Journal of Scientific Computing manuscript No.
(will be inserted by the editor)

Edge Detection from Non-Uniform Fourier Data Using the Convolutional Gridding Algorithm

Adam Martinez · **Anne Gelb** · **Alexander Gutierrez**

Received: date / Accepted: date

Abstract Detecting edges in images from a finite sampling of Fourier data is important in a variety of applications. For example, internal edge information can be used to identify tissue boundaries of the brain in a magnetic resonance imaging (MRI) scan, which is an essential part of clinical diagnosis. Likewise, it can also be used to identify targets from synthetic aperture radar (SAR) data. Edge information is also critical in determining regions of smoothness so that high resolution reconstruction algorithms, i.e. those that do not "smear over" the internal boundaries of an image, can be applied. In some applications, such as MRI, the sampling patterns may be designed to oversample the low frequency while more sparsely sampling the high frequency modes. This type of non-uniform sampling creates additional difficulties in processing the image. In particular, there is no fast reconstruction algorithm, since the FFT is not applicable. However, interpolating such highly non-uniform Fourier data to the uniform coefficients (so that the FFT can be employed) may introduce large errors in the high frequency modes, which is especially problematic for edge detection. *Convolutional gridding*, also referred to as the non-uniform FFT (NFFT), is a forward method that uses a convolution process to obtain uniform Fourier data so that the FFT can be directly applied to recover the underlying image. Carefully chosen parameters ensure that the algorithm retains accuracy in the high frequency coefficients. Similarly, the convolutional gridding edge detection algorithm developed in this paper provides an efficient and robust way to calculate edges. We demonstrate our technique in one and two dimensional examples.

This work is supported in part by grants NSF-DMS 1216559 and AFOSR 12004863

A. Martinez
School of Mathematical Sciences, University of Arizona, Tucson, AZ, 85721.
E-mail: admmartnez@gmail.com

A. Gelb
School of Mathematical and Statistical Sciences, Arizona State University, Tempe, AZ, 85287.
E-mail: annegelb@asu.edu

A. Gutierrez
School of Mathematics, University of Minnesota, Minneapolis, MN, 55455.
E-mail: alexg@umn.edu

Keywords Edge detection · Fourier data · Convolutional gridding · Non-Uniform Fast Fourier Transform

1 Introduction

The recovery of piecewise-smooth functions from a finite number of non-equispaced Fourier modes presents several problems. First, as in the case with uniform modes, there is the well known Gibbs phenomenon, which not only manifests itself as unwanted oscillations near the jump discontinuities but also reduces the overall convergence rate to first order, even in smooth regions, [15]. These oscillations may be exacerbated if the non-uniform data are resampled to integer locations, as the interpolation introduces additional error. Moreover, non-uniform sampling patterns are often designed to be sparse in high frequencies, which affects the robustness and accuracy of the interpolation in these regions, [23]. This is of particular concern for piecewise smooth functions or functions with steep gradients, as such features slow the decay rate of the Fourier coefficients.

The second challenge is that the use of non-uniform Fourier data precludes the implementation of an inverse fast Fourier transform (FFT), meaning that the complexity for a one dimensional reconstruction is $\mathscr{O}(n^2)$ instead of $\mathscr{O}(n\log n)$ for n given Fourier samples. This issue has been addressed in a variety of ways, depending on the application. One of the most common approaches to improving the computational efficiency is to design a window function that, when convolved (numerically) with the given Fourier data, allows for the approximation of the integer coefficients of a related function. The inverse FFT can then be straightforwardly applied with a cost close to $\mathscr{O}(n\log n)$. This method is commonly referred to as ‘non-uniform FFT’ (NFFT), [6], or ‘convolutional gridding’, [17, 19], which is the preferred term in the MRI community. Since our research is motivated by MRI, we will adopt that term here as well.

In addition to providing a computational speed up mechanism, convolutional gridding may also help to combat the convergence issues that are caused by trying to reconstruct a compactly supported function from a finite interval of Fourier data. Specifically, the (convolving) window function is designed to be “essentially” compactly supported in both domains. In so doing, we are assured a faster decay in the Fourier coefficients without introducing a large aliasing error.[1] Unfortunately, since the underlying function is typically only piecewise smooth, the Gibbs phenomenon is still present. While filtering helps to mollify the oscillatory effects, it also necessarily causes a loss of resolution near the internal boundaries, thereby negatively impacting clinical diagnoses (or target identification in SAR imaging).

High order reconstruction via spectral reprojection can successfully be applied to the convolutional gridding approximation, [23], resulting in faster convergence without degradation at the internal boundaries. However, in this case the edges of the function, or equivalently its regions of smoothness, must be known a-priori. Edge detection is also useful in itself as applications such as target identification and image segmentation rely heavily on accurate information about internal boundary structures. Since edge detection from uniform Fourier data is a well studied problem (see e.g., [13]) one option may be to first interpolate the data to uniform modes. To do this would be computationally less efficient and may even be less accurate, especially if large interpolation errors in the high frequency region are incurred. Edge detection can also be performed on the reconstructed image directly, e.g. by the

[1] This is assuming that the other parameters of the method, namely the density compensation factors, are suitably chosen, see e.g. [6, 16, 17, 19].

Canny edge detection method, [4]. However, as our numerical examples will demonstrate, such low order techniques are not capable of distinguishing between smooth variation in an image and its genuine edges.

Recovering a band-limited function $\hat{f}$ from its non-uniform samples is a well studied problem in sampling theory. In this regard, iterative techniques using compressed sensing are becoming more prevalent, and it is indeed possible to detect edges of a piecewise smooth function from its non-uniform Fourier data using a two stage process – by first recovering the uniform Fourier coefficients and then using an edge detection method such as the one described in [13]. The purpose of this paper is to establish a framework for determining edges *directly* using the convolutional gridding algorithm.

Our convolutional gridding edge detection algorithm has several advantages. First, with properly chosen parameters, interpolation errors can be largely controlled. Second, the method is computationally efficient since the FFT can be directly employed. Third, the technique is inherently multi-dimensional, provided that an edge can be suitably defined. Finally, convolutional gridding is a widely used algorithm in many applications for which there are reliable software packages. Since our adaptation only modifies the input data by multiplicative factors, it can be easily adopted in real applications.

The paper is organized as follows: Section 2 describes the convolutional gridding method for one dimensional function reconstruction. Section 3 explains the adaptation of the convolutional gridding method to recover the edges of a piecewise smooth function. Numerical examples in one and two dimensions are provided in Section 4, and in Section 5 we give some concluding remarks.

2 Convolutional Gridding

Consider a compactly supported piecewise smooth function f on $[-\pi,\pi]$. Suppose that Fourier data,

$$\hat{f}(\omega_k) = \int_{-\infty}^{\infty} f(x)e^{-i\omega_k x}dx = \int_{-\pi}^{\pi} f(x)e^{-i\omega_k x}dx, \tag{1}$$

are collected at non-integer values ω_k, $k = -N,\cdots,N$. In what follows we describe the convolutional gridding method to approximate the underlying function f.

At the heart of the convolutional gridding function is an (essentially) compactly supported smooth window function ϕ defined on $[-\pi,\pi]$ that is also (essentially) band limited in the Fourier domain. The choice of ϕ inherently affects the amount of aliasing error in the reconstruction and the efficiency of the calculation. A more extensive list of ideal properties for ϕ will be described in Section 3.1.

Let us define the function $g := f\cdot\phi$. The convolutional gridding algorithm proposes that the integer Fourier coefficients $\hat{g}(l)$, $l = -N,\cdots,N$, are constructed via convolution so that the FFT can be used to compute the Fourier partial sum

$$S_N g = \sum_{l=-N}^{N} \hat{g}(l)e^{ilx}. \tag{2}$$

The approximation of f is then given by $S_N g/\phi$.

We now turn our attention to the construction of $\hat{g}(l)$, $l = -N,\cdots,N$, whose analytic expression is given by

$$\hat{g}(l) = \int_{-\infty}^{\infty} f(x)\phi(x)e^{-ilx}dx = \int_{-\pi}^{\pi} f(x)\phi(x)e^{-ilx}dx. \tag{3}$$

Since f is only known at its non-uniform Fourier samples, $\hat{f}(\omega_k)$, we make the approximation

$$f(x) = \int_{-\infty}^{\infty} \hat{f}(\omega) e^{i\omega x} d\omega \approx \sum_{k=-N}^{N} \alpha_k \hat{f}(\omega_k) e^{i\omega_k x}, \tag{4}$$

where the weights $\{\alpha_k\}_{k=-N}^{N}$ are known as density compensation factors (DCFs) in applications such as MRI. There are many strategies for choosing DCFs, see e.g. [6,16,19], and we will discuss some of these ideas further in Section 3.1.

By substituting (4) into (3) we obtain

$$\hat{g}(l) \approx \int_{-\pi}^{\pi} \left(\sum_{k=-N}^{N} \alpha_k \hat{f}(\omega_k) e^{i\omega_k x} \right) \phi(x) e^{-ilx} dx = \sum_{k=-N}^{N} \alpha_k \hat{f}(\omega_k) \hat{\phi}(l - \omega_k). \tag{5}$$

Since $\hat{\phi}$ is (essentially) compactly supported by design, (5) can be truncated as

$$\hat{g}(l) \approx \sum_{k \text{ s.t.} |l-\omega_k|<q} \alpha_k \hat{f}(\omega_k) \hat{\phi}(l - \omega_k), \tag{6}$$

where q depends directly on the decay rate of $\hat{\phi}$. Replacing (5) with (6) is critical since $\hat{g}(l)$ cannot be computed by fast transform methods, and therefore the resulting calculation of (2) is only efficient if $\hat{g}$ has minimum support. Clearly, (6) suggests that the convolutional gridding method can also be viewed as a type of interpolation procedure. Hence the DCFs must be chosen carefully to avoid large interpolation error.

The inverse FFT is now used to construct an approximation for g from which ϕ is divided out in order to yield an approximation to f. The process is summarized in Algorithm 1.

Algorithm 1 *Convolutional Gridding*

Given $\widehat{f}$ at non-uniform frequency locations ω_k, $k = -N, \cdots, N$, for a piecewise smooth function f on $[-\pi, \pi]$:

1.Choose window function ϕ, DCFs $\{\alpha_k\}_{k=-N}^{N}$, and truncation parameter q.

2.Define a new function $g = f \cdot \phi$, so that $\widehat{g}(l) = \int_{-\infty}^{\infty} \hat{f}(\tau)\hat{\phi}(l-\tau)\,d\tau$

3.Regrid Fourier data to integer locations:

$$\hat{g}(l) \approx \sum_{k \text{ s.t.} |l-\omega_k|<q} \alpha_k \hat{f}(\omega_k) \hat{\phi}(l - \omega_k) =: \hat{g}_{cg}(l)$$

4.Reconstruct g on an equispaced grid using a standard inverse FFT:

$$S_N g_{cg}(x) = \sum_{l=-N}^{N} \widehat{g}_{cg}(l) e^{ilx}$$

5.Divide out window function:

$$f_{cg}(x) = \frac{S_N g_{cg}(x)}{\phi(x)} \tag{7}$$

end

We note that since the window function ϕ approaches zero at the ends of the interval, the result is usually "zero padded", i.e., f_{cg} is assumed to have zero value whenever ϕ is small, so that the final step in the algorithm remains well-conditioned. This is an appropriate assumption for applications such as MRI where any non-zero values near the boundaries are machine artifacts.

Figure 1 shows the convolutional gridding approximation of the simple step function,

$$f(x) = \begin{cases} 0, & \text{if } |x| \geq \frac{\pi}{2} \\ 1, & \text{if } |x| < \frac{\pi}{2}, \end{cases} \tag{8}$$

using DCFs given by (22) and window function defined in (21).

Fig. 1: Comparison of convolutional gridding reconstruction (red solid) of (8) given Fourier data (1) with $N = 32$ on jittered samples, (9), with standard Fourier reconstruction given uniform samples (blue dashed). The DCFs are given in (22), and the window function ϕ is given by (21) with $c = .6$ and $\lambda = 1$. The truncation parameter is $q = 12$.

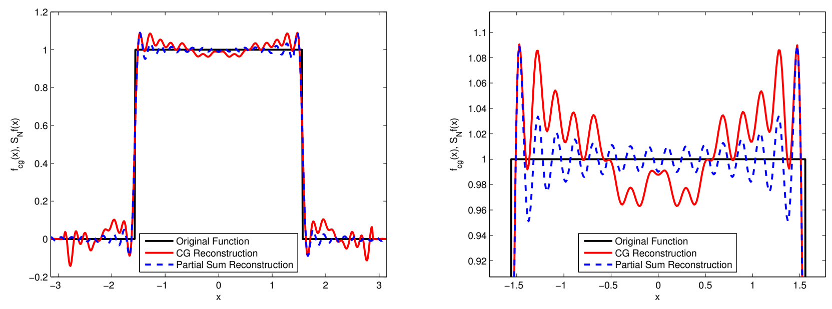


Figure 1 demonstrates that convolutional gridding yields oscillations coming from several sources. The Gibbs phenomenon is evident in the oscillations that increase in magnitude near the jump discontinuities of f for both the standard Fourier reconstruction (blue dashed line) as well as the convolutional gridding approximation (red solid line). However, the convolutional gridding algorithm admits additional oscillatory effects in the smooth regions. While filtering reduces the Gibbs related oscillations, it does not alleviate those caused by regridding, [23]. We point out that there are several methods that accurately reconstruct functions between the jump discontinuities without first interpolating the Fourier data. Interested readers should see [1,10] for the descriptions of two such methods. However, all high resolution methods require the edges of f to be known a-priori.

2.1 Sampling Schemes

Before describing the convolutional gridding approach for edge detection, we first introduce two examples of non-uniform sampling schemes that are representative in applications. They are depicted in Figure 2.

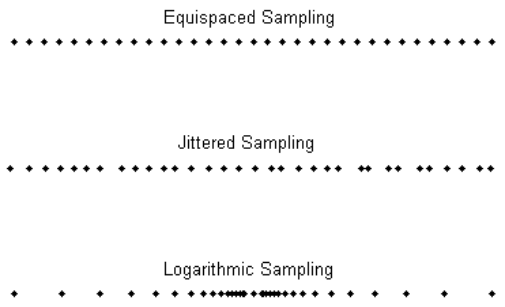
1. **Jittered Sampling:**

$$\omega_k = k \pm \mathscr{U}(0,\theta), \ k = -N,\ldots,N, \tag{9}$$

where $\mathscr{U}$ denotes a uniform distribution over the interval indicated. The jitter at every node k is independently determined and may therefore be positive or negative. In our experiments we set $\theta = \frac{1}{4}$ as the maximal jitter magnitude so that the sequence, $\omega_k, k \in \mathscr{Z}$, forms a Riesz basis for $L^2[-\pi,\pi]$ (satisfying Kadec's $1/4$ formula), [5]. Jittered sampling mimics the situation when a machine fails to sample at the exact integer locations.

2. **Logarithmic Sampling:** In this case the measurements are oversampled in the low frequencies and become increasingly sparse in the high frequencies. We let $\omega_0 = 0$ and the remaining $\omega_1, \ldots, \omega_N$ are distributed logarithmically between some lower bound $a > 0$ and a maximum frequency T. One dimensional logarithmic sampling is analogous to the various two dimensional spiral schemes that are prescribed as sampling trajectories in some MRI machines (see Figure 12).

Fig. 2: Depiction of the various sampling patterns



3 Convolutional Gridding for Edge Detection

Suppose now that f has a single jump discontinuity at the point $\xi \in (-\pi,\pi)$. We define the jump function of f as

$$[f](x) = f(x^+) - f(x^-), \tag{10}$$

which takes on the value $[f](\xi)$ at $x = \xi$ and 0 elsewhere. An equivalent formulation which is more convenient for our purposes is given by

$$[f](x) = [f](\xi) I_\xi(x), \tag{11}$$

where the indicator function $I_\xi(x)$ has value 1 at $x = \xi$ and 0 everywhere else. It is straightforward to expand (11) for multiple jumps.

We must first regularize $I_\xi(x)$ as it is nontrivial only at a single point, so therefore not feasibly represented by a partial Fourier sum. For example, we define

$$H_\xi(x) = e^{-\frac{(x-\xi)^2}{2\sigma^2}}, \tag{12}$$

which converges to $I_\xi(x)$ as $\sigma^2 \to 0$. We use (12) to approximate (11) as

$$\tilde{f}(x) := [f](\xi)H_\xi(x) \approx [f](x). \tag{13}$$

Note that $\tilde{f}(x)$ is a smooth function. The first term approximation of the Fourier transform of $H_\xi(x)$ at ω_k is given by [9]

$$\widehat{H}_\xi(\omega_k) \approx \sigma e^{-i\omega_k\xi} e^{-\frac{1}{2}\sigma^2\omega_k^2}\sqrt{\frac{\pi}{2}}\ . \tag{14}$$

As long as the discontinuity is not too close to the ends of the domain, that is $|\pi - \xi| > \delta$ for small $\delta > 0$, the remaining Fourier transform terms are of higher order and may be discarded. Combining (13) and (14) we see that

$$\begin{aligned}\widehat{\tilde{f}}(\omega_k) &\approx \widehat{[f](\xi)H_\xi}(\omega_k)\\ &= [f](\xi)\widehat{H}_\xi(\omega_k)\\ &\approx [f](\xi)\sigma e^{-i\omega_k\xi}e^{-\frac{1}{2}\sigma^2\omega_k^2}\sqrt{\frac{\pi}{2}}\ .\end{aligned} \tag{15}$$

We also need a relationship between the jump function $[f]$ and the given Fourier data. This is accomplished using integration by parts as

$$\begin{aligned}\hat{f}(\omega_k) &= \int_{-\pi}^{\pi} f(x)e^{-i\omega_k x}\,dx\\ &= \int_{-\pi}^{\xi^-} f(x)e^{-i\omega_k x}\,dx + \int_{\xi^+}^{\pi} f(x)e^{-i\omega_k x}\,dx\\ &= \frac{f(x)e^{-i\omega_k x}}{-i\omega_k}\bigg|_{-\pi}^{\xi^-} + \frac{f(x)e^{-i\omega_k x}}{-i\omega_k}\bigg|_{\xi^+}^{\pi} - \int_{-\pi}^{\pi} f'(x)\frac{e^{-i\omega_k x}}{-i\omega_k}\,dx\\ &\approx \frac{f(\xi^-)e^{-\omega_k\xi}}{-i\omega_k} - \frac{f(\xi^+)e^{-i\omega_k\xi}}{-i\omega_k} + \mathscr{O}\left(\frac{1}{\omega_k^2}\right)\\ &\approx \frac{[f](\xi)e^{-i\omega_k\xi}}{i\omega_k},\end{aligned} \tag{16}$$

yielding

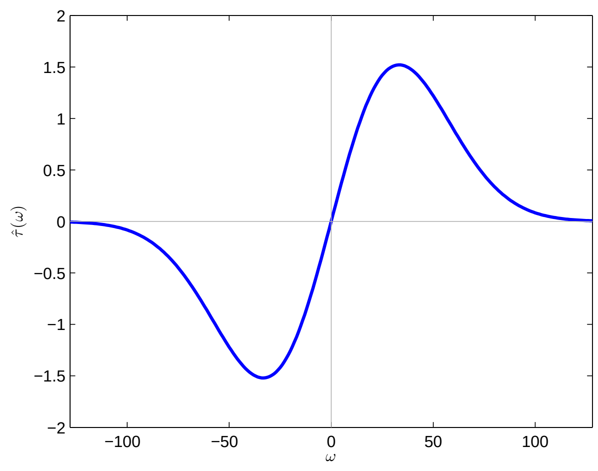
$$[f](\xi) \approx \hat{f}(\omega_k)i\omega_k e^{i\omega_k\xi}. \tag{17}$$

Finally, by substituting (17) into (15) we obtain an expression for the Fourier coefficients of the approximation of the jump function in (13) as

$$\widehat{\tilde{f}}(\omega_k) \approx \widehat{f}(\omega_k)i\omega_k\sigma e^{-\frac{1}{2}\sigma^2\omega_k^2}\sqrt{\frac{\pi}{2}} = i\hat{\tau}(\omega_k)\hat{f}(\omega_k)\ . \tag{18}$$

Figure 3 displays the resulting "filter" $\hat{\tau}(\omega)$. Note that both the low and high frequency coefficients are consequently damped.

Fig. 3: Edge detection filter in Fourier space



For multiple jumps, $\xi_j, j = 1, \cdots, M$, (11) is simply expanded as

$$[f](x) = \sum_{j=1}^{M} [f](\xi_j) I_{\xi_j}(x) \approx \sum_{j=1}^{M} [f](\xi_j) H_{\xi_j}(x) =: \tilde{f}(x), \tag{19}$$

and a similar derivation yields the same expression for (18). There is additional error, however, since the derivative term in (16) is now aggregated to include the total number of jumps. Also, when edges are in close proximity, the global nature of the Fourier coefficients will lead to interfering oscillations propagating from each jump discontinuity.

From here we may directly implement the convolutional gridding edge detection method by employing Algorithm 1. Specifically, f is replaced with the jump function approximation, $\tilde{f}$ in (19), with the corresponding Fourier data $\widehat{\tilde{f}}(\omega_k)$ computed using (18). We can summarize the above process with the following algorithm:

Algorithm 2 *Convolutional Gridding Edge Detection*

Given $\widehat{f}$ at non-uniform frequency locations ω_k, $k = -N, \cdots, N$, for a piecewise smooth function f on $[-\pi, \pi]$:

1.Choose window function ϕ, DCFs $\{\alpha_k\}_{k=-N}^{N}$, and truncation parameter q.

2.Define a new function $\tilde{g} = \tilde{f} \cdot \phi$, so that $\widehat{\tilde{g}}(l) = \int_{-\infty}^{\infty} \widehat{\tilde{f}}(\tau)\hat{\phi}(l-\tau)\,d\tau$, where $\tilde{f}$ is the approximation of the jump function $[f]$.

3.Choose parameter σ for the regularized indicator function (12) and calculate $\widehat{\tilde{f}}(\omega_k)$ from (18).

4.Regrid $\widehat{\tilde{f}}$ to integer locations:

$$\widehat{\tilde{g}}(l) \approx \sum_{k\,\text{s.t.}\,|l-\omega_k|<q} \alpha_k \widehat{\tilde{f}}(\omega_k)\hat{\phi}(l-\omega_k) =: \widehat{\tilde{g}}_{cg}(l) \tag{20}$$

5.Reconstruct $\tilde{g}_{cg}$ on an equispaced grid using a standard inverse FFT:

$$S_N \tilde{g}_{cg}(x) = \sum_{l=-N}^{N} \widehat{\tilde{g}}_{cg}(l) e^{ilx}$$

6.Divide out window function:

$$\tilde{f}_{cg}(x) = \frac{S_N \tilde{g}_{cg}(x)}{\phi(x)}$$

Note that if we choose $\phi \approx 1$ except near the boundaries, we eliminate the need for the final deapodization step.

end

3.1 Parameter Selection for Convolutional Gridding Edge Detection

We now describe how the parameters of the convolutional gridding edge detection method should be chosen.

The window function ϕ: The window function for convolutional gridding edge detection should satisfy the following properties (the first three also apply to standard function reconstruction, [19]):

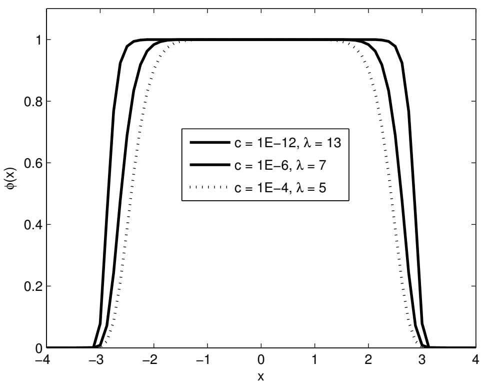
1. To minimize the computational cost while maintaining high accuracy, $\widehat{\phi}(\omega)$ should be "essentially" compactly supported. This will ensure that each $\hat{\tilde{g}}_{cg}(l)$ in (20) can be computed with minimal number of operations.
2. $\phi(x)$ should be nonzero in the reconstruction interval, since the final approximation is given by $f = g/\phi$. However, we can assume that the underlying function f is "zero padded" and therefore there are no discontinuities near the boundaries. This is a reasonable assumption in applications such as MRI in which we are primarily concerned with depiction of spatially compact body parts.
3. Since we are reconstructing a periodic extension of g, any non-zero values outside the domain of interest will be aliased back into the approximation. We can reduce the impact of aliasing by choosing ϕ to be zero outside the domain of interest.
4. Since $[f](x)$ and consequently $[g](x)$ is sparse, the numerical division by ϕ is particularly susceptible to round-off error when ϕ is small. We can reduce this susceptibility by requiring $\phi \approx 1$ except near the boundaries, which eliminates the need to perform division, i.e. the final step in Algorithm 2. Consequently we cannot recover edges that occur near the ends of the domain. As noted previously, this does not have negative implications in applications such as MRI as nonzero values near the boundaries are considered to be machine artifacts.

One function that satisfies these conditions is

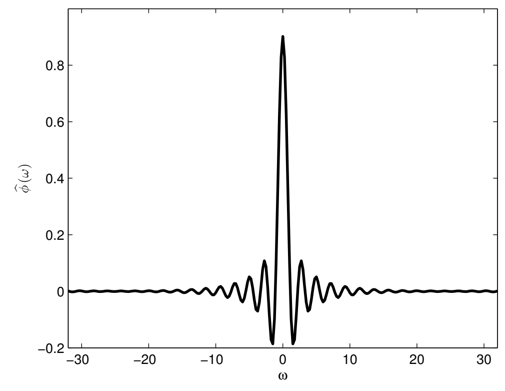
$$\phi(x) = e^{-cx^{2\lambda}}, \qquad c, \lambda > 0 \tag{21}$$

When c is very small, ϕ is sometimes called a "super-Gaussian." It was used effectively in [14, 20] as a robust inner product weight for spectral reprojection. Unfortunately, there is no explicit formula for $\widehat{\phi}(\omega)$. Numerical implementation in one dimension is accurate and straight forward, but for our two dimensional examples we use the Kaiser Bessel window, [6, 19], which is part of the two dimensional NUFFT software package developed by Fessler et al.[2]

[2] `web.eecs.umich.edu/~fessler/code/`.



(a) Window function $\phi(x) = e^{-cx^{2\lambda}}$



(b) Fourier transform $\hat{\phi}(\omega)$

Fig. 4: Window function and corresponding Fourier transform with $c = 10^{-12}$ and $\lambda = 13$

Figure 4 shows several super-Gaussian window functions and depicts the Fourier transform with $c = 10^{-12}$ and $\lambda = 13$. The transform is essentially compactly supported in $[-15, 15]$, which allows us to truncate the approximation of the convolution integral to include values only within this window.

Density Compensation Factors (DCFs): One way to choose the density compensation factors α_k is to interpret the approximation to (5) as numerical quadrature. For example,

$$\begin{aligned}\alpha_k &= \frac{1}{2}(\omega_{k+1} - \omega_{k-1}), \qquad \text{for } k = -N+1, ..., N-1,\\ \alpha_{-N} &= \omega_{-N+1} - \omega_{-N},\\ \alpha_N &= \omega_N - \omega_{N-1}.\end{aligned} \tag{22}$$

are the trapezoidal rule quadrature weights.

Remark 1 *For sampling patterns that collect data more sparsely for increasing N, choosing the DCFs as quadrature weights will* not *yield a pointwise converging approximation to the regridded coefficients, either for function reconstruction or edge detection. This is due to the fact that the resolution of the samples does not increase with N. Nevertheless, the error in the regridded coefficients may be somewhat mitigated by the choice of window function ϕ and even the regularized indicator function H. We include the trapezoidal DCFs in our numerical examples to demonstrate the robustness of our edge detection method. Better DCFs and window functions for edge detection will be studied in future work.*

Remark 2 *There are several implications associated with how the window function ϕ is selected.*

1. *As stated above, if the function is presumed to be zero near the boundaries (zero-padded), then choosing $\phi = 1$ except near the boundaries eliminates the need for the final deapodization step, and reduces the susceptibility of the approximation to round-off error. Consequently we must also assume that there are no edges near the boundary (which is consistent with zero padding).*

2. *Determining edges from the regridded $\hat{\tilde{g}}(l)$ should be easily accomplished using standard Fourier based edge detection methods such as in [13].*[3] *However, this requires extra processing, which is costly in two dimensions.*
3. *There does not appear to be a closed form representation of $\hat{\phi}$ for a smooth $\phi \approx 1$ throughout the interior of the domain. Because of this, in two dimensions we use the Kaiser Bessel window for ϕ. Although not close to 1 throughout the interior of the domain, the Kaiser Bessel window has the advantage of having explicitly known Fourier coefficients, which accounts for its widespread use in commercial software packages. There are two main consequences, however. First, when a Kaiser Bessel window is used, the resulting $\hat{\tilde{g}}$ is either not as accurate, since the corresponding decay rate of $\hat{\phi}$ is slower, or is more expensive to compute, since more terms are required in (20). Second, deapodization must now be performed in the final step of the process, causing additional round-off error. If conventional codes are to be used, deapodization must be done prior to any additional processing to recover edges, meaning that it is impossible to separate $\hat{\tilde{g}}$ from the recovered image. Since our convolutional gridding edge detection only modifies the input data via (18), we are able to complete all of our processing before deapodization.*

Having chosen a window function and DCF technique, we now illustrate the use of Algorithm 2 on the sawtooth function given by

$$f(x) = \begin{cases} \frac{-\pi - x}{2\pi}, & \text{if } x \leq 0 \\ \frac{\pi - x}{2\pi}, & \text{if } x > 0. \end{cases} \tag{23}$$

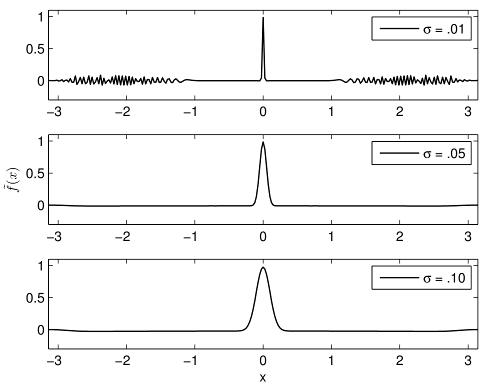
Figure 5(a) demonstrates that the DCFs generated by (22) may be adequate for recovering the regridded Fourier coefficients for the regularized jump function, (19), as long as σ is not too small, i.e. when the regularized indicator function is more "blurred". This is also suggested by Figure 5(b), which displays the corresponding decay rate of $\hat{\tilde{g}}_{cg}$. As will be discussed further in Section 4, the analogous DCFs for the two dimensional case may yield too large of an error in the high frequency coefficients to produce any meaningful results.

Iterative methods are now often used to determine the DCFs in the convolutional gridding algorithm, as it has been demonstrated that the resulting approximation has less interpolation error, [16, 19]. Because of that, they are also more effective for our convolutional gridding edge detection, especially in two dimensions. Figures 6 and 7 compare the iterative and trapezoidal rule DCFs for the jittered and logarithmic sampling patterns, respectively.

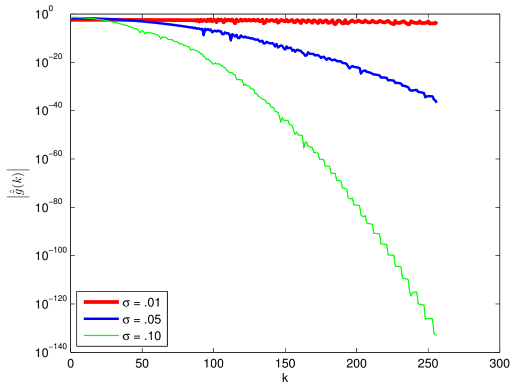
As expected, the trapezoidal rule DCFs increase in value as the density of points decreases in the log sampling distribution. (The small jump at the origin in Figure 7(a) occurs since the distribution starts at a set distance from the origin.) Iterative DCFs, on the other hand, have a more even distribution, which is consistent with the linear least squares problem solved to obtain them. In the case of jittered sampling, the difference is less discernible, as the density between the points is basically uniform throughout the domain. A band limited matrix of DCFs was designed in [11] for one dimensional convolutional gridding. While numerical convergence to the underlying function is dramatically improved in the one dimensional case, the corresponding two dimensional DCFs have yet to be constructed. Hence we leave their use for future work.

Finally, Figure 8 shows the results for the convolutional gridding edge detection method when complex Gaussian noise is added to the Fourier samples for both the trapezoidal rule

[3] In this case, although the operations are not commutative, it is clear that there are admissible concentration factors in the method described in [13] that account for these interim interpolating steps.

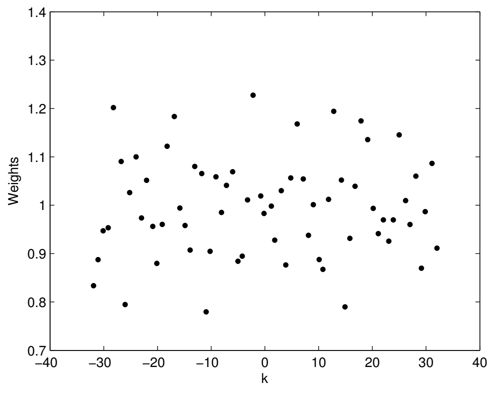


(a) Jump function approximation

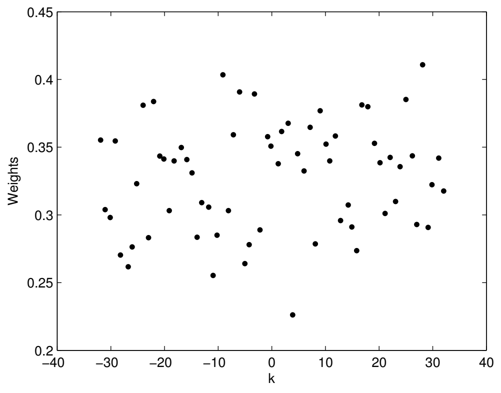


(b) Regridded Fourier coefficients

Fig. 5: (a) Convolutional gridding edge detection for (23) given 513 logarithmically spaced Fourier samples. The DCFs are given by (22), and the window function is given by (21) with $c = 10^{-12}$ and $\lambda = 13$. The regularized indicator function is given by (12) with various choices of σ. (b) Log plot of $|\hat{\tilde{g}}_{cg}(k)|$.



(a) Trapezoidal rule DCFs for jittered sampling



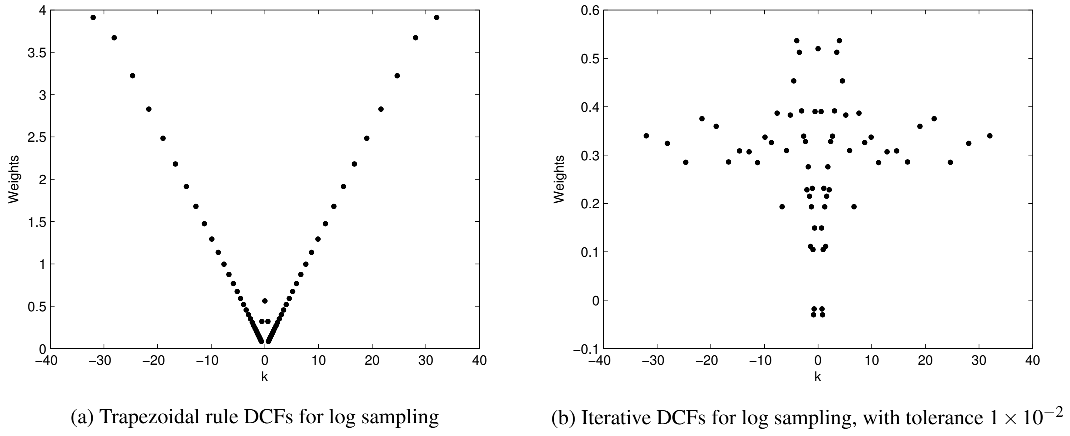
(b) Iterative DCFs for jittered sampling

Fig. 6: A comparison of trapezoidal rule and iterative DCFs for jittered sampled Fourier data. $N = 32$.

and iterative DCFs. Note that the trapezoidal rule DCFs increase the contribution of the high frequency modes, and the corresponding errors are exacerbated by the added noise. It is also evident that reducing the noise in the smooth regions will necessarily cause "smearing" over the jump locations.

4 Numerical examples

We now provide both one and two dimensional numerical experiments to demonstrate the efficacy of the convolutional gridding edge detection method. For context, we compare our results with those that perform edge detection as a post-processing procedure. Specifically,



(a) Trapezoidal rule DCFs for log sampling (b) Iterative DCFs for log sampling, with tolerance 1×10^{-2}

Fig. 7: A comparison of trapezoidal rule and iterative DCFs for logarithmically sampled Fourier data. $N = 32$.

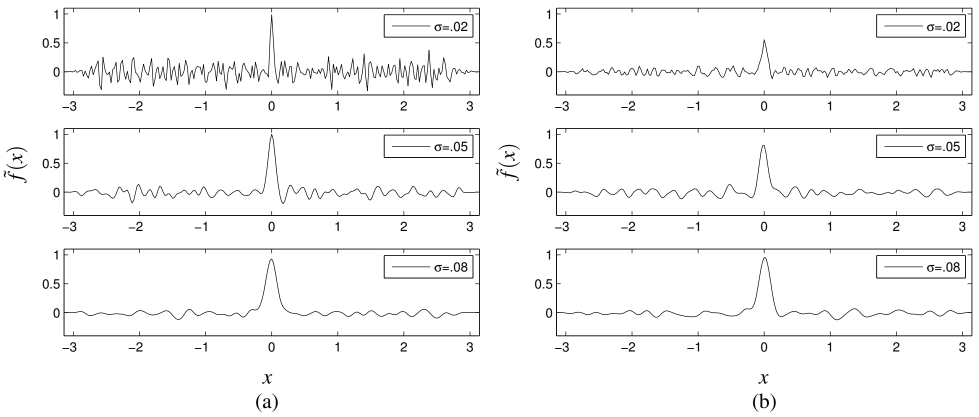


Fig. 8: Jump function approximation for (23) given logarithmically sampled Fourier data with $N = 256$ using (a) trapezoidal DCFs and (b) iterative DCFs. Complex Gaussian noise with a variance of $.01^2$ is introduced to each Fourier mode. Indicator function width parameter σ is varied.

we compute the discrete (uniform) Fourier coefficients of f_{cg} in (7) and then employ the concentration factor method, [13], which constructs the jump function approximation from a finite sampling of uniform Fourier data. We observe that the results are very similar, although as mentioned previously, using edge detection as a post-processing method is more costly. We note that the original concentration factor method can be made to be high order in smooth regions by using an oscillatory regularized indicator function that has canceling moments in the corresponding harmonic partial sum expansion, [12]. However, more oscillations are introduced near the jump discontinuities. In the case where the Fourier data is non-equally spaced, the error incurred at the regridded high frequency modes exacerbates these oscillatory effects and causes inaccurate jump values. Since the underlying sequence is non-harmonic, the convolutional gridding edge detection algorithm uses a non-oscillatory regularized indicator (bump) function, e.g. (12), resulting in a smoothing effect of the jump function approximation (recall Figure 3). As will be demonstrated later, it is still able to distinguish jump discontinuities from steep gradients, which is not typically the case for image based edge detection methods, such as those found in [4,7]. Because the regridded coefficients are more likely to be inaccurate in the high frequency modes, it is better to apply the non-oscillatory regularized indicator function, as it will help to damp the inaccurate high frequency coefficients. Results from the concentration factor method based on given integer Fourier data, using the same regularized indicator function in (12) to determine the concentration factors, are also included for reference.

4.1 One dimensional convolutional gridding edge detection

Figure 9 compares our convolutional gridding edge detection results for (23) to the results given by the concentration factor method used both as a post-processing technique for the convolutional gridding function approximation (Algorithm 1) as well as when the Fourier data is uniformly sampled. In the convolutional gridding cases, the data was sampled logarithmically. To demonstrate robustness, we used the trapezoidal DCFs given in (22), and note that using the iterative DCFs did not yield significantly better results for our examples. We also used the window function defined by (21) with $\sigma = .03$. The same comparison is made in Figure 10 for a function containing six jump discontinuities. The steep gradient between the fifth and sixth jump emphasizes the need for accurate high frequency information.

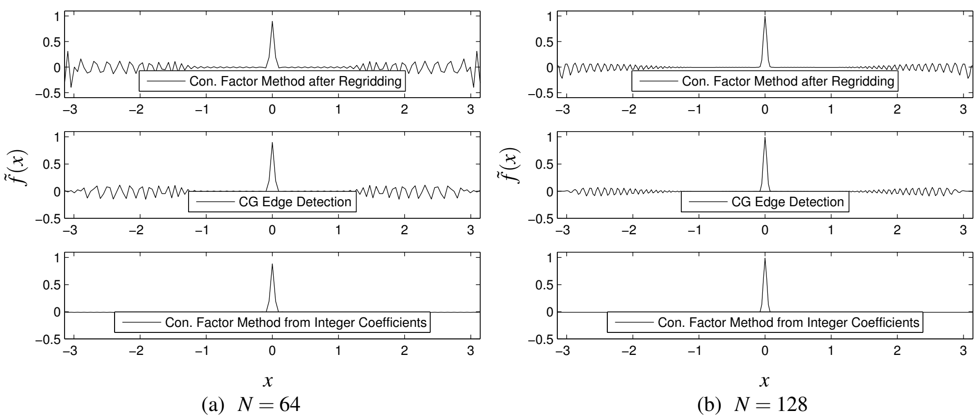
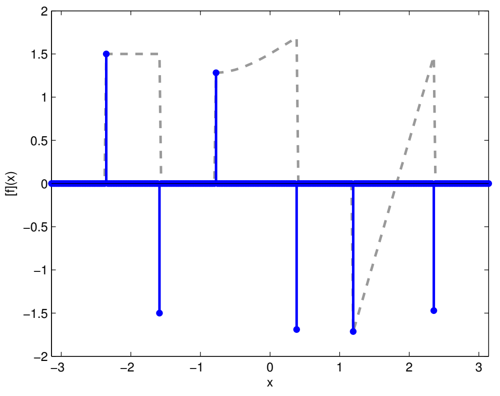


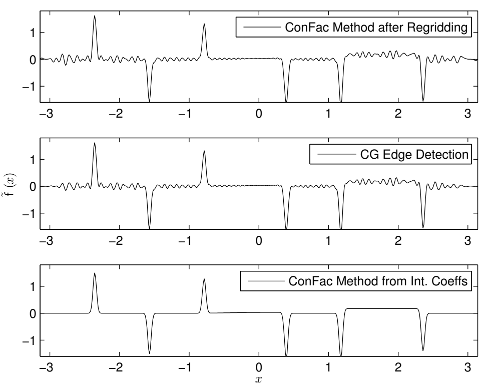
Fig. 9: Edge detection results for the sawtooth function (23) using (top) the concentration factor method after convolutional gridding, (middle) the convolutional gridding edge detection method, and (bottom) the concentration factor method given uniform Fourier data.

4.2 Two dimensional convolutional gridding edge detection

In two dimensions we face the added challenge of not having an intuitive definition of a jump discontinuity. To describe how our convolutional gridding method may be applied, we



(a)



(b)

Fig. 10: (a) Original function and jump function. (b) Comparison between convolutional gridding edge detection and concentration factor method. The concentration factor method is performed both after convolutional gridding and for given integer coefficients. In each case $N = 256$.

use an analogous integration by parts approach to define a two dimensional "edge", but note that other definitions may be more useful for a particular application domain.

Suppose $f : \mathbb{R}^2 \to \mathbb{R}$ is a compactly supported piecewise smooth function on some proper subset of $[-\pi, \pi]^2$ for which we are given non-uniform Fourier data,

$$\widehat{f}(\omega_k, \omega_\ell) = \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} f(x,y) e^{-i(\omega_k x + \omega_\ell y)} dx\, dy, \tag{24}$$

where $-N < k, l < N$. We define the jump function as

$$[f](x,y) := \sum_{j=1}^{M} [f](P_j) I_{P_j}(x,y), \tag{25}$$

where the x and y components of each discontinuity P_j, $j = 1, \cdots, M$, are given by (ξ, η), and $I_{P_j}(x,y)$ is the indicator function which has value 1 at each P_j and 0 everywhere else. As discussed previously, it is not straightforward how the jump value, $[f](P)$, should be calculated. Our derivation below demonstrates that there is a reasonable interpretation to $[f](P)$ that relates the Fourier coefficients of the image to (25). For ease of presentation we assume that f has one jump discontinuity at the point (ξ, η). The derivations that follow extend naturally for multiple jumps, assuming that the internal boundaries of the underlying image consist of smooth curves.

As in the one dimensional case, we will need to regularize the indicator function I_P, leading to the approximation

$$[f](x,y) \approx \sum_{j=1}^{M} [f](P_j) H_{P_j}(x,y) =: \tilde{f}(x,y). \tag{26}$$

The simplest approach to constructing H_P is to use a separable function, e.g.

$$H_P(x,y) = H_\xi(x)H_\eta(y),$$

and then choose each one dimensional regularized indicator function according to the previously established admissibility. For example, we can use (12) to obtain

$$H_P(x,y) = \exp\left(-\frac{(x-\xi)^2+(y-\eta)^2}{2\sigma^2}\right). \tag{27}$$

Assuming that the internal edges do not lie near the ends of the domain, the corresponding Fourier transform can be approximated by its leading term

$$\widehat{H}_P(\omega_k,\omega_\ell) \approx 2\pi\sigma^2 e^{-i(\xi\omega_k+\eta\omega_\ell)}e^{-\frac{1}{2}\sigma^2(\omega_k^2+\omega_\ell^2)}. \tag{28}$$

We also need a relationship between the given Fourier data, $\hat{f}(\omega_k,\omega_\ell)$, and the approximate jump function, (26). As in the one dimensional case, we will apply integration by parts. Let us first define $[f](\xi,y) := f(\xi^+,y)-f(\xi^-,y)$ to be the jump function in y along the coordinate of discontinuity in x. Then, ignoring higher order terms we have for each fixed y:

$$\begin{aligned}\int_{-\pi}^{\pi} f(x,y)e^{-i\omega_k x}dx &\approx \left(\frac{f(x,y)e^{-i\omega_k x}}{-i\omega_k}\bigg|_{-\pi}^{\xi^-} + \frac{f(x,y)e^{-i\omega_k x}}{-i\omega_k}\bigg|_{\xi^+}^{\pi}\right)\\ &= -\frac{e^{-i\omega_k\xi}}{i\omega_k}(f(\xi^-,y)-f(\xi^+,y))\\ &= \frac{e^{-i\omega_k\xi}}{i\omega_k}[f](\xi,y). \end{aligned}\tag{29}$$

By assumption, the only discontinuity in $[f](\xi,y)$ is at $y=\eta$. Thus, disregarding higher order terms, we can substitute (29) into (24) and obtain

$$\begin{aligned}\hat{f}(\omega_k,\omega_\ell) &\approx \frac{e^{-i\omega_k\xi}}{i\omega_k}\left(\int_{-\pi}^{\eta^-}[f](\xi,y)e^{-i\omega_\ell y}dy + \int_{-\eta^+}^{\pi}[f](\xi,y)e^{-i\omega_\ell y}dy\right)\\ &\approx -\frac{e^{-i(\xi\omega_k+\eta\omega_\ell)}}{\omega_k\omega_\ell}[f](\xi,\eta), \end{aligned}\tag{30}$$

where we say that $[f](\xi,\eta)$, which approximates the difference of f in x and y across an internal boundary curve at the point (ξ,η), is the approximate jump value $[f](P)$ in (25). Note that when $[f](\xi,y)$ is continuous (in y) the right hand side in (30) is zero, that is, only higher order terms remain in the integration by parts approximation. The same is true, of course, if we reverse the order of calculation, and $[f](x,\eta)$ is continuous in x. This yields an important consequence for our method, namely, that if an edge occurs in either the x or y direction, but *not* both, it will not be located. Figure 11(b) demonstrates this unwanted effect. From this perspective it may be advantageous to define a two dimensional edge as the magnitude of the gradient vector. However, in this case the relationship between the Fourier coefficients and the jump function is nonlinear, and the corresponding derivation becomes more complicated. As will be demonstrated later, another way to alleviate this difficulty is to reconstruct the jump function twice – first as prescribed above, and second to a rotated image.

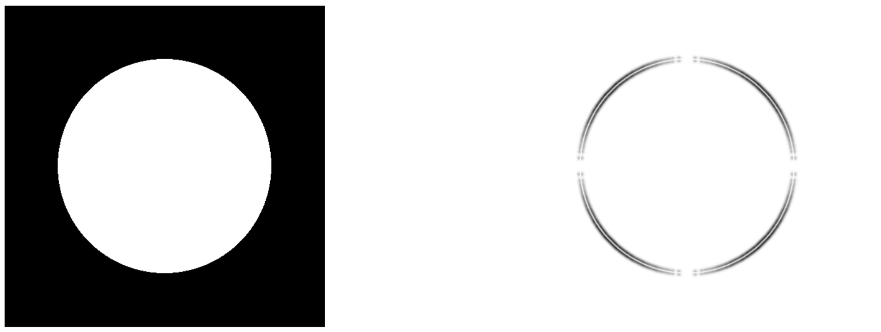
Finally, we obtain an approximation to the Fourier transform of the jump function using (30) and (28) as

$$\widehat{[f]}(\omega_k,\omega_\ell) \approx \widehat{\tilde{f}}(\omega_k,\omega_\ell) = [f](\xi,\eta)\widehat{H_p}(\omega_k,\omega_\ell) \approx -2\pi\sigma^2\omega_k\omega_\ell\hat{f}(\omega_k,\omega_\ell)e^{-\frac{1}{2}\sigma^2(\omega_k^2+\omega_\ell^2)}. \tag{31}$$

The coefficients in (31) are now directly implemented into the two dimensional version of Algorithm 2 to recover the approximation of the edge map given by (26). Although we plot the approximation to $|\tilde{f}(x,y)|$ in all of our examples that follow, we note that it is feasible that some useful information may be obtained from retaining the signed non-zero values in (26).

In our two dimensional experiments we used the convolutional gridding code provided by in the Image Reconstruction Toolbox[4] which uses an iterative process to construct the DCFs and the Kaiser Bessel window as $\phi(x,y)$. We note again (see Remark 2) that the first three requirements outlined in Section 3.1 for a window function are satisfied, but the Kaiser Bessel window does not have the value of 1 in most of the domain. Hence we must divide by ϕ (deapodization), yielding additional round off error in the reconstruction of the two dimensional edge map.

Figure 11 demonstrates the results for (26) when the Fourier data is sampled uniformly. The target image is a circle of unit value. As predicted by the formulation of (30), the method is unable to detect edges that lie only in the horizontal or vertical direction. In previous investigations concerning edge detection in images from uniform Fourier data, it was found that the results were best when a line by line approach was used in each dimension and then combined to form an edge map, [2,3,8]. In the case of non-uniform sampling, this would mean that the convolutional gridding reconstruction would first have to be computed, and then the FFT used to generate uniform Fourier coefficients.



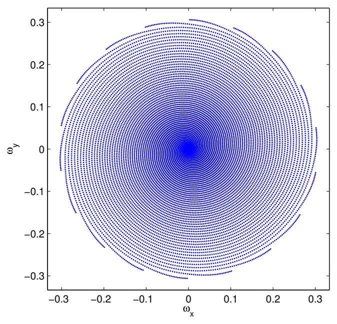
(a) target image (b) edge map

Fig. 11: Target image and corresponding edge map given 257×257 equispaced Fourier samples.

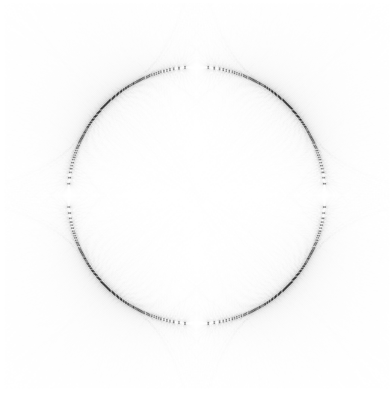
Figure 12(a) displays a representative trajectory from which Fourier samples are collected in MRI. In our examples we use this spiral trajectory made up of 463,247 sampling

[4] `http://web.eecs.umich.edu/~fessler/code/`

points, approximately a 680^2 grid. Figure 12(b) depicts the corresponding edge map for the target image using the convolutional gridding approximation of (26).



(a) spiral sampling example



(b) edge map

Fig. 12: (a) Fourier sampling pattern (b) convolutional gridding edge detection.

As in Figure 11(b), the edges are not seen on the vertical and horizontal axes. One way to rectify this for both the uniform and non-uniform cases is to rotate the figure before applying the edge detection algorithm. This will cause the "missing" edges to occur at the angle of rotation, rather than on the vertical and horizontal axes. Specifically, we compute

$$\tilde{f}_R(x,y) = \mathbf{R}^{-1}\mathbf{E}\mathbf{R}\widehat{\tilde{f}}(\omega_x,\omega_y), \tag{32}$$

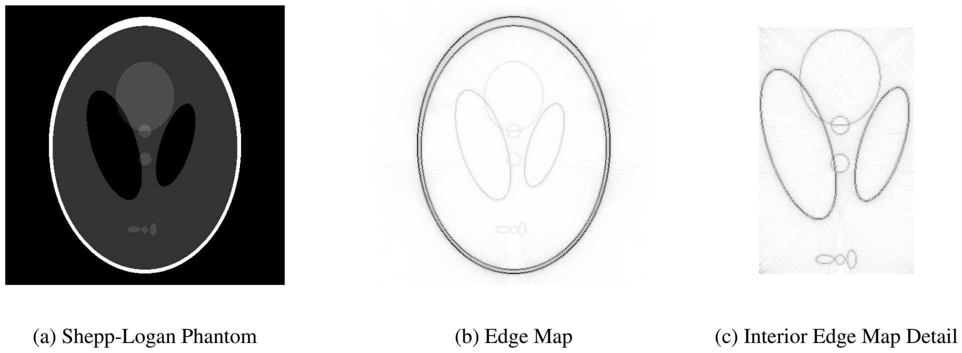
where $\mathbf{R}$ is the standard rotation matrix

$$\mathbf{R} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix},$$

$\mathbf{E}$ is the convolutional gridding matrix, and $\widehat{\tilde{f}}(\omega_x,\omega_y)$ are the Fourier samples of the approximated regularized jump function. Images with missing edges occurring at different angles are then combined (averaged or otherwise) to provide a complete description of the edges.

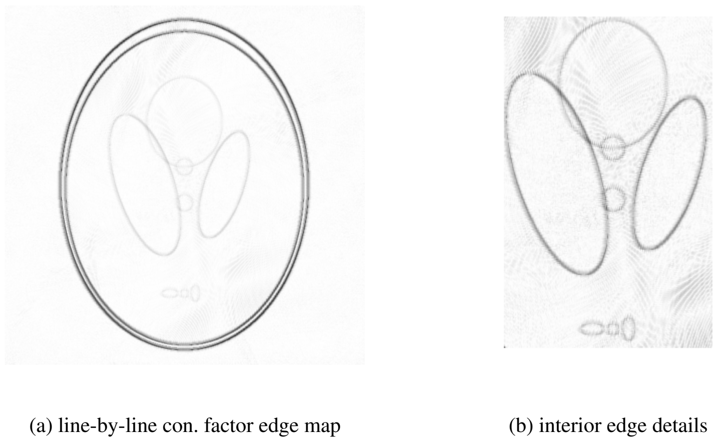
Figure 13 demonstrates our edge detection method on the famous Shepp-Logan phantom image, e.g. [22]. Here we have implemented the rotation described by (32), combining results using $\theta = 0$ and $\theta = \frac{\pi}{4}$ and then taking the maximum between the two reconstructions.

We note that there are still unwanted artifacts present in Figure 13(b) and (c). Furthermore, the aggregation of rotated images (done by taking the maximum in order to enusre the edges are captured) may indeed intensify these unwanted features. Thresholding based on a-priori knowledge of the underlying image may help to remove some of the unwanted artifacts. For comparison purposes, Figure 14 uses a line by line implementation of the concentration factor method in the vertical and horizontal directions on data that have first been processed by the usual convolutional gridding algorithm. The results from both directions are averaged to form the edge map. Besides the extra computational cost, it appears that the error caused by the final deapodization step results in additional oscillations, some of which



(a) Shepp-Logan Phantom (b) Edge Map (c) Interior Edge Map Detail

Fig. 13: Convolutional gridding edge detection for the Shepp-Logan phantom using rotation. The data are sampled at $463,247$ non-uniform points.



(a) line-by-line con. factor edge map (b) interior edge details

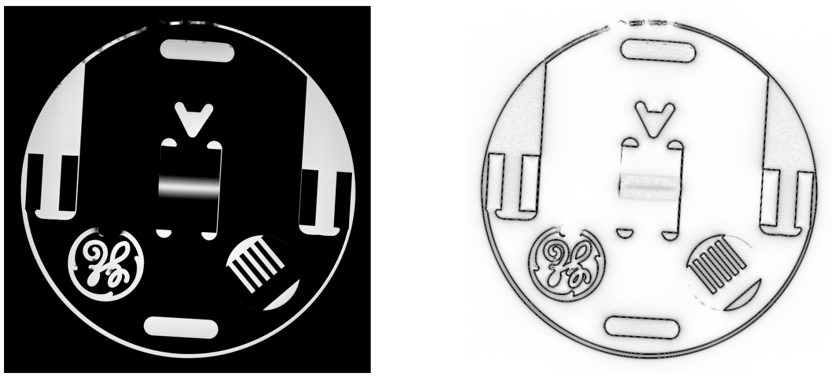
Fig. 14: Edge reconstruction from a line-by-line concentration factor method applied as a post-processing method to the convolutional gridding approximation. The reconstructed uniform Fourier data are placed on a $513X513$ grid.

have almost the same intensity values as those in the Shepp Logan phantom. This will be investigated more in future work.

Figure 15 demonstrates the convolutional gridding edge detection method on an actual MRI image. Here we are given a highly resolved (833×833) reconstructed image from which we compute the non-uniform Fourier data on a spiral sampling pattern resembling Figure 12. The total number of non-uniform points used is $463,247$, which is approximately 680^2.

4.3 Comparison to Other Edge Detection Algorithms

It is also feasible to post-process the image obtained via convolutional gridding using a pixelated edge detection algorithm, such as in [4, 7]. These methods essentially detect edges by



(a) target image (b) edge map

Fig. 15: GE phantom and its edge map.

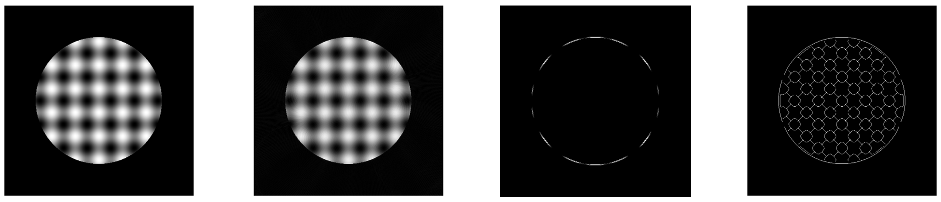
approximating (low order) first derivatives using neighboring pixel data. While very effective for piecewise constant images, they typically are unable to distinguish edges from steep gradients or even smooth variation. A distinct advantage of the convolutional gridding edge detection method over these low order pixel based methods is that, because of its ability to discern edges from steep gradients and other types of variation, it is less likely to register false edges. Figure 16 compares the results of the convolutional gridding edge detection method with an edge map obtained by applying the Canny edge detection algorithm, [4], found in the MATLAB image processing toolbox. In the latter case, the target image is first reconstructed using the standard convolutional gridding technique. The target image, seen in Figure 16(a), is created by overlaying the function given by

$$h(x,y) = \frac{1}{2}\left(\cos(4x)^2 + \sin(4y)^2\right) \tag{33}$$

onto the circle of the type depicted in Figure 11(a). It should be noted that the patterned surface inside the circle contains no discontinuities and has continuous derivatives. Moreover, the "holes" seen on the circle's boundary in Figure 16(c) accurately depict a smooth transition from the interior surface to that boundary. On the other hand, as demonstrated in Figure 16(d), the Canny method is unable to distinguish the smooth variation in (33) from the actual edges, which may cause a false registration of edges in clinical diagnoses. It was also shown in [18] that noise introduced into the Fourier data, even in the uniform case, will further exacerbate the problem of false registration when a pixel based method is used.

5 Concluding Remarks

In this paper we developed an edge detection method from non-uniform Fourier data using the convolutional gridding algorithm. Our method has several advantages. First, it is efficient since the FFT can be employed. Second, large interpolation errors in the high frequency coefficients can be controlled by careful selection of the parameters in the convolutional



(a) target image (b) CG reconstruction (c) CG edge detection (d) CG reconstruction followed by Canny edge detection

Fig. 16: Comparison of convolutional gridding edge detection with Canny edge detection used as a post-processing method. 463,247 non-uniform points were used.

gridding algorithm. This is especially important as critical information about edges is carried in these modes. Third, it is a single step process, meaning that the edges are determined directly from the Fourier data and the image does not have to be reconstructed. This implies that processes such as target identification and tissue segmentation can be achieved without first reconstructing an image. Moreover, as demonstrated by our numerical examples, our method is more accurate than edge detection algorithms that post-process reconstructed images. Finally, our method appears to be robust in the presence of noise, although this issue must be further explored.

Our method requires the use of several parameters, which can potentially be tuned for particular applications. While the window function design appears to be fairly robust, it is critical that the construction of the DCFs ensure that error is minimized in the high frequency coefficients. This will be the topic of future investigations, especially for multi-dimensions. In addition, as our method is described here, it does not take advantage of any prior information about the underlying function. Sparsity enforcing techniques, such as the one described in [21], can be employed with some modification to enhance the quality of the edge map. The edges might also be characterized in some specific way, depending on the application. We will explore algorithms that consider prior information in future investigations.

6 Appendix

The following script implements the convolutional gridding edge detection algorithm in one dimension for given Fourier coefficients of a sawtooth function:

```
 %% Gridding Reconstruction
% This script uses the Fourier data of f at 2N+1 nonuniform points to
% reconstruct an approximation of the jump function [f](x).

clear all; clc; close all;
%% Non-Uniform Sampling pattern
% No. of input Fourier modes is 2N+1
N = 256;

% "Jittered" samples
% k = [-N:N] + sign( (rand(1,2*N+1)-.5) ) .* rand(1, 2*N+1) *.25;

% Logarithmic sampling
 k = [-fliplr(logspace(-.25, log10(N), N)) 0 logspace(-.25, log10(N), N)];

%% Function and Fourier coefficients
numpts = 513;
x = linspace(-pi,pi,numpts);
x(end) = [];

% unit ramp
f = @(x) ((-pi-x)/2/pi).*(x<=0)+((pi-x)/2/pi).*(x>0);
fx = f(x);
fxhat = 1./(2*pi*1i*k).';
fxhat(N+1) = 0;

%% Noise is introduced to spectral data
noiseVar = .01^2; % noise variance
noiseCfs = sqrt(noiseVar/2)*randn(2*N+1,1) + ...
                         1i*(sqrt(noiseVar/2)*randn(2*N+1,1));
fxhat = fxhat + noiseCfs;

%% Jump Function transform approximation at non-uniform points

sigma = .03; % width parameter for regularized indicator

% Gaussian Indicator function
jhat = fxhat*1i.*k.'*2*sigma*sqrt(pi/2).*exp(-1/2*sigma^2*k.'.^2);

%% (TRAPEZOIDAL) Density compensation
wts = zeros(1,2*N+1);
for ind_a = 2:2*N
   wts(ind_a) = .5*k(ind_a+1)-.5*k(ind_a-1);
end
wts(1) = k(2)-k(1); wts(end) = k(end)-k(2*N);

jweighted = jhat.*wts.';

%% Gridding
c = 1.E-12; % Gaussian width
lambda = 13;

e = 15; % phi_hat(w) = 0, |w|>e

```

```
phi = @(x) exp(-c*(x).^(2*lambda));

% The gridding procedure
intModes = -N:N;
eqMat = repmat(intModes,2*N+1,1);
kMat= repmat(k.',1,2*N+1);
loc = abs(eqMat-kMat)<= e;
phat = fourierCoeffs(phi(x), (eqMat-kMat).', x);
regrid = jweighted.'*(phat.*loc);

%% Reconstruction
% Compute a Fourier partial sum reconstruction
% Fourier matrix (2pi-periodic function)
four = exp(1i*x.'*(-N:N));

% The partial sum
g = four*regrid.';
g = real(g);

%% Plot results
figure;
plot(x, fx, 'k','LineWidth',2);
hold on;
plot(x,g,'b','LineWidth',2);
xlabel('x');
ylabel('y');
xlim([-pi pi])
legend('Original Function','Edge Reconstruction')
```

References

1. ADCOCK, B., GATARIC, M., AND HANSEN, A. C. On stable reconstructions from univariate non-uniform fourier measurements. 2013.
2. ARCHIBALD, R., CHEN, K., GELB, A., AND RENAUT, R. Improving tissue segmentation of human brain MRI through pre-processing by the Gegenbauer reconstruction method. *NeuroImage* (2003).
3. ARCHIBALD, R., AND GELB, A. A method to reduce the Gibbs ringing artifact in mri scans while keeping tissue boundary integrity. *IEEE Medical Imaging* (2002).
4. CANNY, J. A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intell. 8* (1986), 678–698.
5. CHRISTENSEN, O. *An introduction to frames and Riesz bases*. Applied and Numerical Harmonic Analysis. Birkhäuser Boston Inc., Boston, MA, 2003.
6. FESSLER, J., AND SUTTON, B. Nonuniform fast Fourier transforms using min-max interpolation. *IEEE Trans. Sig. Proc. 51* (2003), 560–574.
7. FORSYTH, D., AND PONCE, J. *Computer Vision: A Modern Approach*. Prentice Hall, 2003.
8. GELB, A., AND CATES, D. Segmentation of images from Fourier spectral data. *Communications in Computational Physics* (2009).
9. GELB, A., AND HINES, T. Detection of edges from nonuniform Fourier data. *Journal of Fourier Analysis and Applications 17* (2011), 1152–1179. 10.1007/s00041-011-9172-7.
10. GELB, A., AND HINES, T. Recovering exponential accuracy from nonharmonic fourier data through spectral reprojection. *J. Sci. Comput. 51*, 1 (2012), 158–182.
11. GELB, A., AND SONG, G. A frame theoretic approach to the non-uniform fast Fourier transform. 2013.
12. GELB, A., AND TADMOR, E. Detection of edges in spectral data ii. nonlinear enhancement. *SIAM J. Numer. Anal 38*, 1389–1408.
13. GELB, A., AND TADMOR, E. Adaptive edge detectors for piecewise smooth data based on the minmod limiter. *Journal of Scientific Computing 28*, 2-3 (2006), 279–306.
14. GELB, A., AND TANNER, J. Robust reprojection methods for the resolution of the Gibbs phenomenon. *Applied Computational and Harmonic Analysis 20* (2006), 3–25.
15. GOTTLIEB, D., AND ORSZAG, S. *Numerical Analysis of Spectral Methods: theory and applications*, vol. 26. Society for Industrial and Applied Mathematics, 1993.
16. JACKSON, J., MEYER, C., NISHIMURA, D., AND MACOVSKI, A. Selection of a convolution function for Fourier inversion using gridding [computerised tomography application]. *Medical Imaging, IEEE Transactions on 10*, 3 (1991), 473–478.
17. O'SULLIVAN, J. A fast sinc function gridding algorithm for Fourier inversion in computer tomography. *Medical Imaging, IEEE Transactions on 4*, 4 (1985), 200–207.
18. PETERSEN, A., GELB, A., AND EUBANK, R. Hypothesis testing for fourier based edge detection methods. *Journal of Scientific Computing 51* (2012), 608–630.
19. PIPE, J. G., AND MENON, P. Sampling density compensation in MRI: Rationale and an iterative numerical solution. *Magnetic Resonance in Medicine 41*, 1 (1999), 179–186.
20. PLATTE, R., AND GELB, A. A hybrid Fourier-Chebyshev method for partial differential equations. *Journal of Scientific Computing 39* (2009), 244–264.
21. STEFAN, W., VISWANATHAN, A., GELB, A., AND RENAUT, R. Sparsity enforcing edge detection method for blurred and noisy Fourier data. *J. Sci. Comput. 50*, 3 (2012), 536–556.
22. VISWANATHAN, A. *Imaging from Fourier Spectral Data: Problems in Discontinuity Detection, Non-harmonic Fourier Reconstruction and Point-spread Function Estimation*. PhD thesis, Arizona State University, 2010.
23. VISWANATHAN, A., GELB, A., COCHRAN, D., AND RENAUT, R. On reconstruction from non-uniform spectral data. *J. Sci. Comp. 45*, 1–3 (2010), 487–513.