. Proc. R. Soc. Lond. B., 182(1066):89–102, 1972.

[33] 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.

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

[35] I. S. Gradshteyn, I. M. Ryzhik, A. Jeffrey, and D. Zwillinger. Table of integrals, series, and products. Academic Press, 7 edition, 2007.

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

[37] L. Greengard and C. Stucchio. Spectral edge detection in two dimensions using wavefronts. Appl. Comput. Harmon. Anal., 30(1):69–95, 2011.

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

[39] K. Guo, D. Labate, and W. Lim. Edge analysis and identification using the continuous shearlet transform. Appl. Comput. Harmon. Anal., 27(1):24–46, 2009.

[40] Nathan Halko, Per-Gunnar Martinsson, and Joel A. Tropp. Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. SIAM Review, 53(2):217–288, 2011.

[41] T. Haut, G. Beylkin, and L. Monzón. Solving Burgers' equation using optimal rational approximations. Appl. Comput. Harmon. Anal., 2012. http://dx.doi.org/10.1016/j.acha.2012.03.004 (electronic).

[42] T. S. Haut and G. Beylkin. Fast and accurate con-eigenvalue algorithm for optimal rational approximations. SIAM J. Matrix Anal. Appl., 2012. in press, http://dx.doi.org/10.1137/1108211901, see also arXiv:1012.3196 [math.NA].

[43] G. Herman, A. Lent, and S. Rowland. Art: Mathematics and applications: A report on the mathematical foundations and on the applicability to real data of the algebraic reconstruction techniques. Journal of Theoretical Biology, 42(1):1 – 32, 1973.

[44] R. A. Horn and C. R. Johnson. Matrix analysis. Cambridge University Press, Cambridge, 1990.

[45] 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.

[46] 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.

[47] 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.

[48] 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.

[49] 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.

[50] G. Kutyniok and D. Labate. Introduction to shearlets. pages 1–38, 2012.

[51] H. J. Landau and H. O. Pollak. Prolate spheroidal wave functions, Fourier analysis and uncertainty II. Bell System Tech. J., 40:65–84, 1961.

[52] H. J. Landau and H. O. Pollak. Prolate spheroidal wave functions, Fourier analysis and uncertainty III. Bell System Tech. J., 41:1295–1336, 1962.

[53] J-Y. Lee and L. Greengard. The type 3 nonuniform FFT and its applications. J. Comput. Phys., 206(1):1–5, 2005.

[54] 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.

[55] F. Natterer. The Mathematics of Computerized Tomography. Wiley, NY, 1986.

[56] F. Natterer and F. Wübbeling. Mathematical Methods in Image Reconstruction. SIAM, Philadelphia, PA, 2001.

[57] X. Pan, E. Y. Sidky, and M. Vannier. Why do commercial CT scanner still employ traditional, filtered back-projection for image reconstruction? Inverse Problems, 25(12):123009, 36, 2009.

[58] J. R. Partington. An introduction to Hankel operators, volume 13 of London Mathematical Society Student Texts. Cambridge University Press, Cambridge, 1988.

[59] J. Radon. Über the bestimmung von funktionen durch ihre integralwerte längs gewisser mannigfaltigkeiten. Berichte Sächsische Akademie der Wissenschaften Leipzig Math.-Phys. Kl., 69:262–267, 1917.

[60] M. Reynolds, G. Beylkin, and L. Monzón. On generalized Gaussian quadratures for bandlimited exponentials. Appl. Comput. Harmon. Anal., 2012. In press, http://dx.doi.org/10.1016/j.acha.2012.07.002.

[61] M. Reynolds, G. Beylkin, and L. Monzón. Rational approximations for tomographic reconstructions. Inverse Problems, 2012. Submitted for publication.

[62] 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.

[63] J. A. Scales and A. Gersztenkorn. Robust methods in inverse theory. Inverse Problems, 4(4):1071–1091, 1988.

[64] L. Shepp and B. Logan. The Fourier reconstruction of a head section. IEEE Transactions on Nuclear Science, NS-21:21–43, 1974.

[65] D. Slepian. Prolate spheroidal wave functions, Fourier analysis and uncertainty IV. Extensions to many dimensions; generalized prolate spheroidal functions. Bell System Tech. J., 43:3009–3057, 1964.

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

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

[68] M. Unser. Sampling - 50 Years After Shannon. Proc. IEEE, 88(4):569–587, 2000.

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

[70] Ö. Yilmaz. Seismic data analysis: processing, inversion, and interpretation of seismic data. Number 10. SEG Books, 2001.

[71] N. Young. An Introduction to Hilbert Space. Cambridge University Press, 1988.

Appendix A

On Generalized Gaussian Quadratures for Band-limited Exponentials [60]

Matthew Reynolds, Gregory Beylkin and Lucas Monzón

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

Abstract

We review the methods in [11] and [69] for constructing quadratures for band-limited exponentials and introduce a new algorithm for the same purpose. As in [11], 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.

Key words: quadratures for band-limited exponentials, prolate spheroidal wave functions, ℓ^∞-minimization, Gaussian quadratures, generalized Gaussian quadratures, matrix pencil method, HSVD method

A.1 Introduction

We revisit the construction of quadratures for band-limited 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 [11] 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 [11] 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

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

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 $2c$ [11].

An alternative approach in [69] yields quadratures to integrate band-limited 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 [11] for $w = 1$ and [69] may be used interchangeably (we discuss this further in this paper). We note that the method in [11] allows us to

construct quadratures for a weight function that does not have to be positive (see e.g. [9, Section 5]).

We present a new approach for designing quadratures in $\mathcal{E}_c$ using a setup similar to [11] but computing nodes as eigenvalues of matrices rather than zeros of an eigenpolynomial. 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 [36] and check the results using our own implementation of the primal-dual potential reduction algorithm [54]. Additionally, for the weight $w(x) = 1$, we compare the accuracies and behavior of the error of our new quadratures and those obtained using [11] and [69]. 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 band-limited exponentials found applications in solving partial differential equations (see e.g. [15, 22, 62]). 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. [62]). Another application of quadratures for integration of band-limited exponentials (with a weight) yields a fast Discrete Fourier transform in polar and spherical coordinates in the Fourier space [8] (see also [5] 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 A.2 we briefly describe the two methods for finding Gaussian-type quadratures for band-limited functions. In Section A.3 we consider a method for finding quadrature nodes for band-limited functions as the eigenvalues of a matrix. In Section A.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 A.5. Finally, in Section A.6 we compare the new quadratures with those obtained in [11] and [69].

A.2 Preliminaries

A.2.1 Quadratures for band-limited functions via trigonometric moments

Let us briefly summarize a method in [11] for generating quadratures to integrate the family of exponentials $\left\{e^{ibx}\right\}_{|b|\leq c}$ with a real-valued weight function w. First, we compute the trigonometric moments

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

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

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

is oversampled. We then arrange the trigonometric moments $\{u_n\}_{n=-N}^{N}$ as the entries of a self-adjoint Toeplitz matrix $\mathbf{T} = \{u_{n-n'}\}_{0\leq n,n'\leq N}$. 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 $\left\{e^{ic\frac{n}{N}x}\right\}_{n=0,\ldots,N}$. 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 [9, Section 5]).

Computing the eigenvector $\mathbf{q}^{(s)} = [q_0, \ldots q_N]^t$ 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 $\{\gamma_j\}_{j=1}^N$, $\gamma_j \neq 0$, it is shown in [11, Theorem 4.1] that there exist weights $\{w_j\}_{j=1}^N$ 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

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

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, $j = 1, \ldots, N$ 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 $\mathbf{T}\mathbf{x}_0 = \mathbf{e}_0$, where $\mathbf{e}_0 = [1, 0, \ldots 0]^t$, 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, $j = 1, \ldots, N$ 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 [66, 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 [69] obtained by a different approach that uses the PSWFs.

A.2.2 Quadratures for band-limited functions via PSWFs

In [69] quadratures are constructed using the PSWFs, which form a basis for band-limited functions. The approach closely follows the classical method of obtaining Gaussian quadratures for polynomials. The PSWFs satisfy

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

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

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

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

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

Slepian and Pollak [67] observed that ψ_j are also the eigenfunctions of the differential operator

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

i.e., they coincide with the classical Prolate Spheroidal Wave functions of mathematical physics. In (A.2), the eigenvalues η_j form a strictly increasing, positive sequence.

Since the PSWFs form a Chebyshev system, the approach for computing quadratures in [69] first finds ψ_j by solving (A.2) and then computes the M nodes as zeros of ψ_M, $\psi_M(x_j) = 0$, $j = 1, \ldots M$. It is observed in [69] 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 [38]). 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 [33]. This fact that the PSWFs satisfy the second order differential equation in (A.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 [33] is applicable in that case as well.

A.3 Computing quadrature nodes as eigenvalues

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

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

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 A.8.1, we recover the Golub-Welsch algorithm [34].

A.3.2 Quadratures for inner products of band-limited exponentials

Let us now apply the approach illustrated in Section A.3.1 to finding quadratures for exponentials with bandlimit c. Since the collection of exponentials $\left\{e^{ibx}\right\}_{|b|\leq c}$ 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

$$\left|\int_{-1}^{1} e^{ibx} w\left(x\right) dx - \sum_{m=1}^{M} e^{ibx_m} w_m\right| < \epsilon, \quad |b| \leq c, \tag{A.3}$$

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

To solve this problem, we consider

$$G(b,b') = \int_{-1}^{1} e^{i\frac{b}{2}x} e^{-i\frac{b'}{2}x} w(x) dx, \;\; |b|, |b'| \leq c, \tag{A.4}$$

which we discretize as

$$\int_{-1}^{1} e^{i\frac{c}{2}\frac{n}{N}x} e^{-i\frac{c}{2}\frac{n'}{N}x} w(x) dx, \;\; n,n' = -N,\ldots,N, \tag{A.5}$$

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

$$\mathbf{G}_{nn'} = \int_{-1}^{1} e^{ic\frac{n}{N}x} e^{-ic\frac{n'}{N}x} w(x) dx, \;\; n,n' = 0,\ldots,N, \tag{A.6}$$

which over-samples 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 A.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

$$|\mathbf{G}_{nn'} - \mathbf{Q}_{nn'}| < \epsilon, \;\; n,n' = 0,\ldots,N, \tag{A.7}$$

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

$$\mathbf{Q}_{nn'} = \sum_{m=1}^{M} e^{icx_m\frac{n}{N}} w_m e^{-icx_m\frac{n'}{N}}, \;\; n,n' = 0,\ldots,N. \tag{A.8}$$

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 $\mathbf{B}$,

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

These submatrices may be written as

$$\mathbf{A} = \widetilde{\mathbf{X}}\mathbf{W}\mathbf{Y}, \;\; \mathbf{B} = \widehat{\mathbf{X}}\mathbf{W}\mathbf{Y}, \tag{A.9}$$

where

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

and

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

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,

$$\mathbf{E} = \mathrm{diag}\left(e^{icx_1/N}, \dots, e^{icx_M/N} \right). \tag{A.10}$$

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

$$\begin{aligned} \mathbf{A}^\dagger\mathbf{B} &= (\mathbf{W}\mathbf{Y})^\dagger \widetilde{\mathbf{X}}^\dagger\widehat{\mathbf{X}}\mathbf{W}\mathbf{Y} = (\mathbf{W}\mathbf{Y})^\dagger \left(\widetilde{\mathbf{X}}^\dagger\widetilde{\mathbf{X}}\right)\mathbf{E}\mathbf{W}\mathbf{Y} \\ &= (\mathbf{W}\mathbf{Y})^\dagger \mathbf{E}\left(\mathbf{W}\mathbf{Y}\right), \end{aligned} \tag{A.11}$$

using that $\mathbf{W}\mathbf{Y}$ has full rank and $\widetilde{\mathbf{X}}^\dagger\widetilde{\mathbf{X}} = \mathbf{I}_{M\times M}$. Thus, since the non-zero eigenvalues of $(\mathbf{W}\mathbf{Y})^\dagger \mathbf{E}\left(\mathbf{W}\mathbf{Y}\right)$ 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 (A.7), we need to form $\mathbf{A}^\dagger\mathbf{B}$ from the matrix $\mathbf{G}$ in (A.6). However, since the matrix $\mathbf{G}$ is extremely ill-conditioned (due to oversampling), we use instead its rank M approximation computed via the SVD,

$$\mathbf{G} = \mathbf{U}\Sigma\mathbf{V}^*. \tag{A.12}$$

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, $\Sigma_M = \mathrm{diag}\left(\sigma_0, \sigma_2, \dots \sigma_{M-1}\right)$. We then truncate (A.12) as

$$\mathbf{G}_M = \mathbf{U}_M\mathbf{\Sigma}_M\mathbf{V}_M^*, \tag{A.13}$$

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

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

Following (A.9), we write the corresponding matrices $\mathbf{A}_M$ and $\mathbf{B}_M$ as

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

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

$$\widetilde{\mathbf{U}}_M = \{\mathbf{U}_{\tilde{n}m}\}_{\substack{\tilde{n}=0,\ldots,N-1 \\ m=0,\ldots,M-1}}, \;\; \widehat{\mathbf{U}}_M = \{\mathbf{U}_{\hat{n}m}\}_{\substack{\hat{n}=1,\ldots,N \\ m=0,\ldots,M-1}}. \tag{A.14}$$

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

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

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 band-limited exponentials may be found as eigenvalues was also observed by Yu Chen [23].

A.3.3 Algorithm for computing quadrature nodes

We describe the algorithm, derived above, for computing quadrature nodes for band-limited functions given a weight function $w(x)$, bandlimit c, and accuracy ϵ. We address the computation of quadrature weights later in Section A.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 (A.14).

(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 [11] and [69], 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 [49] or the matrix-pencil method [45, 46, 47] 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.

A.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 A.8.2).

A.4.1 Finding weights via least squares

To find the weights w_m, $m = 1, \ldots, M$ that satisfy (A.3), 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 $\mathbf{Vw} = \mathbf{u}$, 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

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

The performance of our quadrature nodes using least squares weights is illustrated in Table A.2 and Figure A.5(a).

This approach to finding weights is related to the method used in [11] since we also solve a Vandermonde system. However, in [11] 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 $\mathbf{Vw} = \mathbf{u}$. Since $\mathbf{V}^*\mathbf{V}$ may be evaluated explicitly yielding a matrix of size $M \times M$, we solve $\mathbf{V}^*\mathbf{Vw} = \mathbf{V}^*\mathbf{u}$.

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

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

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.

A.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 [36]. 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 A.5(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 A.5(a)). The nodes and weights computed by both, least squares and ℓ^∞ minimization, are displayed in Table A.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 [69] 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.

A.5 Examples

A.5.1 An example of linear array antenna

Let us find quadrature nodes for the integral

$$u^{(c)}\left(B, l, \cos\theta\right) = \frac{1}{2}\int_{-1}^{1} I_0\left(\pi B\sqrt{1-\left(\frac{x}{l}\right)^2}\right) e^{icx\cos(\theta)}dx, \tag{A.15}$$

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 (A.15) is computed in [18, Eq. 6.7] by a different approach. However, our approach is simpler and yields similar results. Given the weight function

$$w(x) = I_0\left(\pi\sqrt{1-x^2}\right), \tag{A.16}$$

we obtain its trigonometric moments as

$$u_n^{(c)} = \frac{1}{2}\int_{-1}^{1} e^{icxn/N} w(x)\, dx = \operatorname{sinc}\left(\sqrt{(c\,\frac{n}{N})^2 - \pi^2}\right), \quad n = -N, \ldots N, \tag{A.17}$$

corresponding (up to a factor) to the samples of the radiation pattern. Identity (A.17) may be obtained extending formula 6.616.5 in [35, p. 698]. We also note that the weight function (A.16) is a scaled version of the so-called Kaiser window (see e.g. [48]).

We form

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

with $N = 252$ and $c = 10\pi$, and use Algorithm 1 in Section A.3.3. We truncate the SVD of the matrix $\mathbf{G}$ at the (normalized) singular value σ_{22}, $\sigma_{22}/\sigma_0 \approx 1.2 \cdot 10^{-15}$, yielding 22 quadrature nodes. Using the ℓ^∞ residual minimization (see Section A.4.2), we compute the weights resulting in a quadrature with maximum absolute error $\epsilon = 1.21 \cdot 10^{-14}$. We verify the accuracy of this quadrature numerically and illustrate the result in Figure A.1. This quadrature should be compared with that corresponding to the bandlimit 20π in [18, Table 6.3] since we integrate on $[-1, 1]$ instead of $[-1/2, 1/2]$ as in [18].

A.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

$$w(x) = (x - 1/10) \cdot e^{-(3\pi x/5 - 1/5)^2} + 1/(5e), \tag{A.18}$$

we calculate the nodes and weights for the bandlimit $c = 5\pi$, choosing $N = 127$ and the singular value $\sigma_{14}/\sigma_0 = 5.0 \cdot 10^{-14}$. Figure A.2(a) illustrates the weight function $w(x)$, $x \in [-1, 1]$, and Figure A.2(b)

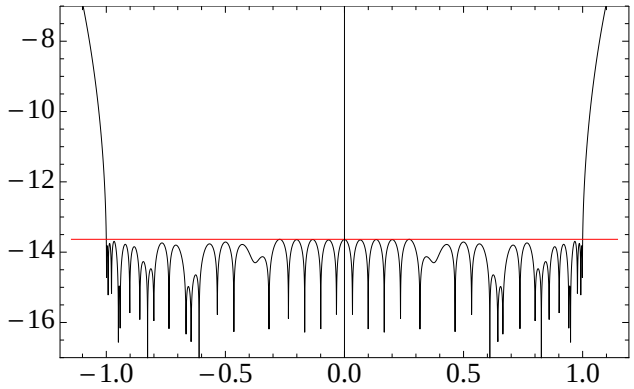


Figure A.1: The logarithm of the error of the quadrature with 22 nodes ($c = 10\pi$) for the weight function (A.16). The quadrature weights were generated via ℓ^∞ minimization. The horizontal line at $2.32 \cdot 10^{-14}$ indicates the maximum ℓ^∞ error within the bandwidth.

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 A.2(c), where the maximum error is $6.68 \cdot 10^{-14}$.

We note that the approach in [11] also allows us to obtain quadratures for weight functions $w(x)$ that are not sign-definite as is shown in [8].

A.6 Comparison with quadratures in [11] and [69]

Let us illustrate the impact of using weights obtained via ℓ^∞ minimization for the nodes computed in [69] and [11]. For this comparison we choose the weight function $w = 1$. In Table A.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 [69] and [11]. In Table (A.3) we compare the maximum errors using different approaches to computing weights.

Next, Figure A.3 compares the error of quadratures using nodes and weights from [69] and the same nodes but with weights found via ℓ^∞ minimization. A similar comparison for the quadratures from [11] is provided in Figure A.4. As expected, in all cases the ℓ^∞ minimization produces a better maximum error.

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 A.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.

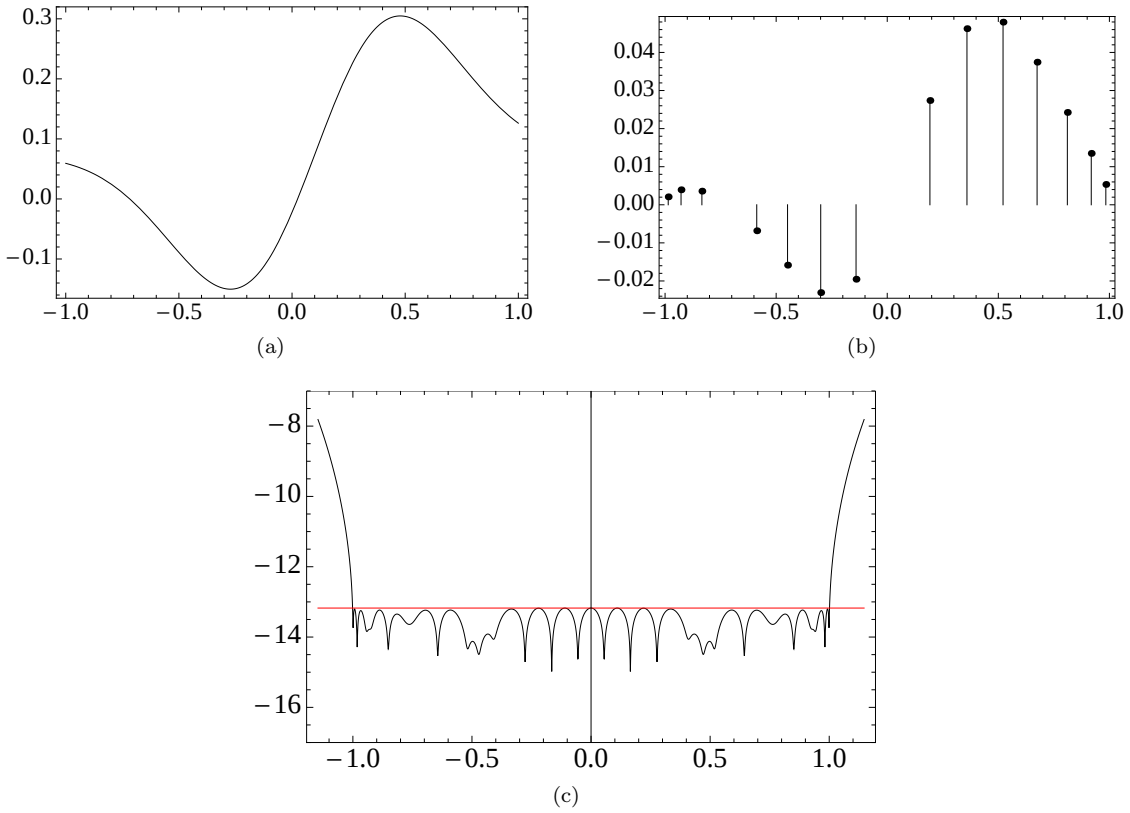


Figure A.2: (a) The weight function w in (A.18) and (b) the corresponding quadrature weights computed via ℓ^∞-minimization. We note that the quadrature weights follow the shape of the weight function w. In (c) we display the logarithm of the error of the quadrature with 14 nodes ($c = 5\pi$). The horizontal line at $6.68 \cdot 10^{-14}$ indicates the maximum ℓ^∞-error within the bandwidth.

		Maximum error from:		Maximum error using:	
c	# of nodes	[11]	[69]	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 A.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.

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 A.6. However, the attained maximum error is significantly worse than using all other approaches to compute weights.

A.7 Conclusions

In this paper we introduced a new algorithm for finding quadrature nodes for band-limited exponentials and considered two different approaches to compute the corresponding quadrature weights. As in [11], the

	Maximum error with nodes from:		
Weights	[11]	[69]	this paper
From [11] and [69]	$1.2 \cdot 10^{-7}$	$8.3 \cdot 10^{-8}$	
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 A.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 [11] and [69] 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.

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 [11], where such Vandermonde systems are square.

The new quadratures are slightly more accurate than those in [11] and [69], 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, in our experience, the nodes are always located inside the support of the weight function.

A.8 Appendix

A.8.1 Golub-Welsch algorithm

We show how to derive the well known Golub-Welsch algorithm [34] using the results in Section A.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

$$p_{n+1}(x) = (a_{n+1}x + b_{n+1})\, p_n(x) - c_{n+1} p_{n-1}(x), \tag{A.19}$$

where $p_{-1}(x) \equiv 0$, $p_0 \equiv 1$, $a_n > 0$ and $c_n > 0$ for $n = 0, \ldots, N-1$. Following [34], we write the recursion as the matrix equation

$$x\,\mathbf{p}(x) = \mathbf{T}\mathbf{p}(x) + (1/a_n)\, p_N(x)\, \mathbf{e}_n, \tag{A.20}$$

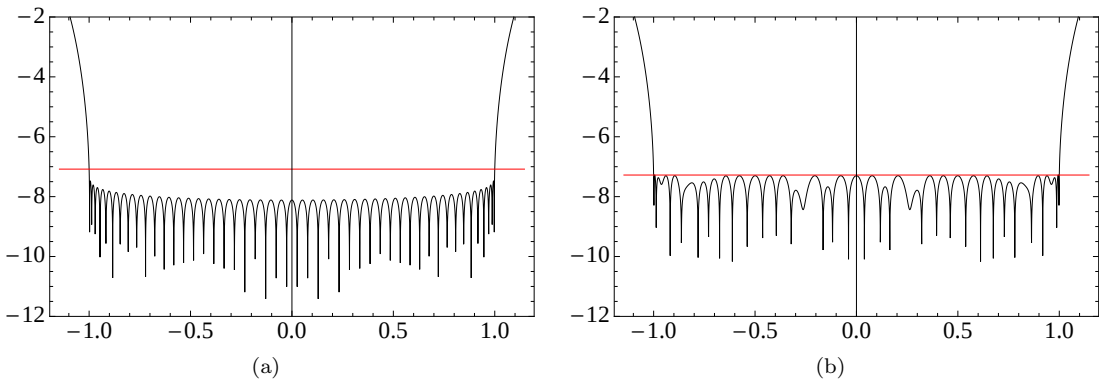


Figure A.3: Logarithm of the error of the 24 node quadrature for bandlimit $c = 50$. In (a) we use nodes and weights from [69] and in (b) nodes from [69] and weights generated via ℓ^∞ minimization. The horizontal lines at $8.30 \cdot 10^{-8}$ in (a) and at $5.26 \cdot 10^{-8}$ in (b) indicate the maximum ℓ^∞ error within the bandwidth.

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

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

Due to (A.20), $p_N(x_j) = 0$ 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 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 A.3.1, that is, via matrices of inner products. We define the matrices $\mathbf{A}$ and $\mathbf{B}$ by

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

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 (A.19) implies that the matrix $\mathbf{B}$ is tridiagonal. In fact, because of (A.19), $x\, p_n(x)$ is a linear combination of p_{n-1}, p_n and p_{n+1} which gives

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

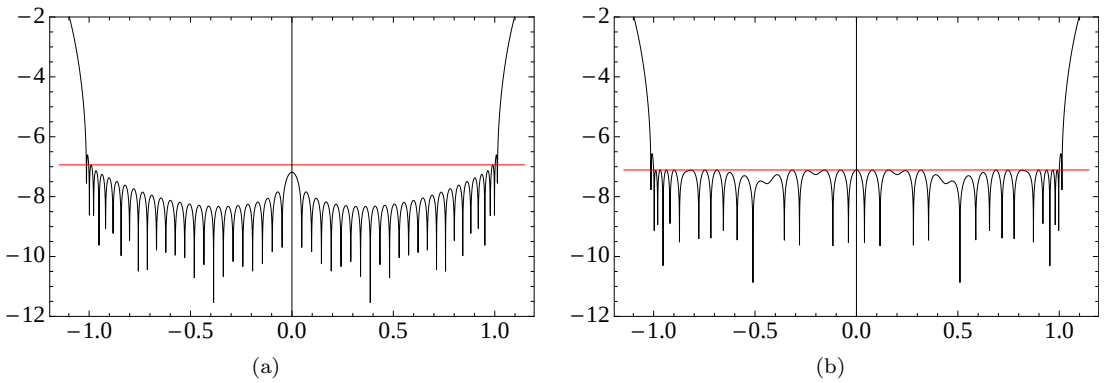


Figure A.4: Logarithm of the error of the 24 node quadrature for bandlimit $c = 50$. In (a) we use nodes and weights from [11] and in (b) nodes from [11] and weights generated via ℓ^∞ minimization. The horizontal lines at $1.15 \cdot 10^{-7}$ in (a) and at $7.76 \cdot 10^{-8}$ in (b) indicate the maximum ℓ^∞ error within the bandwidth.

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 $\mathbf{A} = \mathrm{diag}\left(\|p_0\|^2, \ldots, \|p_{N-1}\|^2 \right)$. 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.

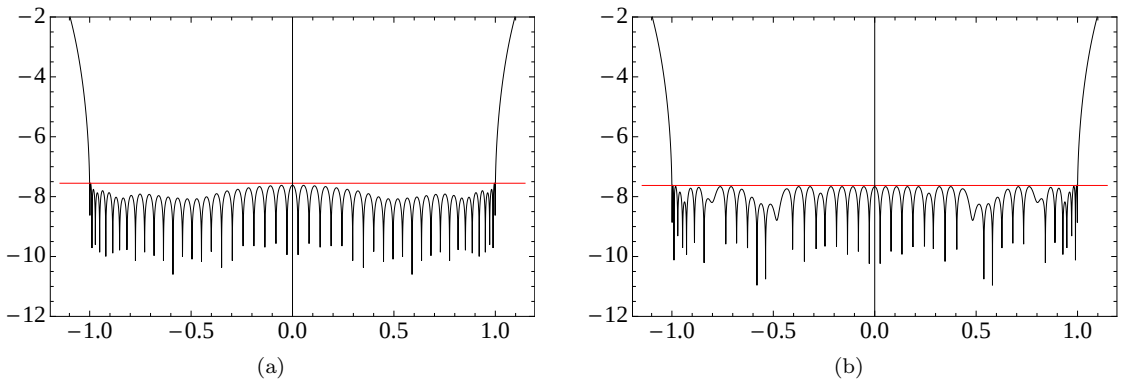


Figure A.5: Logarithm of the error of the 24 node quadrature for bandlimit $c = 50$ of this paper. In (a) we show the error using weights obtained via ℓ^2 minimization and in (b) using weights obtained via ℓ^∞ minimization. The horizontal lines at $2.80 \cdot 10^{-8}$ in (a) and at $2.36 \cdot 10^{-8}$ in (b) indicate the maximum ℓ^∞ error within the bandwidth.

A.8.2 Formulation of ℓ^∞ residual minimization as a second-order cone program

We review the primal-dual interior-point method of [54], the algorithm we implemented in extended precision to compare with the results obtained using CVX [36]. For further details we refer to [16].

We define a second order cone program (SOCP) as

$$\begin{aligned} &\text{minimize} && \mathbf{f}^t\mathbf{x} \\ &\text{subject to} && \|\mathbf{u}_i\| \le t_i, && i = 1,\dots,N, \\ &&& \mathbf{u}_i = \mathbf{A}_i\mathbf{x} + \mathbf{b}_i, && i = 1,\dots,N, \\ &&& t_i = \mathbf{c}_i^T\mathbf{x} + d_i, && i = 1,\dots,N, \end{aligned} \tag{A.21}$$

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

$$\begin{aligned} &\text{maximize} && -\sum_{i=1}^{N} \left(\mathbf{b}_i^t\mathbf{z}_i + d_i w_i\right) \\ &\text{subject to} && \sum_{i=1}^{N} \left(\mathbf{A}_i^t\mathbf{z}_i + \mathbf{c}_i w_i\right) = \mathbf{f}, && i = 1,\dots,N, \\ &&& \|\mathbf{z}_i\|_2 \le w_i, && i = 1,\dots,N \end{aligned} \tag{A.22}$$

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^∞

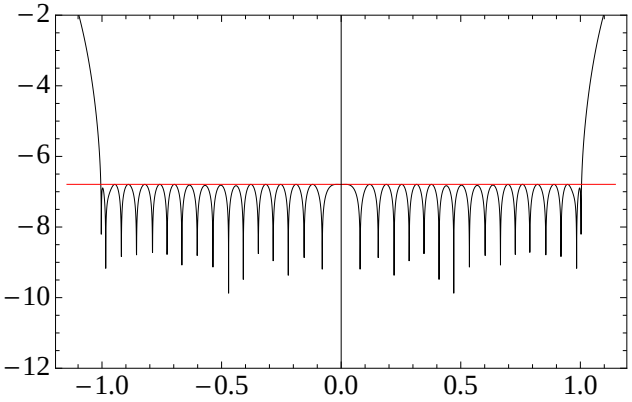


Figure A.6: Logarithm of the error of the 24 node quadrature for bandlimit $c = 50$ with weights generated by minimizing an ℓ^2 residual with an ℓ^∞ constraint. Although the error has a near perfect behavior, the maximum absolute error $1.62 \cdot 10^{-7}$ is worse than in Figures (A.3), (A.4) and (A.5).

residual minimization problem can be recast as a SOCP.

A.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 $i = 1, \ldots, p$, 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 (A.21) and setting $d_i = 0$, $i = 1, \ldots, p$, yields the SOCP for solving the ℓ^∞ residual minimization problem,

$$\text{minimize} \quad t$$

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

A.8.2.2 Primal-dual interior-point method

The primal-dual interior-point algorithm solves (A.21) 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 (A.21) and (A.22) 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. [54]). 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 $\phi\left(x, t\right)$,

$$\phi\left(\mathbf{u}, t\right) = \begin{cases} -log\left(t^2 - \left\|\mathbf{u}\right\|_2^2\right), & \left\|\mathbf{u}\right\|_2 < t \\ \infty & otherwise \end{cases},$$

which approaches infinity as $\left\|\mathbf{u}\right\|_2^2 \to t^2$, 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 $\nu \geq 1$ 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 $\varphi \to -\infty$ then $\eta \to 0$ 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 [54]).

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

$$\begin{bmatrix} \mathbf{H}^{-1} & \overline{\mathbf{A}} \\ \overline{\mathbf{A}}^t & 0 \end{bmatrix} \begin{bmatrix} \delta\mathbf{Z} \\ \delta\mathbf{x} \end{bmatrix} = \begin{bmatrix} -\mathbf{H}^{-1}(\rho\mathbf{Z} + \mathbf{g}) \\ 0 \end{bmatrix}, \tag{A.23}$$

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

$$\mathbf{H} = \begin{bmatrix} \nabla^2\phi(\mathbf{u}_1, t_1) & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \nabla^2\phi(\mathbf{u}_N, t_N) \end{bmatrix}, \quad \mathbf{g} = \begin{bmatrix} \nabla\phi(\mathbf{u}_1, t_1) \\ \vdots \\ \nabla\phi(\mathbf{u}_N, t_N) \end{bmatrix},$$

and

$$\overline{\mathbf{A}} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{c}_1^t \\ \mathbf{A}_2 \\ \mathbf{c}_2^t \\ \vdots \end{bmatrix}, \quad \mathbf{Z} = \begin{bmatrix} \mathbf{z}_1 \\ w_1 \\ \mathbf{z}_2 \\ w_2 \\ \vdots \end{bmatrix}, \quad \delta\mathbf{Z} = \begin{bmatrix} \delta\mathbf{z}_1 \\ \delta w_1 \\ \delta\mathbf{z}_2 \\ \delta w_2 \\ \vdots \end{bmatrix}, \quad \delta\mathbf{x} = \begin{bmatrix} \delta\mathbf{x}_1 \\ \vdots \\ \delta\mathbf{x}_n \end{bmatrix}.$$

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 (A.23) 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 (A.23) becomes ill conditioned, which results in indeterminate search directions.

Appendix B

Rational approximations for tomographic reconstructions [61]

Matthew Reynolds, Gregory Beylkin and Lucas Monzón

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

Abstract

We use optimal rational approximations of projection data collected in X-ray tomography to improve the image resolution. Under the assumption that the object of interest is described by functions with jump discontinuities, for each projection we construct its rational approximation with a small (near optimal) number of terms for a given accuracy threshold. This allows us to augment the measured data, i.e., double the number of available samples in each projection or, equivalently, extend (double) the domain of their Fourier transform. We also develop a new, fast, polar coordinate Fourier domain algorithm which uses our nonlinear approximation of projection data in a natural way.

Using the augmented projections of the Shepp-Logan phantom, we provide a comparison between the new algorithm and the standard Filtered Back-Projection (FBP) algorithm. We demonstrate that the reconstructed image has improved resolution without additional artifacts near sharp transitions in the image.

Key words: optimal rational approximations, signal models, X-ray tomography, Radon transform, polar grids, Unequally Spaced Fast Fourier Transform, Filtered Back Projection, image resolution

B.1 Introduction

As perceptively noted in [57], despite the development of many new algorithms for the inversion of the Radon transform, the quality of image reconstruction (in e.g., X-ray tomography) has not improved noticeably when compared with the output of the traditional Filtered Back-Projection (FBP) algorithm (see, for example [56]). This lack of improvement in image quality may be traced to the fact that the signal model for collected data is subject to the Nyquist sampling criterion. Since we are typically interested in reconstructing piece-wise continuous objects, the collected data (within the standard signal model) may be insufficient to resolve the image in the vicinity of discontinuities.

We introduce a different signal model for collected data and, as a consequence, for reconstruction. Assuming that the object of interest is described by functions with a limited number of jump discontinuities, the measured projections have, in the worst case, the same type of discontinuities (for most projections only the first derivative is discontinuous). As shown in [13], this implies that the projections are well approximated by proper rational functions yielding a robust recovery of signals from band-limited data. In this paper, we demonstrate that a rational model of a (periodic) signal allows us, in principle, to recover its entire Fourier series from a small number of samples. In practice, even though the presence of noise limits such recovery, a rational model still outperforms models based on the Nyquist sampling criterion. For objects with a limited number of isolated singularities, optimal rational approximations of projection data yield a significant improvement in resolution without introducing artifacts near singularities.

Classical Shannon-Nyquist sampling theory (see, for example, [68]) describes band-limited signals as a linear combination of sinc-functions. This signal model requires at least two samples per period corresponding to the highest frequency present in the signal. In practice, the data must be oversampled to attenuate the impact of noise and allow for local interpolation. The Shannon-Nyquist model may be generalized by replacing the sinc-functions by an alternative basis set, e.g., splines or wavelets [68]. However, in all such models the sampling requirements are directly related to the highest frequency present in the signal.

For band-limited periodic functions (i.e., trigonometric polynomials of degree less or equal to N), it

is well-known that they may be recovered from their samples via

$$f(x) = \sum_{l=0}^{2N} f\left(\frac{l}{2N+1}\right) D_N\left(x - \frac{l}{2N+1}\right), \quad x \in [0,1), \tag{B.1}$$

where

$$D_N(x) = \frac{1}{2N+1} \sum_{|l| \le N} e^{2\pi i l x} = \frac{1}{2N+1} \frac{\sin(2N+1)\pi x}{\sin \pi x}$$

is the Dirichlet kernel.

In our signal model, a real-valued periodic function is represented by a rational function with $4M+1$ real parameters,

$$g(x) = a_0 + 2\mathrm{Re} \sum_{m=1}^{M} \frac{w_m}{e^{-2\pi i x + \eta_m} - 1}, \quad x \in [0,1) \tag{B.2}$$

where $w_m \in \mathbb{C}$, $\eta_m \in \mathbb{C}$, $\mathrm{Re}(\eta_m) > 0$, and $a_0 \in \mathbb{R}$ is a constant. The frequency content of this rational function does not depend on the number of terms M but, rather, on the proximity of nodes $e^{-\eta_m}$, $m = 1, \ldots, M$ to the unit circle (the use of the term "nodes" will become clear later). This is fundamentally different from the signal model (B.1), where the sampling rate is controlled by the degree N, i.e., the number of terms in the model. In other words, we have a sparse representation for functions with isolated singularities as described in [13].

One of the goals of this paper is to present a robust reconstruction algorithm for the signal model (B.2) in the presence of noise. Our algorithms are based on finding the Fourier domain representation of $\widehat{g}$ via decaying exponentials [13]. Using this representation, we augment the measured data, i.e., double the number of available samples in each projection or, equivalently, extend (double) the domain of their Fourier transform. We note that our approach to improving resolution of reconstruction near singularities differs from those in recent papers, see for example [31, 20, 19, 29, 30, 39, 37, 50].

We also present a new fast numerical algorithm for inverting the Radon transform. Following [8], we construct a polar grid in the Fourier domain that allows us to mimic the Fourier slice theorem in setting up the image reconstruction. Constructing an image from such grids via the Unequally Spaced Fast Fourier transform (USFFT) [28, 6, 53] requires $\mathcal{O}(N^2 \log N)$ operations, where N is the number of projection samples and the number of projections is $\mathcal{O}(N)$.

We start in Section B.2 by a brief description of our rational model and provide background information in Section B.3. In Section B.4 we introduce a new approach for the computation of weights in the rational model of projection data. Then, in Section B.5, we describe a new, fast, Fourier domain algorithm (dubbed Polar Quadrature Inversion) for tomographic reconstruction using polar grids. Finally, in Section B.6, we perform numerical tests demonstrating the improved resolution resulting from the use of nonlinear approximations for the projections. We also compare the performance of the new fast algorithm and the standard FBP algorithm.

B.2 A rational model for signals

Let us describe analytic relations between samples of g and the parameters of the rational model (B.2). For this purpose, we derive a relation between the Fourier series coefficients of g and the Discrete Fourier transform (DFT) of its samples $g_n = g(n/N)$, $n = 0, \ldots, N-1$, where $N \geq 4M+1$, the total number of parameters in (B.2).

The coefficients $\{a_k\}_{k\in\mathbb{Z}}$ of the Fourier series of g in (B.2) are readily available,

$$a_k = \int_0^1 g(x)\, e^{-2\pi i k x} dx = \sum_{m=1}^{M} w_m e^{-\eta_m k}, \;\; a_{-k} = \overline{a}_k \;\; k \in \mathbb{N}, \tag{B.3}$$

and

$$a_0 = \int_0^1 g(x)\, dx,$$

which coincides with the constant term in (B.2), i.e., the proper rational terms in (B.2) do not contribute to a_0. We emphasize that all Fourier coefficients are fully described by $4M+1$ real parameters whereas the frequency content of g is controlled by the distance of the nodes, $e^{-\eta_m}$, $m = 1, \ldots, M$, from the unit circle, i.e., the real part of the exponents, $\mathrm{Re}(\eta_m)$.

In order to compute the DFT of g_n, we substitute the representation (B.3) into the Fourier series of

g. For any integer $N > 0$, summing the geometric series, we have

$$\begin{aligned} g(x) - a_0 &= \sum_{k\geq 1} \left(a_k e^{2\pi i k x} + \overline{a}_k e^{-2\pi i k x} \right) \\ &= 2\mathcal{R}e \sum_{n\geq 0} \sum_{j=0}^{N-1} \sum_{m=1}^{M} w_m e^{-\eta_m (j+Nn)} e^{2\pi i (j+Nn) x} - 2\mathcal{R}e \sum_{m=1}^{M} w_m \\ &= 2\mathcal{R}e \sum_{j=0}^{N-1} \sum_{m=1}^{M} \frac{w_m}{1 - e^{-\eta_m N} e^{2\pi i N x}} e^{-\eta_m j} e^{2\pi i j x} - 2\mathcal{R}e \sum_{m=1}^{M} w_m. \end{aligned} \tag{B.4}$$

Sampling $g(x)$, we obtain

$$g\left(\frac{n}{N}\right) = a_0 - 2\mathcal{R}e \sum_{m=1}^{M} w_m + 2\mathcal{R}e \sum_{j=0}^{N-1} \sum_{m=1}^{M} \frac{w_m}{1 - e^{-\eta_m N}} e^{-\eta_m j} e^{2\pi i j n/N}, \tag{B.5}$$

where $n = 0, \ldots, N-1$. Computing the DFT of $g\left(\frac{n}{N}\right)$, $n = 0, \ldots, N-1$,

$$\hat{g}_j = \frac{1}{N} \sum_{n=0}^{N-1} g\left(\frac{n}{N}\right) e^{-2\pi i n j/N}, \quad j = 0, \ldots, N-1,$$

we obtain

$$\hat{g}_j = \sum_{m=1}^{M} \frac{w_m}{1 - e^{-\eta_m N}} e^{-\eta_m j} + \sum_{m=1}^{M} \frac{\overline{w}_m}{1 - e^{-\overline{\eta}_m N}} e^{-\overline{\eta}_m (N-j)}, \quad j = 1, \ldots, N-1, \tag{B.6}$$

and

$$\widehat{g}_0 = a_0 - 2\mathcal{R}e \sum_{m=1}^{M} w_m + 2\mathcal{R}e \sum_{m=1}^{M} \frac{w_m}{1 - e^{-\eta_m N}} = a_0 + 2\mathcal{R}e \sum_{m=1}^{M} w_m \left(\frac{1}{1 - e^{-\eta_m N}} - 1 \right).$$

Note that the DFT coefficients and the Fourier coefficients (B.3) of g share the same nodes $e^{-\eta_m}$, $m = 1, \ldots, M$. On the other hand, the DFT coefficients also contain (what we call) companion nodes $e^{\overline{\eta}_m}$, $m = 1, \ldots, M$ which lie outside the unit disk. These companion nodes appear due to aliasing, i.e., folding of Fourier series coefficients corresponding to high frequencies onto low frequencies. We note that if the sampling rate is sufficient, then the DFT coefficients $\hat{g}_j$, $j = 1, \ldots, N/2 - 1$ accurately approximate the Fourier series coefficients with the same indices and the companion nodes may be ignored. In our case, due to presence of singularities, we have to deal with the potential influence of the companion nodes. The algorithm for finding nodes given the DFT coefficients is described in Section B.3.3 and the details of computing weights are described in Section B.4.

In the presence of noise, recovering the $4M + 1$ real parameters in (B.2) requires appropriate oversampling of g. However, the required sampling rate is typically less than the rate dictated by the Nyquist criterion.

Finally, let us emphasize that the exponential representation of the Fourier coefficients in (B.3) defines the rational spatial representation (B.2) and vice versa [13]. We use this correspondence throughout the paper.

B.3 Preliminary considerations

B.3.1 Tomographic reconstruction problem

We consider the classical problem of X-ray tomography, the inversion of the Radon transform. Given a function u defined on the plane, the Radon transform is defined as integral over the lines $\{\mathbf{x} : \ s = \boldsymbol{\nu} \cdot \mathbf{x}\}$ parametrized by $s \in [-1, 1]$ and $\boldsymbol{\nu} = (\cos\theta, \sin\theta)$ a unit vector,

$$v(s, \boldsymbol{\nu}) = (Ru)(s, \boldsymbol{\nu}) = (R_\theta u)(s) = \int_{\mathbb{R}^2} u(\mathbf{x}) \delta(s - \boldsymbol{\nu} \cdot \mathbf{x}) \, d\mathbf{x}. \tag{B.7}$$

Note that the points $(s, \boldsymbol{\nu})$ and $(-s, -\boldsymbol{\nu})$ define the same line. The inversion of the Radon transform may be accomplished via the FBP algorithm which may be written in operator form,

$$I = R^* K R,$$

where the back-projection operator (the dual transform) is defined as

$$(R^* v)(\mathbf{x}) = \int_{\|\boldsymbol{\nu}\|=1} v(s, \boldsymbol{\nu}) \mid_{s=\boldsymbol{\nu}\cdot\mathbf{x}} d\boldsymbol{\nu}$$

and the convolution operator (filter) as

$$(Kf)(s) = \frac{1}{2} \int_{\mathbb{R}} |\rho| \, \hat{f}(\rho) \, e^{2\pi i \rho s} d\rho.$$

We also have from (B.7) the Fourier slice theorem,

$$(Fu)(\rho\boldsymbol{\nu}) = \int_{\mathbb{R}^2} u(\mathbf{x}) e^{-2\pi i \rho \boldsymbol{\nu}\cdot\mathbf{x}} d\mathbf{x} = \int_{\mathbb{R}} (Ru)(s, \boldsymbol{\nu}) \, e^{-2\pi i \rho s} ds,$$

which we use to build a fast and accurate Fourier domain reconstruction algorithm as an alternative to FBP. In this paper we only examine the so-called parallel beam tomography, i.e. the most simple geometry for the tomographic reconstruction problem. However, the methods used in this paper are applicable to other, more complex geometries, such as fan beam tomography.

B.3.2 Quadratures for the disk

In order to have an accurate and fast reconstruction algorithm in the Fourier domain, we construct, for any user-specified accuracy, quadratures for integration in the Fourier domain. Following [8, 11], we discretize integrals using polar grids in a disk in the Fourier domain using quadratures for band-limited exponentials. In the radial variable these quadratures integrate exponentials against the weight $|\rho|$. In the angular variable we integrate using the trapezoidal rule.

Let us first examine the behavior the Fourier transform (in polar coordinates) of a real function supported within the box $B = [-1, 1] \times [-1, 1]$,

$$\hat{f}(c\rho, \theta) = \int_B f(\mathbf{x}) e^{-ic\rho \boldsymbol{\nu}\cdot\mathbf{x}} d\mathbf{x} = \int_0^{2\pi} \int_0^{\sqrt{2}} f(r, \varphi) e^{-icr\rho\cos(\theta-\varphi)} r dr\, d\varphi,$$

where $\boldsymbol{\nu} = (\cos\theta, \sin\theta)$ and, for notational convenience, the integration domain is extended to a disk (in space) containing B. The size of the disk in the Fourier domain is controlled by the bandlimit parameter c and we assume that the function $\hat{f}(c\rho, \theta)$ is negligible for $\rho > 1$. Under this assumption let us show that, for $0 \le \rho \le 1$, the Fourier series

$$\hat{f}\,(c\rho, \theta) = \sum_{l\in\mathbb{Z}} q_l\,(c\rho)\, e^{i\theta l},$$

may be truncated, i.e., the coefficients $q_l\,(c\rho)$ are negligible for $|l| > L$, for some L. Computing

$$\begin{aligned} q_l(c\rho) &= \frac{1}{2\pi}\int_0^{2\pi} \hat{f}(c\rho, \theta) e^{il\theta} d\theta \\ &= \int_0^{2\pi}\int_0^{\sqrt{2}} f(r, \varphi) \left(\frac{1}{2\pi}\int_0^{2\pi} e^{-icr\rho\cos(\theta-\varphi)} e^{il\theta} d\theta \right) r dr\, d\varphi, \end{aligned} \tag{B.8}$$

we obtain

$$q_l(c\rho) = (-i)^l \int_0^{2\pi}\int_0^{\sqrt{2}} f(r, \varphi) J_l(cr\rho) e^{il\varphi} r dr\, d\varphi, \quad l \in \mathbb{Z}, \tag{B.9}$$

and hence

$$|q_l(c\rho)| \le 2\pi \|f\|_\infty \max_{x\in[0,\sqrt{2}c]} |J_l(x)|\,.$$

Using [1, 9.1.62] and [1, 6.1.38], for $x > 0$ and $l \ge 7$, we have

$$|J_l(x)| \le \frac{1}{l!}\left(\frac{x}{2}\right)^l \le \frac{1}{2\pi}\left(\frac{ex}{2l}\right)^l.$$

Hence, with $l \geq e\sqrt{2}c$, we obtain

$$|q_l(c\rho)| \leq \|f\|_\infty 2^{-l},$$

and, for some L,

$$L \geq \max\left\{7, e\sqrt{2}c, \log_2\left(2\|f\|_\infty \epsilon^{-1}\right)\right\}$$

we arrive at

$$\left|\sum_{|l|>L} q_l\left(c\rho\right) e^{i\theta l}\right| \leq 2\|f\|_\infty \sum_{l=L+1}^{\infty} 2^{-l} = \|f\|_\infty 2^{-L+1} < \epsilon.$$

Using this estimate for the Fourier series of $\widehat{f}$, we obtain

$$\left|\hat{f}\left(c\rho,\theta\right) - \sum_{l=-L}^{L} q_l\left(c\rho\right) e^{i\theta l}\right| \leq \epsilon. \tag{B.10}$$

A similar conclusion was made in [56] in the context of sampling for tomographic reconstruction algorithms.

Since the function f is real and $(\rho, \theta+\pi)$ and $(-\rho,\theta)$ describe the same point in a disk, we extend $\widehat{f}\left(c\rho,\theta\right)$ for negative ρ as

$$\widehat{f}\left(-c\rho,\theta\right) = \widehat{f}\left(c\rho,\theta+\pi\right) = \overline{\widehat{f}\left(c\rho,\theta\right)},$$

so that the coefficients $q_l\left(c\rho\right)$ in (B.10) for negative ρ are extended as

$$q_l\left(-c\rho\right) = \overline{q_{-l}\left(c\rho\right)}.$$

As demonstrated, the number of angular modes to represent $\widehat{f}$ is proportional to c, so that we can use polar grid quadratures in [8] to discretize the inverse Fourier transform of $\widehat{f}$ within the unit disk $D = \{\mathbf{p} \mid \|\mathbf{p}\| \leq 1\}$,

$$f\left(\mathbf{x}\right) = c^2 \int_D \hat{f}(c\mathbf{p}) e^{ic\mathbf{p}\cdot\mathbf{x}} d\mathbf{p} = \frac{c^2}{2} \int_{-\pi}^{\pi} \int_{-1}^{1} \hat{f}\left(c\rho,\theta\right) e^{ic\rho(x_1\cos\theta + x_2\sin\theta)} \left|\rho\right| d\rho d\theta, \tag{B.11}$$

where $\mathbf{x} = (x_1, x_2)$, $x_1, x_2 \in [-1,1]$. We obtain

$$\left|f\left(\mathbf{x}\right) - \frac{\pi c^2}{N_\theta} \sum_{\nu=-N_\rho}^{N_\rho} \sum_{k=0}^{N_\theta-1} w_\nu \hat{f}\left(c\rho_\nu,\theta_k\right) e^{ic\rho_\nu(x_1\cos\theta_k + x_2\sin\theta_k)}\right| \leq \epsilon, \tag{B.12}$$

where ρ_ν, $|\rho_\nu| < 1$, are (unequally spaced) quadrature nodes on the diameters, w_ν are corresponding quadrature weights, and $\theta_k = 2\pi k/N_\theta$. The choice of quadratures on the diameters (and the number of nodes) depends on the bandlimit c and the desired accuracy ϵ. The number of diameters, N_θ, is proportional to c.

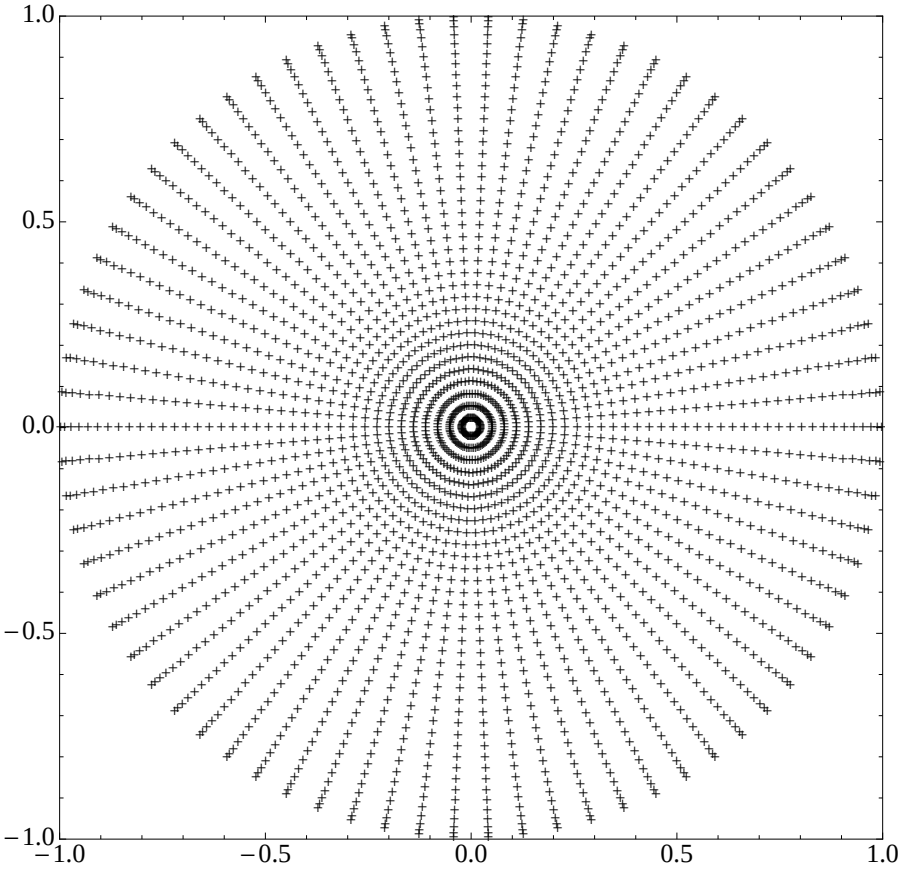


Figure B.1: A polar grid for integration in the Fourier domain where the number of nodes per diameter is 74 and the number of diameters is 37. This grid (along with appropriate weights) is designed to yield an accuracy of $1.68 \cdot 10^{-7}$ for computing the inverse Fourier transform of functions with bandlimit 31.75π (i.e., supported within the disk) .

An example of such a grid is illustrated in Figure B.1. The sum in (B.12) is evaluated on a $N_{image} \times N_{image}$ cartesian grid using USFFT at the cost of $\mathcal{O}\left(c^2 \log c\right)$ since N_{image}, N_ρ and N_θ are all proportional to c.

Remark 1 The construction in [8] is based on Slepian-type functions for the mapping of a square in space to a disk in the Fourier domain. This differs from the original construction in [65] of disk-to-disk mapping since, in our case, variables do not separate and no ordinary differential equation is available to define such functions. On the diameters we use quadratures constructed in [11] for the weight $|\rho|$. Unlike quadratures for polynomials, the nodes of quadratures for band-limited exponentials do not concentrate excessively towards the end points of diameters. As a result, even for large bandlimits, we avoid clustering of nodes near such points [15]. We believe that in the tomography setup, the Slepian functions, i.e., the eigenfunctions of the square-to-disk space-limiting and band-limiting operator, provide a class of functions that naturally describe both, the measured data and the reconstructed image. The quadratures described above yield a numerical tool for computing within well-localized functions of this class.

B.3.3 Methods of approximation via exponentials

For each real-valued, sampled projection, we compute its DFT and approximate positive frequencies via a linear combination of exponentials with (near) minimal number of terms for a user-selected accuracy. For this purpose, we use the algorithm in [12, 14], the key steps of which we present below.

Given a user-selected accuracy $\epsilon > 0$ and $2L+1$ equally spaced samples of a complex-valued function $h(c\xi)$, $\xi \in [0,1]$, the algorithm yields an approximation

$$\left| h(c\frac{l}{2L}) - \sum_{m=1}^{M} w_m e^{-c\eta_m l} \right| < \epsilon, \;\; \mathcal{R}\mathrm{e}\,(\eta_m) > 0, \tag{B.13}$$

for $0 \le l \le 2L$, where the number of terms, M, is near minimal for the choice of c and ϵ. In this formulation, the constant $c > 0$ scales the problem to the interval $[0,1]$.

Provided that $h(c\xi)$ is sufficiently sampled, by defining $\tau_m = 2L\eta_m$, and replacing $l/(2L)$ with a continuous variable ξ in (B.13), we obtain

$$\left| h(c\xi) - \sum_{m=1}^{M} w_m e^{-c\tau_m \xi} \right| < \epsilon', \tag{B.14}$$

for $\xi \in [0,1]$. The new error ϵ' is only slightly worse than ϵ.

Algorithm 3.1 [14, Appendix A.1]

The algorithm to produce the approximation (B.13) has the following steps:

- Construct the $(L+1) \times (L+1)$ Hankel matrix $\mathbf{H}_{ll'} = h_{l+l'}$, where $h_{l+l'} = h(c\frac{l+l'}{2L})$, $0 \le l, l' \le L$.
- Find a vector $\mathbf{u}$ satisfying $\mathbf{Hu} = \sigma\overline{\mathbf{u}}$ with positive σ close to ϵ. A solution is guaranteed by Tagaki's factorization [44] and may be reduced to finding the Singular Value Decomposition (SVD) of $\mathbf{H}$. Given singular values $\sigma_0 \ge \sigma_1 \ge \cdots \ge \sigma_M \ge \cdots \ge \sigma_L$, we choose M such that $\epsilon \approx \sigma_M/\sigma_0$ and the corresponding singular vector $\mathbf{u} = \{u_l\}_{l=0}^{L}$.
- Compute the roots γ_m of the polynomial

$$u(z) = \sum_{l=0}^{L} u_l z^l. \tag{B.15}$$

- The exponents η_m in equation (B.13) are defined by the roots γ_m via $\eta_m = -\log(\gamma_m)/c$, where log is the principal value of the logarithm. We also compute $\tau_m = 2L\eta_m$.

For our purpose, we only need the roots γ_m of (B.15) located inside the unit disk yielding only decaying exponentials in (B.13).

The usual last step of this algorithm is to form a linear Vandermonde system

$$h\left(\frac{cl}{2L}\right) = \sum_{m=1}^{M} w_m \gamma_m^l \tag{B.16}$$

and solve for the weights w_m as the best fit in the ℓ^2 sense. However, due to the nature of problems in tomography, in the next section we depart from the conventional approach and calculate weights differently.

Remark 2 An important feature of this algorithm is that it detects the level of noise in the data. Specifically, the rate of decay of the singular values of the matrix $\mathbf{H}$ changes significantly once the ratio σ_m/σ_0 reaches the level of noise in the signal. Indeed, unlike a coherent signal, noise does not have an efficient representation via exponentials. For this reason, the gain in accuracy of fitting the data becomes negligible as we add additional terms. This directly affects the rate of decay of singular values, which we use as a tool for the

selection of the singular value, σ_M, and hence, the number of terms M in the representation. For more details see [13].

Remark 3 We want to point out alternatives to Algorithm 3.1 for finding (near) optimal approximations via linear combinations of exponentials. For this purpose, we may use the HSVD [49] or the (equivalent) matrix pencil method [45, 46, 47]. The advantage of such methods is that we avoid the step of finding roots of the polynomial (B.15). On the other hand, Algorithm 3.1 is faster (if properly implemented) since we need to find only a single singular vector of a structured (Hankel) matrix. The alternative methods mentioned above require computing all singular vectors up to the index corresponding to that of the accuracy threshold.

B.4 Calculating weights for tomography problems

Following Algorithm 3.1, for each projection, we construct the Hankel matrix $\mathbf{H}$ using its DFT coefficients (B.6) with indices corresponding to positive frequencies. Since construction of the Hankel matrix requires the total number of samples to be odd, if necessary, we drop the largest positive frequency, so that

$$\mathbf{H}_{ll'} = \widehat{g}_{l+l'+1}, \;\; l, l' = 0, \ldots, \widetilde{N},$$

where $\widetilde{N} = \lfloor N/4 + 1/2 \rfloor - 1$ and $\lfloor \, \rfloor$ denotes the floor function. Using this Hankel matrix $\mathbf{H}$, we compute the appropriate singular vector and find the roots γ_m of the polynomial as described in previous section. However, instead of using (B.16), we develop a new algorithm to find the weights for reasons we explain next.

One of the problems we encountered using Algorithm 3.1 is that some roots γ_m may end up just outside the unit disk. Since such roots (typically with small weights) would prevent us from extrapolating the Fourier data beyond the original bandlimit, we discard them at the expense of introducing an additional error. The following approach aims to isolate such errors to the vicinity of singularities (responsible for a slow decay of the Fourier coefficients and causing this effect). Since Algorithm 3.1 produces a near optimal number of terms (without any reference to the physics of the problem), it may sometimes introduce a growing exponential with a small weight. As mentioned at the end of Section B.2, the exponential representation of

the DFT coefficients (B.6) differs from that of the Fourier coefficients (B.3) in that it has a second sum,

$$\sum_{m=1}^{M} \frac{\overline{w}_m}{1-e^{-\overline{\eta}_m N}} e^{-\overline{\eta}_m (N-j)}, \quad \mathcal{R}\mathrm{e}\,(\eta_m) > 0. \tag{B.17}$$

Since, in the construction of the Hankel matrix $\mathbf{H}$, we use only the DFT coefficients with indices corresponding to positive frequencies, we have $j \leq N/2-1$ in (B.17). Consequently, the sum in (B.17) typically has a small contribution within this index range, although, as a function of j, it contains growing terms, $e^{\overline{\eta}_m j}$. Since the form (B.13) does not account for these terms explicitly, their contribution may result in roots γ_m just outside the unit disk. Also, such an effect may be caused by the presence of noise.

The proximity of a node to the unit circle controls the frequency contribution of that node while the position of the corresponding pole (of the rational approximation) is directly associated with the location of a singularity [13]. Therefore, the impact of removing nodes just outside the unit disk should be localized to neighborhoods of singularities. However, when we remove nodes from just outside the unit circle and use the ℓ^2 norm to calculate the weights via (B.16), the error spreads out to a large neighborhood of the singularity as illustrated in Figures B.2 (a) and (c), and Figure B.3 (a). In order to remedy this situation, we calculate weights by minimizing the ℓ^1 norm of the the residual with respect to the original spatial data. We choose to minimize the ℓ^1 norm due to its well known sparsity properties (see, e.g., [63]). The effect of using the ℓ^1 norm to calculate weights is illustrated in Figures B.2 (b) and (d), and Figure B.3 (b).

Let us describe the details. Once we obtain the exponent η_m via Algorithm 3.1 and select all those with $\mathcal{R}e\,(\eta_m) > 0$, we proceed to compute weights in the space domain. Specifically, using the fact that the function is real we extend the approximation of positive frequencies to negative frequencies and analytically sum the Fourier series to obtain a rational function of the form (B.2). Denoting the grid at which we measure the projection data as $\{x_j\}_{j=0}^{N-1}$, we discretize (B.2) and obtain a Cauchy-like system to solve for the weights w_m.

$$g_n = g\left(\frac{n}{N}\right) = \widehat{g}_0 + 2\mathrm{Re}\sum_{m=1}^{M} w_m \left(\frac{1}{e^{-2\pi i n/N+\eta_m}-1} - \frac{1}{1-e^{-\eta_m N}} + 1\right), \tag{B.18}$$

where $n = 0, \ldots, N-1$. Let us denote the $N \times 2M$ matrix of this system as $\mathbf{C}$ and the right hand side $\{g_n - \widehat{g}_0\}_{n=0}^{N-1}$ as $\mathbf{g}$. We note that the contribution of terms $1/\left(1-e^{-\eta_m N}\right) - 1$ is minor since $e^{-\eta_m N}$ is typically small.

We first solve

$$\mathbf{Cw} = \mathbf{g}, \quad \text{with} \quad \text{argmin}_{\mathbf{w}} \left\| \mathbf{Cw} - \mathbf{g} \right\|_2 , \tag{B.19}$$

where $\mathbf{g}$ is the vector containing the (shifted) data samples. We then verify if the residual is within the error tolerance. If it is not (which is the case for only a few projections), we then solve

$$\mathbf{Cw} = \mathbf{g}, \quad \text{with} \quad \text{argmin}_{\mathbf{w}} \left\| \mathbf{Cw} - \mathbf{g} \right\|_1 . \tag{B.20}$$

In this approach we rely on an observation that if solving (B.19) satisfies the error tolerance, then the difference in using ℓ^1 or ℓ^2 norms is insignificant.

For solving (B.19) we use the SVD while for (B.20) we use convex optimization. For the latter, since $\mathbf{C}$ is complex valued, we solve via a second order cone program implemented in CVX [36]. An alternative approach for solving (B.20) relies on the iteratively re-weighted least squares (IRLS) algorithm (see, e.g. [63]).

We illustrate the difference between using the ℓ^2 norm vs the ℓ^1 norm in an example. In Figure B.2 (a), using the ℓ^2 norm, we show a clearly visible artifact near one of the sharp transitions in the projection. This should be compared with Figure B.2 (b) where, using (B.20), the error is significantly reduced away from the sharp transition. The comparison of the errors for the entire projection is presented in Figures B.3 (a) and (b).

Remark 4 If we were to form the $N/2 \times N/2$ Hankel matrix $\mathbf{H}$ using all $\widehat{g}_j$, $j = 1, \ldots N-1$ (N is even) and, thus, insist on the form (B.6), then the resulting matrix is equivalent to a self-adjoint Toeplitz matrix. This may be seen by multiplying $\mathbf{H}$ by the matrix $\mathbf{J}$ with entries $\left\{\delta_{i,N/2-j-1}\right\}_{i,j=0,\ldots,N/2-1}$. This leads to a construction similar to that in [11] and, in this case the roots (except those on the unit circle) indeed come in pairs, $\gamma_m^{out} = 1/\overline{\gamma_m^{in}}$. However, in this case some nodes may lie on the unit circle which creates problems with the extension. An approach to deal with such nodes may provide an alternative to the method of this paper.

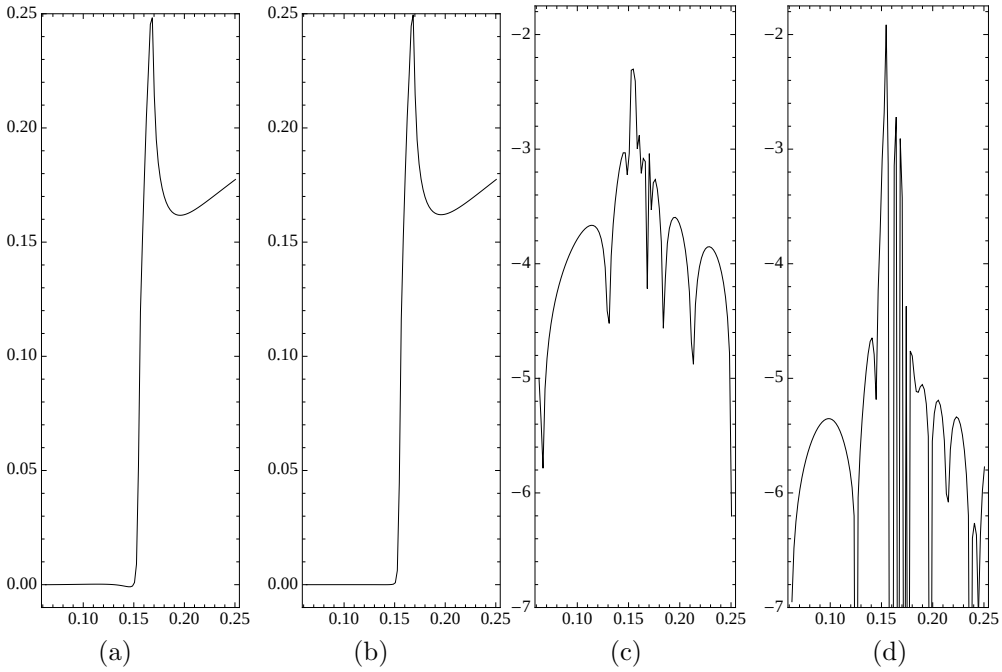


Figure B.2: Rational approximation of a projection in the vicinity of a sharp transition. The result of using weights obtained by ℓ^2 minimization of the residual (a) and ℓ^1 minimization of the residual (b). The corresponding errors are illustrated in (c) and (d). We observe a significant improvement in error localization using the ℓ^1 norm.

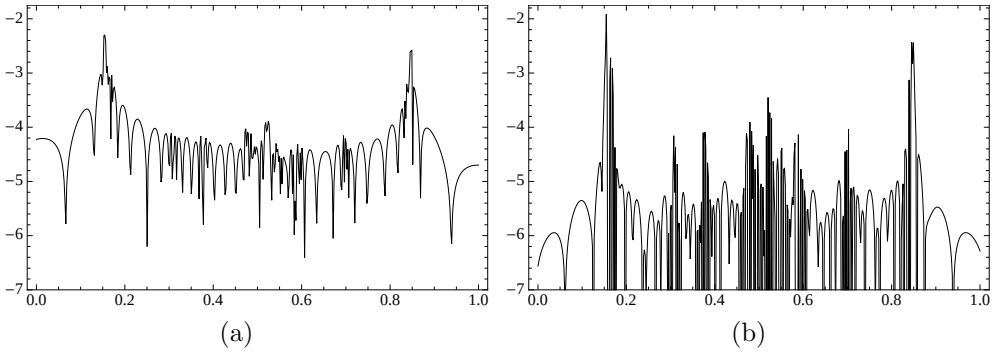


Figure B.3: Errors in rational approximation of the entire projection in Figure B.2 using weights obtained by ℓ^2 minimization of the residual (a) and ℓ^1 minimization of the residual (b). We observe a consistent improvement of the error away from sharp transitions.

B.5 Polar Quadrature Inversion

Current numerical implementations of Fourier methods for inverting the Radon transform, the so-called gridding methods, use interpolation in the Fourier domain to generate values on a rectangular (typically square) equally spaced grid so that inversion can then proceed by using the FFT [56, Section 5.2]. Instead, our new algorithm uses a carefully designed polar grid described in Section B.3.2 and, in this sense, is consistent with the setup of the projection slice theorem. This grid allows us to accurately discretize the Fourier domain integral (B.11) to represent the image via the sum in (B.12). We evaluate this sum via USFFT [28, 6, 53]. Note that the only inputs into this algorithm are the values of the Fourier transform at the grid points. Finding these values is fairly straightforward due to the functional form of our approximation of projections.

We denote projections as $g_n(k)$, $n = 0, \ldots, N-1$ and $k = 0, \ldots, N_\theta - 1$, where index k identifies a projection corresponding to the angle $\theta_k = \pi k / N_\theta$, and index l identifies a location within a projection. For each projection k we compute the DFT,

$$\widehat{g}_j(k) = \sum_{n=0}^{N-1} g_n(k) e^{-2\pi i j n / N},$$

yielding both positive and negative discrete frequencies. We restrict our approximation to the maximum possible (odd) number of positive frequencies, denoted in the sequel as $\widetilde{N}$. For each projection k, we use Algorithm 3.1 to construct (near) optimal approximations of the values $\widehat{g}_j(k)$, $j = 1, \ldots, \widetilde{N}$,

$$\left| \widehat{g}_j(k) - \sum_{m=1}^{M_k} w_{mk} e^{-c\eta_{mk} j} \right| \leq \epsilon, \quad j = 1, \ldots, \widetilde{N}, \quad \mathcal{R}e\,(\eta_{mk}) > 0, \tag{B.21}$$

where $\epsilon > 0$ is the user-supplied accuracy and the weights w_{mk} are calculated using either (B.19) or (B.20), whichever is appropriate (see considerations in Section B.4). For each projection k, we then construct a representation (B.21) using Algorithm 3.1 yielding a functional form of the Fourier transform of the projection,

$$\widehat{g}^{ext}(k, \rho) = \sum_{m=1}^{M} w_{mk} e^{-c\tau_{mk}\rho}, \quad \rho \geq 0, \quad \mathcal{R}e\,(\tau_{mk}) > 0, \tag{B.22}$$

where $\tau_{mk} = \widetilde{N}\eta_{mk}$. While (B.22) corresponds to measured data for $\rho \in [0, 1]$, this functional form allows us to extrapolate this expression to values $\rho > 1$. Since we consider an image to be a function with at most

jump discontinuities, its Fourier transform has a slow rate of decay in radial directions, not consistent with the exponential decay of (B.22). For this reason, we restrict extrapolation to a finite radius where it provides a reasonable approximation. We have chosen to evaluate (B.22) in $\rho \in [0, 2]$, effectively defining the new bandlimit as $c_{new} = 2c$. This typically results in the Fourier values of sufficiently large magnitude near the boundary of the disk of radius c_{new}. To ensure sufficient decay towards the boundary, we apply a radial window $W(\rho)$ in the Fourier domain. We note that without extrapolation via (B.22) a window has to be applied within the disk of radius c, thus affecting much more significantly the wave numbers corresponding to measured data. We obtain

$$\widehat{g}^{bl}(k, \rho) = \widehat{g}^{ext}(k, \rho) W(\rho), \tag{B.23}$$

where we specify our choice of the window later. Since the image is a real valued function, we define

$$\widehat{g}^{bl}(k, -\rho) = \overline{\widehat{g}^{bl}(k, \rho)} \tag{B.24}$$

for each projection, $k = 0, \ldots, N_\theta - 1$ and then sample $\widehat{g}^{bl}(k, \rho)$ on the diameter using quadrature nodes $\rho_\nu,\ \nu = -N_\rho, \ldots, N_\rho$ described in Section (B.3.2) (we may also choose a quadrature with an even number of nodes on the diameters).

The final step of the algorithm uses the USFFT [28, 6, 53] with input values $\widehat{g}^{bl}(k, \rho_\nu)$, $\nu = -N_\rho, \ldots, N_\rho$ and $k = 0, \ldots, N_\theta - 1$ to produce an image. This step (effectively) delegates interpolation in the Fourier domain to the USFFT (which attains any finite user-specified accuracy). We now summarize the Polar Quadrature Inversion (PQI) algorithm,

Algorithm 5.1

(1) Compute the FFT of the projection data $\{g_n(k)\}_{n=0}^{N-1}$ for each $k = 0, \ldots, N_\theta - 1$.

(2) Using DFT coefficients $\widehat{g}_j(k)$, $j = 1, \ldots, 2\widetilde{N} + 1$ of each projection k, construct its exponential approximation via Algorithm 3.1. The accuracy $\epsilon > 0$ may be estimated by examining the change in the rate of decay of the singular values of the Hankel matrix.

(3) Find the weights using either (B.19) or (B.20).

(4) Evaluate the windowed approximations $\widehat{g}^{bl}(k,\rho)$ in (B.23-B.24) at the quadratures nodes ρ_ν, $\nu = -N_\rho,\ldots,N_\rho$, corresponding to the new bandlimit c_{new} (note that this step includes extrapolation) for each $k = 0,\ldots,N_\theta - 1$.

(5) Obtain the image by computing the sum in (B.12) via the USFFT.

Operation count. Step 1 of the algorithm requires $\mathcal{O}(N_\theta N \log N)$ operations. Using Algorithm 3.1 for all projections in Step 2 currently requires $\mathcal{O}\left(N_\theta N^3\right)$ but may be reduced to $\mathcal{O}\left(N_\theta N M^2\right)$, where M is the number of terms in the exponential approximation (this may be achieved using randomized projections, see the review article [40]). We also note that Step 2 is trivially parallelizable as each projection is treated separately. Step 3 currently requires $\mathcal{O}\left(N_\theta N M^2\right)$ when using (B.19) and a similar complexity to solve (B.20) but with a larger constant. However, only a few projections require using (B.20). The remaining steps require $\mathcal{O}(N^2 \log N)$ operations.

B.6 Numerical examples

We now test the impact on image reconstruction of both the nonlinear approximation of projections and the new PQI algorithm. Following tradition, we demonstrate our reconstruction technique using the Shepp-Logan phantom. To verify the effect of nonlinear approximation, we apply the standard FBP algorithm to both the original projection data and the augmented projections, where we double the number of samples via rational representations. This choice corresponds to the doubling of the bandlimit in the Fourier domain via the PQI algorithm and allows us to isolate the impact of rational approximation on image formation. Namely, using FBP on the original and augmented projections, we observe the impact of rational approximation; by using augmented projections in FBP we compare its output with that of the PQI algorithm.

The test data consist of 512 projections and 512 samples per projection resulting (via the standard FBP algorithm supplied in Matlab^{TM}) in a 512×512 reconstructed image while the output of PQI is a 1024×1024 reconstructed image. In all experiments we use the radial Hann window,

$$W(\rho) = \sin^2\left(\frac{\pi}{2}\rho - \frac{\pi}{2}\right), \quad \rho \in [0,1].$$

We perform two types of experiments: noiseless reconstruction and reconstruction after adding Gaussian noise to each projection.

B.6.1 Noiseless examples

First, we take our original noiseless data set of 512 projections with 512 samples per projection and approximate all of the projections by rational functions. For our experiments in a noiseless setting we use the threshold of $\sigma_M/\sigma_0 \leq 5 * 10^{-4}$ to find rational approximations (see Section B.3.3). Using these approximations, we increase the sampling of each projection by a factor of 2. We then take these augmented projections, with 512 projections with 1024 samples per projection, and apply the standard FBP algorithm yielding a 1024×1024 reconstructed image of the Shepp-Logan phantom. The reconstruction from augmented projections is displayed in Figure B.5 (a) and the reconstruction error is displayed in Figure B.5 (b). Comparing the output of FBP applied to augmented samples with the output of FBP applied to our original data set (512 projections and 512 samples per projection), see Figure B.4, we observe better resolution near the sharp boundaries when augmented projections are used as input. In both cases the error highlights the jump discontinuities of the phantom and contains streak artifacts typical in tomographic reconstructions.

We compare the results of this experiment (using augmented projections with 1024 samples as input for FBP) with the output of the PQI algorithm described in Section B.5 yielding a 1024×1024 reconstructed image of the Shepp-Logan phantom. Using this algorithm we double the size of the bandlimit in the Fourier domain which corresponds to the increase of sampling in each projection by a factor of two in using FBP. The result of the reconstruction using the PQI algorithm is shown in Figure B.6 (a) and the error in Figure B.6 (b). When compared with Figure B.4 we observe that the PQI algorithm achieves a significantly improved resolution without introducing additional artifacts.

This also allows us to compare the output of our algorithm with that of the FBP algorithm on augmented projections. We see no significant difference in the reconstructions Figure B.6 (a) and Figure B.5 (a). A comparison of the reconstruction errors, Figure B.6 (b) and Figure B.5 (b) also shows no significant difference.

B.6.2 Zooming on details

In order to demonstrate the increased resolution, we zoom in on two areas of the reconstruction. The locations of these areas are shown in Figure B.6 (a). In Figures B.7 and B.8 we compare images near the center of the phantom using the FBP and the PQI algorithms, respectively. Figure B.7 compares a 32×32 pixel patch (obtained from the original data set of 512 projections with 512 samples each and extracted from 512×512 pixel reconstruction via FBP) and a 64×64 pixel patch extracted from 1024×1024 reconstruction using the FBP on augmented data. Similarly, in Figure B.8, we compare the same 32×32 pixel patch with a 64×64 patch extracted from 1024×1024 reconstructed image using the PQI algorithm. A similar comparison is shown in Figures B.9 and B.10 but zooming into a different section of the Shepp Logan phantom.

We observe that higher resolution is not accompanied by any additional artifacts and results in visibly sharper images.

B.6.3 Noisy examples

To test the stability of approximation by rational functions in the presence of noise, we add (to each projection) Gaussian white noise with zero mean and standard deviation of $2.5 * 10^{-4}$. The noise level is of the same order as the smallest features captured by the projections. In all experiments involving noise we used the threshold $\sigma_M/\sigma_0 \leq 2 * 10^{-3}$ to find rational approximations (see Remark 2). We construct rational approximations for each noisy projection and increase the sampling rate by a factor of two. We then use the resulting augmented projections as input to the FBP algorithm. The reconstruction error of this experiment is displayed in Figure B.11 (b) and compared with the reconstruction error of the same experiment performed on noiseless data displayed in Figure B.11 (a). The Gaussian noise added to the projections appears as a speckle noise in the image in addition to the error of noiseless reconstruction.

Similarly, we display the reconstruction error of the PQI algorithm applied to noisy data in Figure B.12 (b) next to the error of reconstruction from the noiseless data in Figure B.12 (a). The qualitative difference between the errors in Figure B.12 (b) and Figure B.11 (b) is negligible.

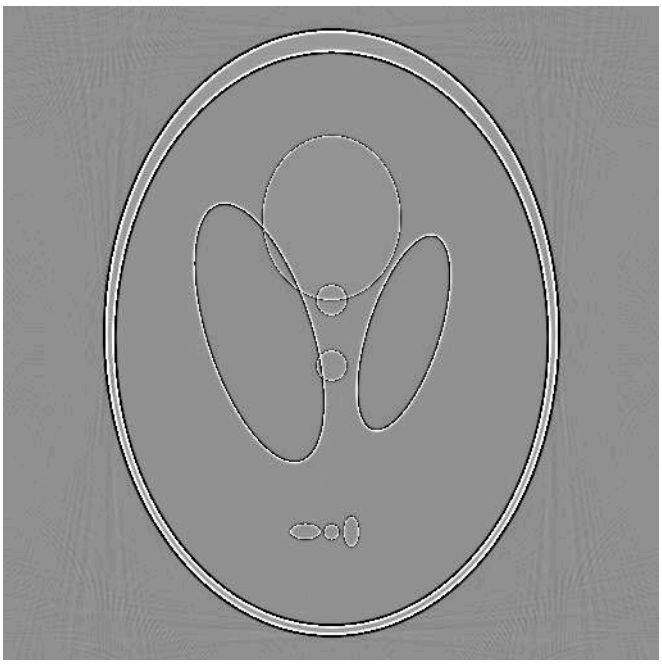
B.7 Discussion

We present an approach to improve the resolution of X-ray tomography using rational approximation of projections and the new, fast Fourier algorithm (PQI) for this purpose. We also provide a comparison between the PQI algorithm and the standard FBP.

While we use rational approximation to improve the resolution in the classical X-ray tomography setup, a similar approach may be applied in Electron Microscope Tomography and in MRI. However, in both of these modalities of non-destructive evaluation one needs to consider additional issues associated with these techniques. In particular, in Electron Microscope Tomography, one has to address a missing cone of data as well as very high noise level when compared with X-ray tomography. In MRI the collected data needs to be carefully re-sampled before using the rational model. An interesting potential application in MRI is to use the PQI algorithm to reduce the sampling requirements in the Fourier domain. We plan to address these issues in our future work.

Finally, we note that by modifying the input into the PQI algorithm, it is possible to produce the same image (within a user-supplied accuracy) as that constructed by an Algebraic Reconstruction Technique (ART), i.e., the PQI algorithm can reproduce the ART solution.

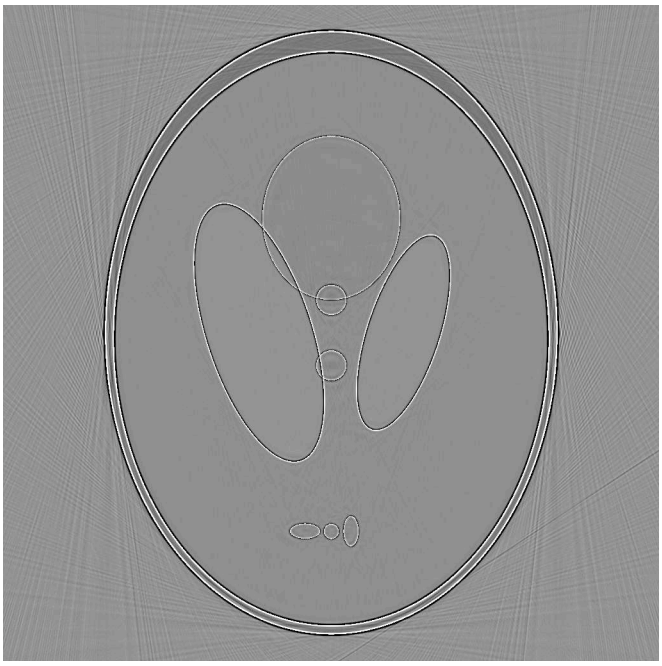
(a)



(b)

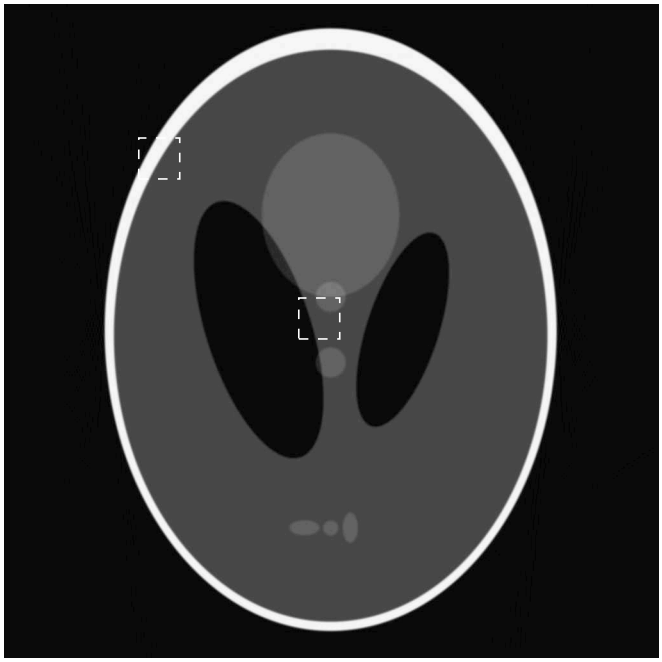
Figure B.4: A 512×512 reconstruction of the Shepp-Logan phantom using FBP algorithm (a) (the grayscale is $[-0.05, 1.05]$) and the corresponding error (b) (the grayscale is $[-.05, .05]$). A combination of filtering errors and under-sampling in angle produces streak artifacts associated with sharp transitions in the phantom.

(a)



(b)

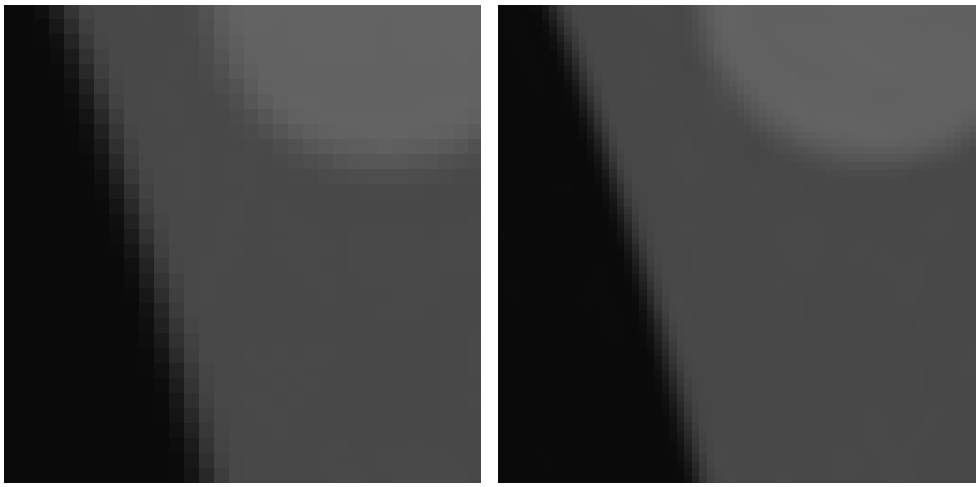
Figure B.5: A 1024×1024 reconstructed image of the Shepp-Logan phantom via the FBP algorithm using projections (with twice as many samples) generated by near optimal rational approximation (a) and the corresponding error (b). Grayscales are the same as in Figure B.4 which should be used for comparison.



(a)

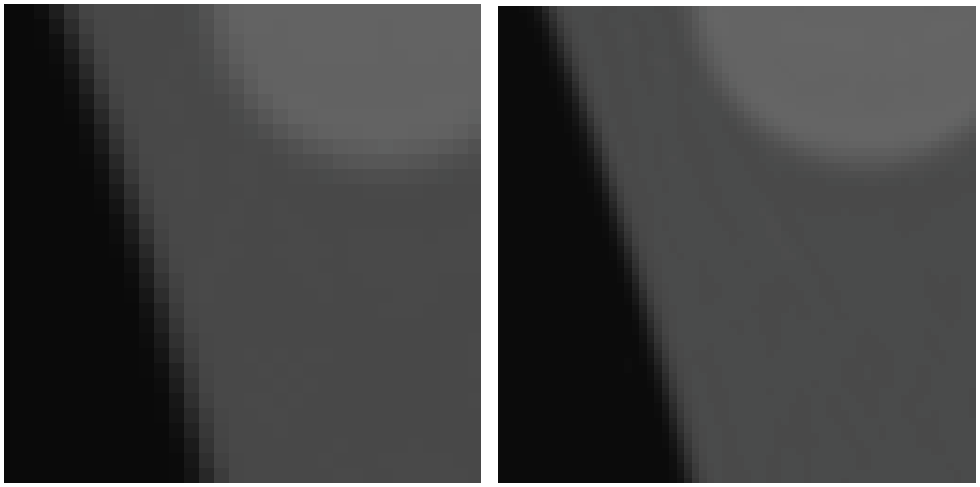
(b)

Figure B.6: A 1024×1024 reconstructed image via the PQI algorithm of Section B.5 (a) and the corresponding error (b). The grayscales are the same as in Figures B.4 and B.5, which should be used for comparison. The two boxes in (a) outline areas of the reconstructed image on which we zoom to examine the reconstruction at a pixel level.



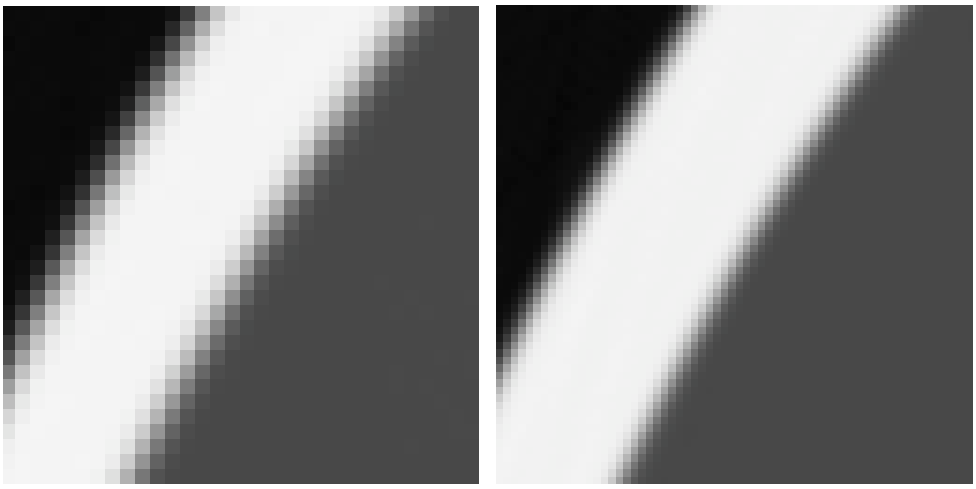
(a) (b)

Figure B.7: A zoom of a small area near the center of the phantom indicated in Figure B.6. In (a) we show a 32×32 square (out of a 512×512 reconstruction via the FBP algorithm) and in (b) the same area with 64×64 samples taken from a 1024×1024 reconstruction via the FBP algorithm using as input augmented projections.



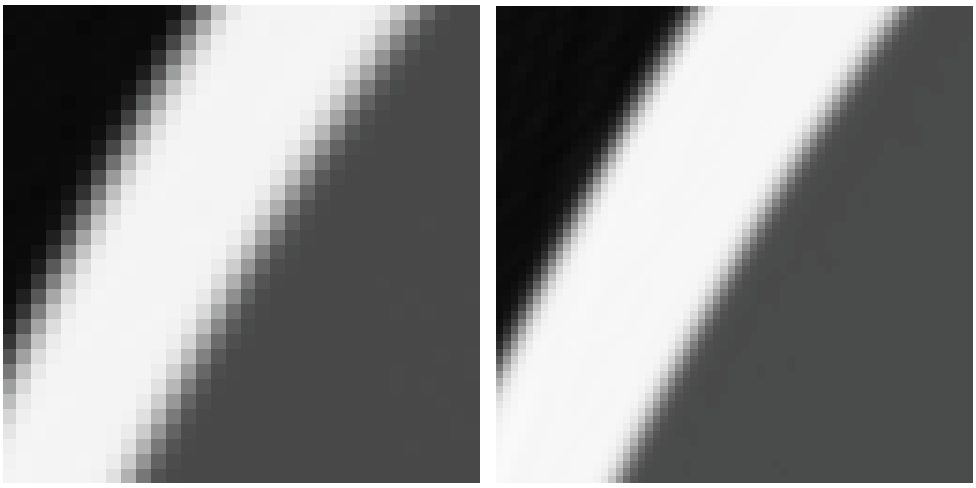
(a) (b)

Figure B.8: A different comparison for the area in Figure B.7. In (a) we show a 32×32 square (out of a 512×512 via the FBP algorithm) and in (b) the same area with 64×64 samples taken from a 1024×1024 reconstruction via the PQI algorithm.



(a) (b)

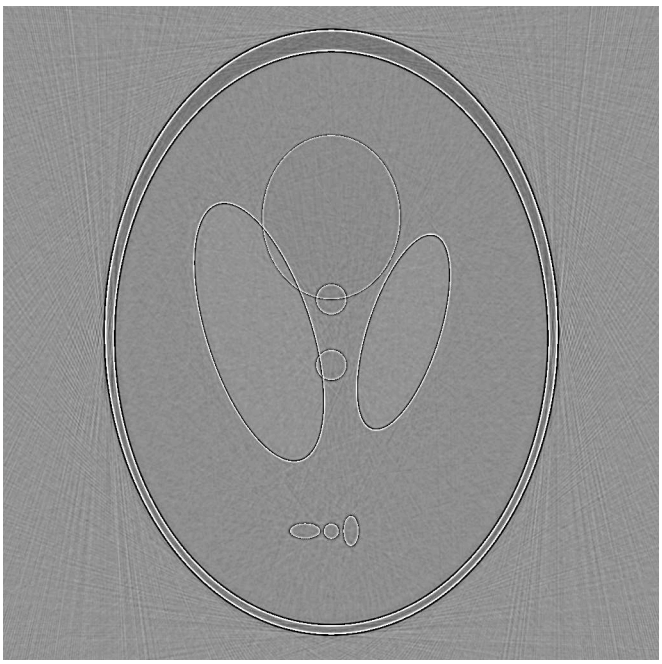
Figure B.9: A zoom of a small area near the sharpest transition in the phantom indicated in Figure B.6. In (a) we show a 32×32 square (out of a 512×512 reconstruction via the FBP algorithm) and in (b) the same area with 64×64 samples taken from a 1024×1024 reconstruction via the FBP algorithm using as input augmented projections.



(a) (b)

Figure B.10: A different comparison for the area in Figure B.9. In (a) we show a 32×32 square (out of a 512×512 via the FBP algorithm) and in (b) the same area with 64×64 samples taken from a 1024×1024 reconstruction via the PQI algorithm.

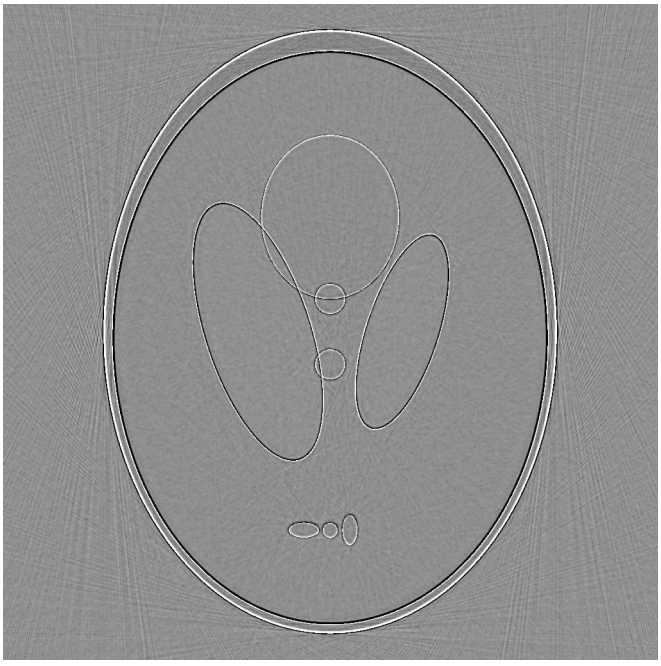
(a)



(b)

Figure B.11: Comparison of errors of 1024×1024 reconstructions using the standard FBP applied to noiseless data (a) and the data with added Gaussian noise (b). We observe that the Gaussian noise creates a speckle component in the error.

(a)



(b)

Figure B.12: The same comparison as in Figure B.11 but using the PQI algorithm of Section B.5. The effect of introducing Gaussian noise is qualitatively and quantitatively the same as that illustrated in Figure B.11.