# Rapid Gridding Reconstruction With a Minimal Oversampling Ratio

Philip J. Beatty*, Dwight G. Nishimura, *Member, IEEE*, and John M. Pauly, *Member, IEEE*

***Abstract*—Reconstruction of magnetic resonance images from data not falling on a Cartesian grid is a Fourier inversion problem typically solved using convolution interpolation, also known as gridding. Gridding is simple and robust and has parameters, the grid oversampling ratio and the kernel width, that can be used to trade accuracy for computational memory and time reductions. We have found that significant reductions in computation memory and time can be obtained while maintaining high accuracy by using a minimal oversampling ratio, from 1.125 to 1.375, instead of the typically employed grid oversampling ratio of two. When using a minimal oversampling ratio, appropriate design of the convolution kernel is important for maintaining high accuracy. We derive a simple equation for choosing the optimal Kaiser–Bessel convolution kernel for a given oversampling ratio and kernel width. As well, we evaluate the effect of presampling the kernel, a common technique used to reduce the computation time, and find that using linear interpolation between samples adds negligible error with far less samples than is necessary with nearest-neighbor interpolation. We also develop a new method for choosing the optimal presampled kernel. Using a minimal oversampling ratio and presampled kernel, we are able to perform a three-dimensional (3-D) reconstruction in one-eighth the time and requiring one-third the computer memory versus using an oversampling ratio of two and a Kaiser–Bessel convolution kernel, while maintaining the same level of accuracy.**

***Index Terms*—Convolution interpolation, gridding, kernel function, three-dimensional (3-D) reconstruction.**

## I. Introduction

In MAGNETIC RESONANCE IMAGING (MRI), the data are obtained in the spatial frequency domain, often called $k$-space. The image reconstruction problem can be posed as a Fourier inversion problem which can be solved efficiently using the fast Fourier transform (FFT) when the data lies on a Cartesian grid. When the data samples do not fall on a Cartesian grid, the MRI community has largely turned to the gridding algorithm due to its efficiency and accuracy [1], [2]. Although we focus primarily on MRI reconstruction, gridding can also be used in cone-beam CT reconstruction and ultrasound tomography [3], [4].

Manuscript received September 30, 2004; revised March 23, 2005. This work was supported by the National Institutes of Health (NIH) under Grant R01 HL 067161, Grant R01 EB002992, Grant P41 RR09784, Grant R01 HL 39297, and R01 HL 75803. The work of P. J. Beatty was supported by a William R. Hewlett Stanford Graduate Fellowship. The Associate Editor responsible for coordinating the review of this paper and recommending its publication was D. Rueckert. *Asterisk indicates corresponding author.*

*P. J. Beatty is with the Magnetic Resonance Systems Research Laboratory, Department of Electrical Engineering, Stanford University, Packard 210, 350 Serra Mall, Stanford, CA 94305 USA (e-mail: pbeatty@mrsrl.stanford.edu).

D. G. Nishimura and J. M. Pauly are with the Magnetic Resonance Systems Research Laboratory, Stanford University, Stanford, CA 94305 USA.

Digital Object Identifier 10.1109/TMI.2005.848376

The gridding algorithm can be divided into two stages: sampling density compensation and inverse Fourier transform; in this paper we consider the later stage only, which is equivalent to the nonuniform FFT (NUFFT) problem type one [5]. Good treatments of density compensation are given by Pipe and Menon and Rasche *et al.* [6], [7]. In the gridding algorithm, the inverse Fourier transform is estimated by convolving the density compensated data with a finite kernel, sampling onto a Cartesian grid, performing an FFT and multiplying by an apodization correction function.

Reconstruction methods similar to gridding also exist, which do not separate the density compensation from the inverse Fourier transform. This can be accomplished by using shift-variant interpolators or iterative methods [8]–[10]. Shift-variant interpolators require a significant precomputation stage and enough memory to hold a unique set of weights for each data sample. When enough memory is available and the same data sample locations are repeatedly reconstructed, as is the case in real-time two-dimensional (2-D) imaging, such methods are appropriate. This paper has been motivated by a desire to efficiently reconstruct three-dimensional (3-D) non-Cartesian data samples. In this case, the memory required to store precomputed weights for shift-variant interpolators can limit the size of the image we are able to reconstruct.

Also limiting is the use of a grid oversampling ratio of two, traditionally used in gridding reconstruction [11]. A grid oversampling ratio of two leads to an eight-times increase in memory required for a 3-D image. By using a minimal oversampling ratio, the required memory drops to 1.4–2.6 times the memory required for the final image. Using a minimal oversampling ratio also leads to a significant reduction in computation time due to the reduced size of the FFT that is performed. However, maintaining high accuracy on a minimally oversampled grid requires a well-suited convolution kernel; in Sections IV and VI, we develop two kernel design methods.

On the one hand, memory constraints make the use of precomputed weights for each data sample undesirable; on the other hand, evaluating the Kaiser–Bessel function several times for each data point can account for most of the computation time. Greengard and Lee have shown that using a Gaussian kernel can be quite efficient; in one dimension, the Gaussian kernel requires two exponential evaluations per data sample point plus two multiplications for each point on the grid to which this data sample is deposited [12]. We achieve a similar efficiency without being limited to a particular kernel shape by using a presampled kernel. Interpolation between these presampled values is used to approximate the continuous gridding kernel. For linear interpolation, interpolating between

two stored values involves one multiplication and one addition, similar to the two multiplications required for the Gaussian kernel. In some cases, the speed improvement is superior to precomputed weights since the presampled kernel can fit easily in the computer cache. We examine both nearest-neighbor and linear interpolation of the presampled kernel and develop expressions for how finely the kernel must be sampled so as not to degrade the reconstruction.

Ideally, we would like the reconstruction computation time to be as short as possible while being confident that no reconstruction artifacts are noticeable in the final image. We define the aliasing amplitude, a metric for gridding accuracy, which allows us to analyze how the grid oversampling ratio, kernel width, and kernel sampling affect the reconstruction accuracy. By choosing gridding parameter combinations that have a low aliasing amplitude, we can have great confidence in the fidelity of our reconstruction. For the aliasing amplitude requirements of MR image reconstruction, parameter combinations with minimal oversampling ratios are much more computationally expedient than combinations which use an oversampling ratio of two. For 3-D images, the memory constraint of the reconstruction computer might also have to be taken into account in the selection of the oversampling ratio. We confirm our analysis by comparing images obtained using gridding to images obtained using an exact inverse Fourier transform.

## II. The Gridding Algorithm

For the rest of this paper, we shall assume that the data has already been density compensated and we will use "gridding" to refer to the inverse Fourier transform step of the reconstruction. The basic gridding algorithm is straightforward, consisting of the following steps.

1) Convolve the $k$-space samples with a gridding kernel.
2) Sample the result onto evenly spaced sample locations (the grid).
3) Perform an FFT (we use fftw) [13].
4) Multiply by an apodization correction function.

Throughout this paper, we treat the 1-D case for pedagogical simplicity; it is easy to extend the result to two and three dimensions when separable gridding kernels are used. We define the variables and functions used throughout this paper in Tables I and II. In addition, we define our image to have unit resolution, so the image pixel locations are at $-N/2, -N/2+1, \ldots, N/2-1$, for even $N$, and we sample $k$-space at $G$ evenly spaced points in the range $[-0.5, 0.5]$, since we measure $k$-space in *cycles* per unit distance. With these conventions, the gridding algorithm can be expressed mathematically as

$$\hat{m}_s[i] = \mathrm{IFT}\{[M_s(k_x) * C(k_x)]\text{Ш}(Gk_x)\}(i)\frac{1}{c(i)}. \quad (1)$$

Note that the apodization correction function $1/c(i)$ is the reciprocal of the inverse Fourier transform of the kernel $C(k_x)$ and is determined completely by the choice of kernel. We would like gridding to perform the mathematical operations given in (1), however, in practice, we use the FFT algorithm, not a continuous inverse Fourier transform. Because of this, we only compute $\hat{m}_s[i]$ for integer values of $i$—that is, we compute image values at discrete pixel locations. As well, we must remember one caveat: if, due to the width of the kernel, $[M_s(k_x) * C(k_x)]\text{Ш}(Gk_x)$ extends outside of the range $[-0.5, 0.5]$, the values outside of this range must be added to the opposite edge of the grid. This is because sampling in the image domain is equivalent to a repetition in $k$-space.

The computation time is the sum of two terms. The first term is due to the convolution and sampling step of the algorithm and is proportional to $DW^2$ for 2-D images and $DW^3$ for 3-D images, where $D$ is the number of data points and $W$ is the width

TABLE I
General Variables and Functions

| | |
|---|---|
| $D$ | Number of $k$-space data points. |
| $N$ | Width of the image, pixels. |
| $G$ | Number of grid sample locations in $k$-space. (samples are $1/G$ inverse pixels apart) |
| $\alpha = G/N$ | Grid oversampling ratio. |
| $k_x$ | $k$-space location, $k_x \in \mathbf{R}$ |
| $x$ | image location, $x \in \mathbf{R}$ |
| $i$ | pixel location, $i \in \{-\frac{N}{2}, -\frac{N}{2}+1, \ldots, \frac{N}{2}-1\}$ |
| $M_s(k_x)$ | Sampled, density corrected $k$-space data. $M_s(k_x) = \sum_{i=1}^{D} M_d(k_{x,i})\delta(k_x - k_{x,i})$ |
| $M_d(k_{x,i})$ | Density corrected $k$-space datum value at $k_{x,i}$. |
| $m_s(x)$ | Inverse Fourier transform of $M_s(k_x)$. $m_s(x) = \sum_{i=1}^{D} M_d(k_{x,i}) \exp(j2\pi k_{x,i} x)$ |
| $\widehat{m}_s[i]$ | Gridding estimate of $m_s(x = i)$. |
| $\varepsilon[i]$ | Aliasing amplitude. |
| $E[i]$ | Image reconstruction error. $E[i] = \lvert m_s(i) - \widehat{m}_s[i] \rvert$ |
| $\text{Ш}(x)$ | Shah function, $\text{Ш}(x) = \sum_{p=-\infty}^{\infty} \delta(x-p)$ |
| $\mathrm{FT}\{f(x)\}(k_x)$ | Fourier transform $\mathrm{FT}\{f(x)\}(k_x) = \int_{-\infty}^{\infty} f(x)\exp(-j2\pi k_x x)dx$ |
| $\mathrm{IFT}\{f(k_x)\}(x)$ | Inverse Fourier transform. $\mathrm{IFT}\{f(k_x)\}(x) = \int_{-\infty}^{\infty} f(k_x)\exp(j2\pi k_x x)dk_x$ |

N.B. Parenthesis, (), are used for functions that take real arguments. Square brackets, [], are used for integer arguments.

TABLE II
Gridding Kernel Variables and Functions

| | |
|---|---|
| $C(k_x)$ | Continuous valued kernel function. (e.g. Kaiser-Bessel, truncated sinc, Gaussian) |
| $c(x)$ | Inverse Fourier transform of $C(k_x)$. |
| $W$ | Width of kernel in grid units. (one grid unit is $1/G$, the distance between samples) |
| $S$ | Kernel sampling density. (number of kernel samples per grid unit) |
| $C_s(k_x)$ | Sampled kernel. $C_s(k_x) = C(k_x)\,\text{Ш}(SGk_x)$ |
| $c_s(x)$ | Inverse Fourier transform of $C_s(k_x)$. |
| $H(k_x)$ | Interpolation function. $H(k_x) = \sqcap(SGk_x)$ for nearest neighbor interpolation. $H(k_x) = \wedge(SGk_x)$ for linear interpolation. |
| $h(x)$ | Inverse Fourier transform of $H(k_x)$. |
| $\widehat{C}(k_x)$ | Continuous valued kernel after sampling and interpolation. $\widehat{C}(k_x) = C_s(k_x) * H(k_x)$ |
| $\widehat{c}(x)$ | Inverse Fourier transform of $\widehat{C}(k_x)$. $\widehat{c}(x) = c_s(x)h(x)$ |

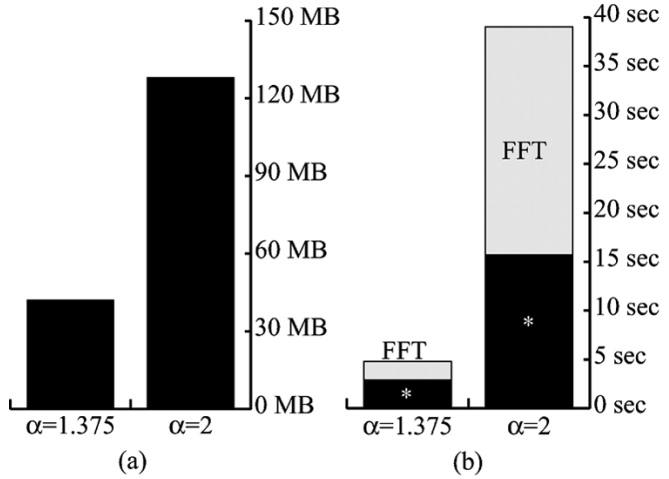


Fig. 1. Reconstruction of a $128 \times 128 \times 128$ image from the 3-D cones trajectory with 2 304 000 points [14]. With a minimal oversampling ratio of 1.375, we used a sampled Kaiser–Bessel kernel, $W = 5$, with linear interpolation. With the typical oversampling ratio of two, we used a Kaiser–Bessel kernel, $W = 4$. Both parameter choices have similar aliasing amplitudes, resulting in similar reconstruction errors (Fig. 10). (a) Memory requirements. (b) Reconstruction time on PC with one 1.67-GHz AMD processor. $*$ refers to the convolution and sampling step of the gridding algorithm. This demonstrates the large gains that can be achieved, especially in 3-D image reconstruction, by using a minimal oversampling ratio and a presampled gridding kernel.

of the kernel on the oversampled grid. We can reduce this term by shrinking $W$ and by reducing the constant of proportionality. The second term is computing the FFT. For grid oversampling ratio $\alpha$ and image width $N$, it is proportional to $\alpha^2 N^2 \log \alpha N$ for 2-D images and $\alpha^3 N^3 \log \alpha N$ for 3D images and can therefore be reduced by choosing a smaller oversampling ratio. As the bulk of the computer memory requirement is in storing the grid, which is of size $\alpha^2 N^2$ for 2-D images and $\alpha^3 N^3$ for 3-D images, a smaller $\alpha$ also reduces our memory requirements.

When we move to a minimal oversampling ratio to decrease the computation time of the FFT, if we wish to maintain the same accuracy, we must increase the width of the kernel, thereby *increasing* the computation time of the convolution. When the kernel is not presampled, the speed gain due to a smaller FFT can be negated by the increased computation of the convolution. This is why using a presampled kernel is important when using a minimal oversampling ratio: it speeds up the convolution, keeping it from dominating the reconstruction time. By using a minimal oversampling ratio and a presampled kernel, both the convolution and FFT stages of reconstruction are considerably reduced, without loss of accuracy. Fig. 1 shows that we obtain an eight times reduction in computation time for a $128 \times 128 \times 128$ 3-D reconstruction. Some further computation reductions can be obtained at the expense of accuracy by decreasing $\alpha$ and $W$, although there is a limit, since we cannot go below an oversampling ratio of one and interpolation must take some positive amount of time.

## III. Aliasing Amplitude: A Metric for Gridding Accuracy

The goal of the gridding algorithm is to find the inverse Fourier transform of $M_s(k_x)$. Put simply, we would like $\hat{m}_s[i] = m_s(x)$ when $x = i$ and the extent to which this is not true is the extent to which gridding is inaccurate. For a given data acquisition, we can calculate the image reconstruction error $E[i] = |m_s(i) - \hat{m}_s[i]|$ where $m_s(i)$ is calculated directly using the sum in Table I and $\hat{m}_s[i]$ is found by performing the gridding reconstruction. It should be noted that although the direct summation method computes an exact inverse Fourier transform, it takes an excessive amount of computation.

While the image reconstruction error gives the inaccuracy of the gridding algorithm for a certain object and trajectory, it offers little insight into the causes of this error and little guidance in designing a more suitable gridding kernel. To gain some insight into the causes of this error, we rewrite (1) completely in the image domain

$$\hat{m}_s[i] = \left\{ [m_s(i)c(i)] * \left[ \frac{1}{G} \text{Ш}\left( \frac{i}{G} \right) \right] \right\} \frac{1}{c(i)}. \tag{2}$$

The sampling onto the grid in $k$-space causes a repetition of $m_s(x)c(x)$ in image space every $G$ pixels. These shifted versions of $m_s(x)c(x)$ alias back into the image, accounting for the error in the gridding estimate. Using (2), we can rewrite the image reconstruction error as

$$\begin{aligned} E[i] &= |m_s(i) - \hat{m}_s[i]| \\ &= \left| \frac{1}{c(i)} \sum_{p \neq 0} m_s(i + Gp) c(i + Gp) \right|. \end{aligned}$$

Since the error depends on $m_s(x)$, it depends on the $k$-space sampling pattern and the object being imaged. As we try to design gridding solutions to have low errors over a range of sampling patterns and objects, it is most helpful to have a metric which is independent of $m_s(x)$, but gives an estimate of the reconstruction errors we can expect. For this purpose, Jackson *et al.* used the relative amount of aliasing energy, which is independent of the image and sampling pattern used and provides a good relative ordering of accuracy [11]. However, the aliasing energy metric used by Jackson has a couple of shortcomings. Since Jackson averaged the aliasing energy over the entire image, we do not get an idea of how the error will vary from position to position in the image. As well, we desire a metric which will predict, at least in terms of the order of magnitude, what we can expect for the image reconstruction error when using a given kernel. By taking the square root of the aliasing energy at each pixel location, we obtain a metric that is both a function of position and can be used to estimate $E[i]$. We call this metric the *aliasing amplitude*, $\varepsilon[i]$.

One way to view this metric is to model $m_s(x)$ as a Gaussian distributed random complex number, independent along $x$, with unit variance. In this case, the expected value of $\hat{m}_s[i]$ is $m_s(i)$ and the standard deviation of location $i$ in the image is identical to the aliasing amplitude at this pixel location

$$\varepsilon[i] = \sqrt{\frac{1}{[c(i)]^2} \sum_{p \neq 0} [c(i + Gp)]^2}. \tag{3}$$

The aliasing amplitude gives an order-of-magnitude estimate for the image reconstruction error. That is, although the error in a reconstructed image is dependent on the object and $k$-space sampling pattern, we can reliably predict the order of magnitude and profile of the error from the aliasing amplitude, as shown

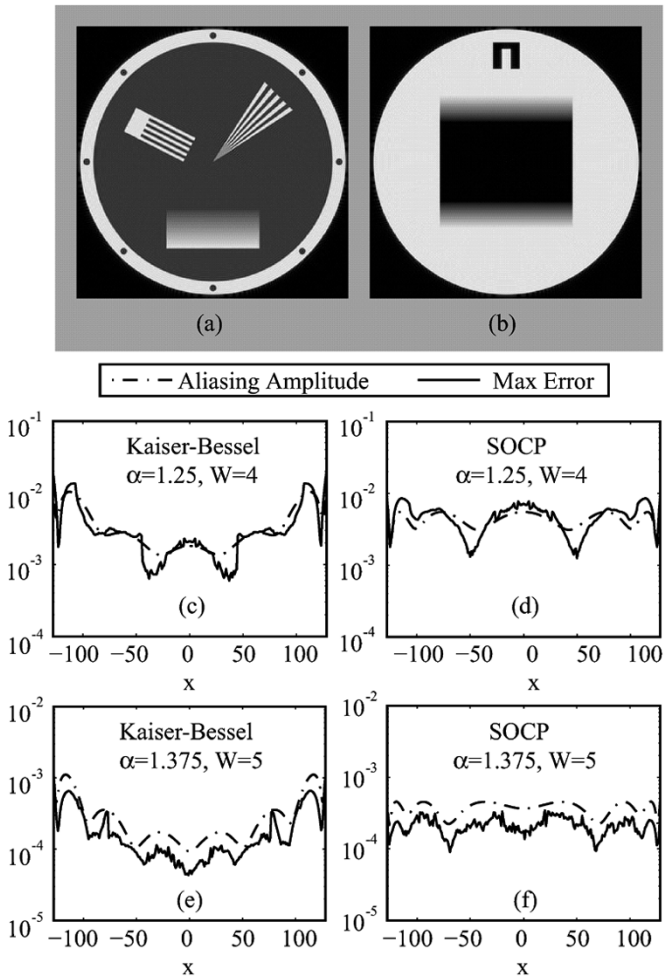


Fig. 2. The aliasing amplitude $\varepsilon[i]$ is a good predictor for the image reconstruction error $E[i]$. Sampling the two numerical phantoms (a) and (b) with both radial and spiral $k$-space trajectories, we obtained four data sets. In order to look at the 1-D error pattern, we used a very low error method in the $y$-direction (oversampling ratio $\alpha = 2$, with a Kaiser–Bessel kernel of width $W = 8$). In the $x$-direction we used the oversampling ratio and kernel shown in (c)–(f), where SOCP refers to the optimal sampled kernels described in Section VI. For each of the kernels in the $x$-direction, we reconstructed four $256 \times 256$ images (two phantoms $\times$ two trajectories) and subtracted them from the ideal reconstructions using the exact inverse Fourier transform. Taking the absolute value we obtained the image reconstruction error $E[i]$. This gave us 1024 error values (4 images, 256 lines per image) at each $x$-location for each kernel. The *Max Error* is the maximum of these 1024 values at each $x$ location. This is very close to the aliasing amplitude plot, showing that the aliasing amplitude is a good indicator of the order of magnitude of the maximum reconstruction error we are likely to see.

in Fig. 2. This allows us to design kernels using the aliasing amplitude as a metric for accuracy.

Fig. 3 provides a reference of the maximum aliasing amplitude of an image for various oversampling ratios and kernel widths. With the typically low signal-to-noise ratio (SNR) of MR images, an aliasing amplitude of 0.1 will likely suppress visible artifacts, allowing us to use a kernel of width three on a grid with an oversampling ratio of 1.125. However, in cases of high dynamic range, or when the region of reconstruction is smaller than the extent of the object, more accuracy may be needed. To cover such cases, an oversampling ratio of 1.25 with a width four kernel will reduce the maximum aliasing amplitude to under 0.01. A maximum aliasing amplitude under 0.001 can

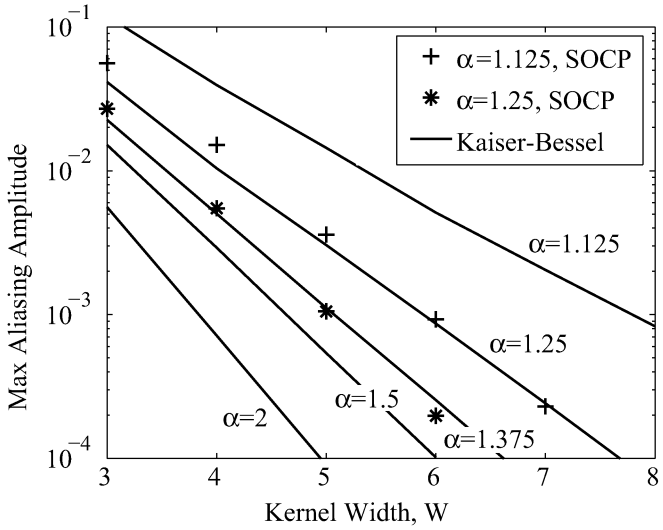


Fig. 3. Maximum aliasing amplitude for a selection of kernels. The shape parameter for the Kaiser–Bessel kernels is found using (5). SOCP refers to the optimal sampled kernels found using second-order cone programming, Section VI.

be achieved with an oversampling ratio of 1.375 and a width five kernel, giving higher accuracy than is likely necessary in MRI with a much lower computation time than can be achieved using an oversampling ratio of two.

In Fig. 4, we compare the accuracy of the parameter combinations with aliasing amplitudes of 0.1 and 0.01 in reconstructing an in vivo sagittal knee image. Although using a parameter combination with an aliasing amplitude of 0.1 performs quite well, it is possible in some cases to see slight degradations in reconstruction quality. This degradation is not apparent when the aliasing amplitude is 0.01 or lower.

## IV. Choosing a Kaiser–Bessel Kernel

One of the difficulties in designing a gridding solution is that a different gridding kernel must be chosen for each choice of oversampling ratio $\alpha$ and kernel width $W$. For example, using a kernel designed for $\alpha = 2$ on a grid with $\alpha = 1.375$ can lead to a large loss in accuracy, as shown in Fig. 5. The error images in Fig. 5(b) and (d) show that the large errors in Fig. 5(a) and (c) can be avoided by designing the kernel for the oversampling ratio used. In the case that the errors in Fig. 5(a) and (c) are acceptable, we would do better to design for this target accuracy and decrease our computation time.

Jackson *et al.* examined many different functions that could be used for gridding kernels and found the Kaiser–Bessel window preferable since it is easily computable and offers near optimal performance [11]. Jackson also calculated the optimal shape parameter for a range of $W$ for oversampling ratios of one and two, as shown in Table III. The shape parameter, $\beta$, determines the shape of the Kaiser–Bessel window. Here, we derive a simple equation which gives near optimal $\beta$ for any $\alpha \in [1, 2]$. A similar equation for $\beta$ has been developed by Wajer *et al.*, but this equation does not work well for minimal oversampling ratios [15]. It should be noted that our metric for optimality of the kernel is slightly different than that used by Jackson, accounting for the discrepancy in $\beta$ values in Table III.

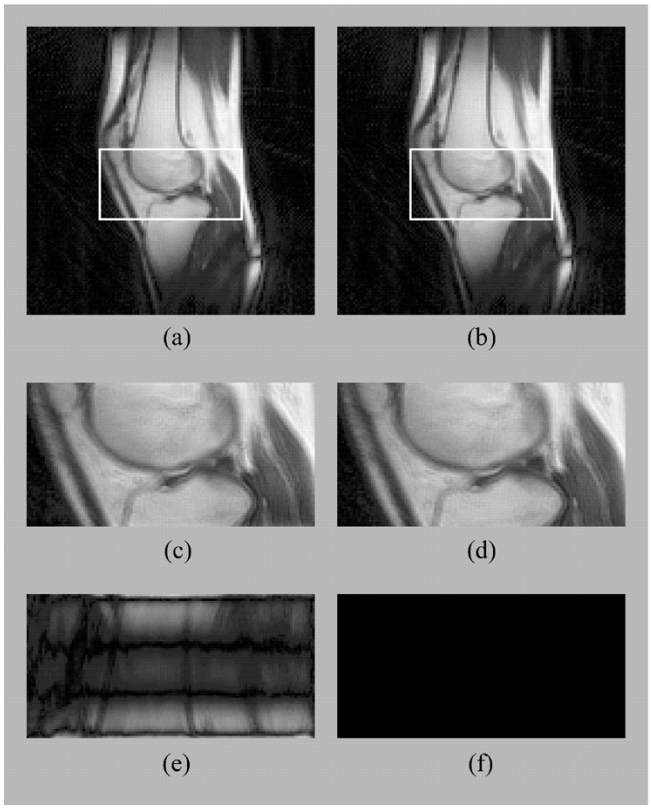


Fig. 4. Sagittal knee images obtained with a 2-D spiral sequence. (a), (c) were reconstructed using $\alpha = 1.125$ and $W = 3$. (b), (d) were reconstructed using $\alpha = 1.25$ and $W = 4$. When the full field-of-view is reconstructed in (a) and (b), no difference in the images is discernible. A more challenging test is to reconstruct a region of interest which is smaller than the extent of the object (c), (d). (e), (f) are error images of (c), (d) windowed up 10 *times*. In this case, the $\alpha = 1.125$ method gives a slight loss of visible accuracy, whereas the $\alpha = 1.25$ method is still exceedingly accurate.

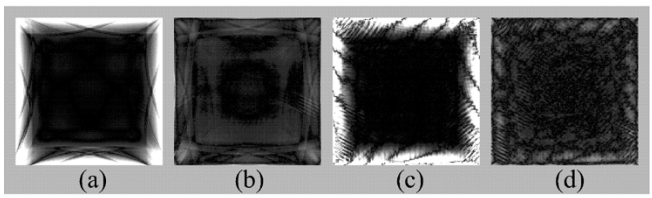


Fig. 5. The $k$-space data for the numerical phantom in Fig. 2(a) was acquired using a radial trajectory in (a) and (b) and a spiral trajectory in (c) and (d). We reconstructed these two data sets on a grid with grid oversampling ratio $\alpha = 1.375$ using two different Kaiser–Bessel kernels, both of width $W = 5$. In (a) and (c), the shape parameter $\beta$ is 11.4410, the shape parameter appropriate when the grid oversampling ratio is two (which it is not in this case). In (b) and (d), the shape parameter is 9.5929, the shape parameter appropriate for the grid oversampling ratio $\alpha = 1.375$ (5). Shown is the image reconstruction error windowed up 1000 *times* relative to Fig. 2(a). By not choosing the kernel appropriate for the oversampling ratio used, (a) and (c) have large reconstruction errors around the edges of the image.

Jackson searched for the kernel that gave the lowest *average* aliasing energy over the image whereas we search for the kernel that gives the lowest *maximum* aliasing amplitude at any point in the image. We refer to this as *optimal* in the rest of this paper. We choose this metric because we are interested in ensuring that the gridding reconstruction does not introduce any visible artifacts into the image. When the average aliasing energy is minimized, much of the aliasing energy is deposited around the edges of the image and the aliasing amplitude at these points is much higher than the average.

TABLE III
COMPARISON OF $\beta$ VALUES

| | $\beta$ | | | |
|---|---|---|---|---|
| | $\alpha = 1$ | | $\alpha = 2$ | |
| $W^a$ | Jackson | Eq. 5 | Jackson | Eq. 5 |
| 2 | 2.3934 | 1.4050 | | |
| 3 | 4.2054 | 3.7830 | 6.6875 | 6.4861 |
| 4 | 5.7567 | 5.6199 | 9.1375 | 8.9962 |
| 5 | 7.4302 | 7.3341 | 11.5250 | 11.4410 |
| 6 | | | 13.9086 | 13.8551 |
| 7 | | | 16.2734 | 16.2522 |
| 8 | | | 18.5547 | 18.6389 |

We use our kernel width parameter, $W$, which is equal to twice the window width parameter used by Jackson *et al.* for $\alpha = 2$.

A good convolution kernel acts as a multibandpass filter in the image domain with passband $x \in [-N/2, N/2]$ and stopbands $x \in [Gp - N/2, Gp + N/2]$, for all integer values of $p \neq 0$. Choosing a larger oversampling ratio allows for wider transition bands. The equations for the Kaiser–Bessel window and its inverse Fourier transform are given by

$$C(k_x) = \frac{G}{W} I_0(\beta\sqrt{1-(2Gk_x/W)^2}) \quad \text{for } |k_x| \leq \frac{W}{2G}$$
$$c(x) = \frac{\sin\sqrt{(\pi Wx/G)^2 - \beta^2}}{\sqrt{(\pi Wx/G)^2 - \beta^2}} \tag{4}$$

where $I_0(k_x)$ is the zero-order modified Bessel function of the first kind.

Intuitively, the best Kaiser–Bessel window is the one for which the central lobe of the inverse Fourier transform of the window, $c(x)$, extends to the edge of the first stopband, as shown in Fig. 6. A narrower central lobe leads to increased error through apodization of the passband whereas a wider central lobe allows the large central lobe into the first stopband. From (4), the first zero of the Kaiser–Bessel window is when $\sqrt{(\pi Wx/G)^2 - \beta^2} = \pi$. Setting $\beta = \pi\sqrt{W^2(\alpha - (1/2))^2/\alpha^2 - 1}$ will position this zero crossing at $x = G - N/2$, as shown in Fig. 6(a). It turns out that we can do slightly better if we allow some of the main lobe to enter into the first stopband, as illustrated by Fig. 6(b). In essence we allow the aliasing amplitude at the edge of the image to rise until it reaches the aliasing amplitude caused by the first sidelobe. The point at which this happens is difficult to solve analytically, however we have found that a slight modification of the equation we obtain by positioning the first zero crossing at $x = G - N/2$ works very well in practice. The modified equation is

$$\beta = \pi\sqrt{\frac{W^2}{\alpha^2}\left(\alpha - \frac{1}{2}\right)^2 - 0.8}. \tag{5}$$

Equation (5) has the advantage that it gives a $\beta$ value which is very close to the optimal $\beta$ value and is simple to calculate. Equation (5) can be used to generate a kernel which will ensure that the gridding solution is highly accurate, considering the $\alpha$ and $W$ selected. Once the $\alpha$ and $W$ have been selected, the only way we have of reducing the computation time is to reduce the constant of proportionality of the $DW^2$ term ($DW^3$ term for

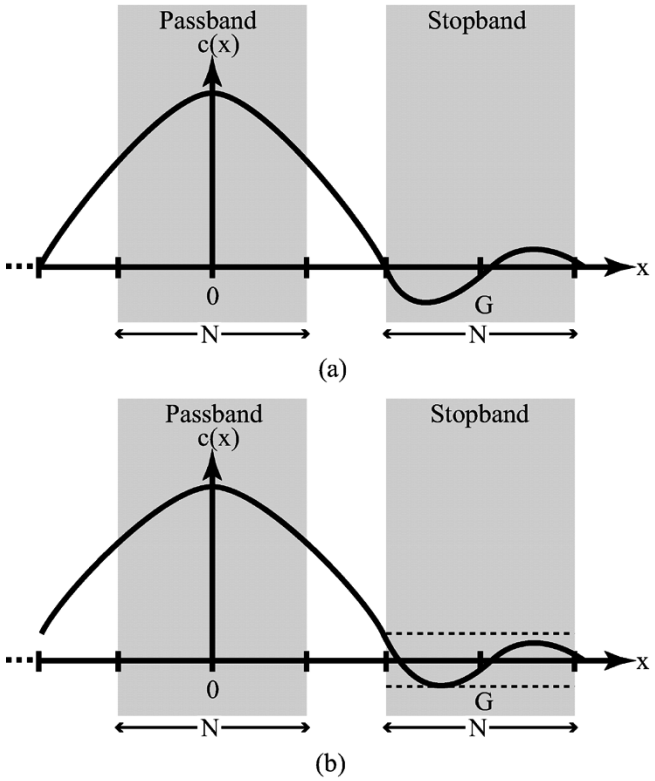


Fig. 6. (a) Positioning the first zero crossing at the edge of the first stopband results in a good kernel design, since it allows the main lobe to be as wide as possible without any of the main lobe contributing to the aliasing amplitude. A wide main lobe lessens apodization in the passband. Since apodization correction increases artifacts aliasing in from the stopbands as well as the passband signal, less apodization in the passband is preferable. (b) We can improve slightly on the kernel design in (a) by allowing the main lobe to enter slightly into the first stopband. In this case we try to keep the aliasing amplitude due to the main lobe in the first stopband equal to the aliasing amplitude caused by the first sidelobe.

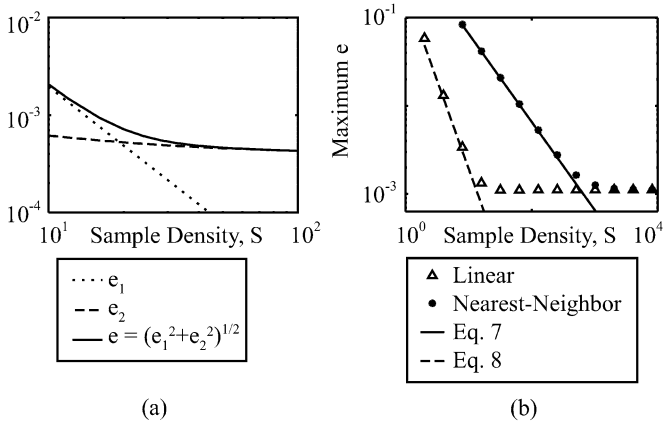


Fig. 7. (a) Dependence of the two components $\varepsilon_1$ and $\varepsilon_2$ of the aliasing amplitude on the kernel sampling density, $S$, for a sampled Kaiser–Bessel kernel with linear interpolation. The kernel width is $W = 5$ and $\beta$ was obtained using (5). Shown here is pixel position $i = 64$, with $N = 128, G = 176$ ($\alpha = 1.375$). Since $\varepsilon_2$ has a weak dependence on the kernel sampling density, once $\varepsilon_1$ drops below $\varepsilon_2$, little is gained by increasing the kernel sampling density. (b) Decrease in maximum aliasing amplitude as kernel sampling density increases. Shown for a sampled Kaiser–Bessel kernel with width, $W = 5$ on a grid with oversampling ratio $\alpha = 1.375$. When the kernel is undersampled, $\varepsilon \approx \varepsilon_1$ and the aliasing amplitude corresponds to (7) and (8). Once the kernel is sufficiently sampled, $\varepsilon \approx \varepsilon_2$, which has little dependence on the interpolation method or kernel sampling density. The advantage of linear interpolation is that it reaches the domain where $\varepsilon \approx \varepsilon_2$ with a much smaller kernel sampling density than using nearest-neighbor interpolation.

3-D). That is, we want to decrease the time it takes to deposit a data sample onto a grid point. The next section shows how this can be done by presampling the kernel and using interpolation.

## V. Presampling the Kernel

Even though the Kaiser–Bessel window is relatively inexpensive to calculate, evaluating it several times for each data point can significantly slow down the algorithm, often accounting for the majority of the computation time. A simple solution is to finely sample the kernel before starting the reconstruction process. We can then look up these sampled values and use nearest-neighbor or linear interpolation when a kernel value is needed. This amounts to replacing $C(k_x)$ with the sampled and interpolated kernel $\hat{C}(k_x)$ (Table II). While it is clear that this scheme offers a significant computation time savings, it is not clear how the aliasing amplitude is affected. We demonstrate that if not done carefully, this sampling can become the dominant source of inaccuracy in the gridding algorithm. We also show how to implement this scheme so that all of the computation time savings are realized without a noticeable loss in accuracy.

The aliasing amplitude of a sampled kernel is both a function of the sampling plus interpolation and the underlying kernel being sampled. With a low kernel sampling density, the dominant source of the aliasing amplitude is the interpolation function and kernel sampling density. As the kernel sampling density increases, the aliasing amplitude becomes more defined by the underlying kernel being sampled. This is easy to see if we consider the two extremes: sampling with one point and using linear interpolation, the resulting kernel is a triangle kernel regardless of the kernel function being sampled. Sampling with infinitely many points, the sampled kernel becomes the underlying kernel function itself and the aliasing amplitude has no dependence on the interpolation function. Appendix I provides a mathematical analysis of the aliasing amplitude of a sampled and interpolated kernel. Here we describe the results of that analysis and give a simple expression for ensuring that a kernel is sampled finely enough that its aliasing amplitude is that of the underlying kernel function being sampled.

One useful property of the aliasing amplitude of a sampled kernel is that it can be written as the quadrature sum of two terms

$$\varepsilon[i] = \sqrt{(\varepsilon_1[i])^2 + (\varepsilon_2[i])^2} \tag{6}$$

where the first term $\varepsilon_1$ is only a function of the interpolation function and sampling density. The second term is a function of the kernel being sampled, but is mainly independent of the interpolation function and sampling density, as shown in Fig. 7(a). Roughly, we can take $\varepsilon_1$ to be the aliasing amplitude due to sampling of the kernel and $\varepsilon_2$ to be the aliasing amplitude inherent in the choice of kernel. The effect of increasing the sampling density is to decrease the value of $\varepsilon_1$. Once $\varepsilon_1$ is an order of magnitude less than $\varepsilon_2$, $\varepsilon[i] \cong \varepsilon_2[i]$ and there is little benefit to

further increasing the kernel sampling density, also illustrated in Fig. 7(a).

The fundamental difference between nearest-neighbor and linear interpolation is how quickly $\varepsilon_1$ decreases as the kernel sampling density is increased. For both interpolation schemes, $\varepsilon_1$ is greatest at the edge of the image, when $i = -N/2$. For the two interpolation schemes this maximum value of $\varepsilon_1$ over the image is derived in Appendix I to be

$$\max(\text{nearest-neighbor } \varepsilon_1) = \frac{0.91}{\alpha S} \tag{7}$$

$$\max(\text{linear } \varepsilon_1) = \frac{0.37}{(\alpha S)^2}. \tag{8}$$

Since $\varepsilon_1$ is inversely proportional to $S$ for nearest-neighbor interpolation, a large kernel sampling density is needed to decrease $\varepsilon_1$ below the value of $\varepsilon_2$. For linear interpolation, $\varepsilon_1$ is inversely proportional to $S^2$, and requires a far smaller kernel sampling density for a comparable decrease in $\varepsilon_1$, as shown in Fig. 7(b).

We illustrate this point with a simple design example; we would like to design a sampled kernel to achieve a maximum aliasing amplitude of about $10^{-3}$ on a grid with oversampling ratio $\alpha = 1.25$. From Fig. 3, we see that this can be achieved with a Kaiser–Bessel kernel with $W = 6$. To ensure that sampling the kernel does not adversely affect the aliasing amplitude, we specify that $\max(\varepsilon_1)$ should be $10^{-4}$. For nearest-neighbor interpolation, this would require a kernel sampling density of 7280. With $W = 6$, this would entail sampling the kernel with 43 680 points. Using linear interpolation, the sampling density would need to be at least 49, requiring only 294 points for a kernel width of six.

Using a nearest-neighbor interpolation kernel or a triangle kernel as the gridding kernel are special cases of presampled kernels, when $S = 1$. In both of these cases, $\varepsilon = \varepsilon_1$ and (7) and (8) can be used to predict the kernel accuracy, as shown in Fig. 8.

We recommend using linear interpolation since the number of samples required to reduce the aliasing amplitude due to nearest-neighbor sampling can be very large. As well, it should be noted that in terms of computation time and memory it is preferable to lose accuracy through choosing a smaller oversampling ratio or kernel width than through overly coarse kernel sampling. Fig. 9 highlights this concept. With an oversampling ratio of two and a kernel width of six, sampling with 3000 points and using nearest-neighbor interpolation Fig. 9(c) leads to a significant increase in error over the unsampled kernel Fig. 9(a) and over sampling with 600 points and using linear interpolation Fig. 9(b). If this level of error is acceptable for the given application, a computation time and memory improvement can be seen by reconstructing using an oversampling ratio of 1.375, a kernel width of five, sampling with 300 points and using linear interpolation, Fig. 9(d), since this has the same level of error.

For the Kaiser–Bessel kernel, (4) can be used for apodization correction. When the kernel is sampled and interpolated, the apodization correction function should change accordingly. In Appendix II, we give an accurate method for determining the apodization correction from the kernel sample values.

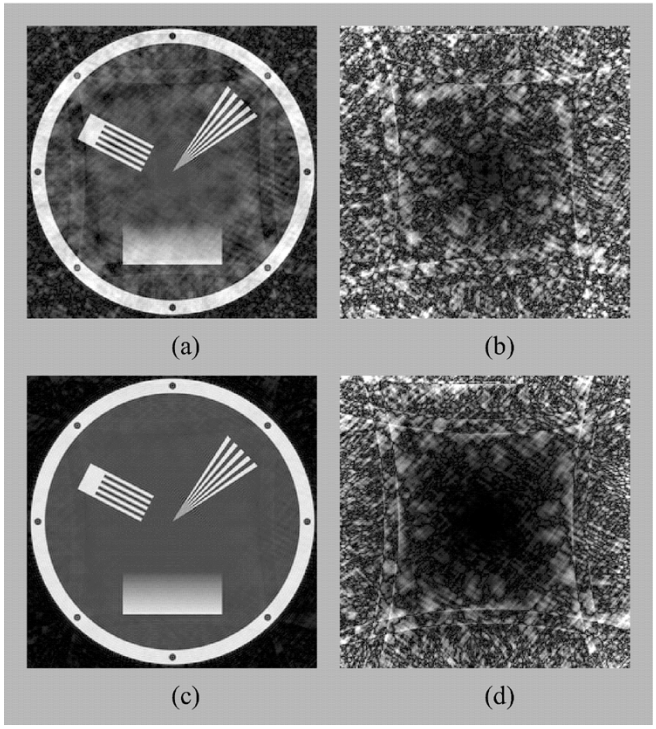


Fig. 8. Two common convolution kernels, the nearest-neighbor kernel and the triangle kernel, are simple cases of presampled and interpolated kernels. In both of these cases, the kernel sampling density, $S = 1$. From (12) in Appendix I, when $S = 1, \varepsilon_2 = 0$, and so $\varepsilon = \varepsilon_1$. (a) Reconstruction with nearest-neighbor interpolation kernel on a grid with oversampling ratio, $\alpha = 2$. (b) Error in (a) windowed up 5 *times*. (c) Reconstruction with a triangle kernel, width $W = 2$, on a grid with oversampling ratio, $\alpha = 2$. (d) Error in (c) windowed up 25 *times*. These errors agree well with the aliasing amplitude. From (7) and (8), the maximum error for nearest-neighbor kernel should be about 5 times that of the triangle kernel. Comparing (b) and (d) shows that this is the case.

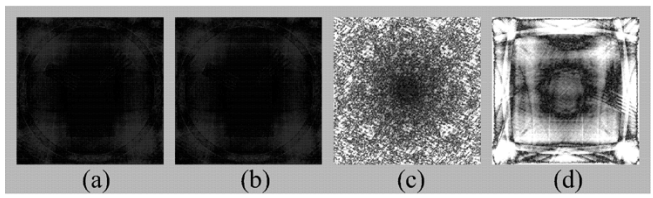


Fig. 9. Numerical phantom in Fig. 2(a) acquired with a radial trajectory. We show the image reconstruction error resulting from different gridding parameters. (a) $\alpha = 2$, Kaiser–Bessel kernel, $W = 6$. (b) Same parameters as (a), but now we sample the kernel with 600 points and use linear interpolation. This gives no increase in the reconstruction error. (c) Same kernel as (a), but here we sample the kernel with 3000 *points* and use nearest-neighbor interpolation. Even though we sample with far more points, using nearest-neighbor interpolation increases the image reconstruction error. (d) $\alpha = 1.375$, Kaiser–Bessel kernel, $W = 5$, sampled with 300 points and linearly interpolated. Compared to the method in (c), (d) uses a smaller oversampling ratio, a narrower kernel, and fewer kernel samples making it much quicker to compute. However, it still has the same level of reconstruction error. All image reconstruction error images have been windowed up 10 000 *times* relative to Fig. 2(a).

## VI. Choosing the Optimal Sampled Kernel

Thus far, we have obtained the sampled kernel by sampling a continuous valued kernel, like the Kaiser–Bessel window. By sampling finely enough and through proper interpolation, the sampled kernel can achieve the same accuracy as its continuous valued template. However, this is not the only way to design a sampled kernel. Starting with a certain kernel width and kernel

sampling density, we can design the sample values directly to achieve the minimum value for the maximum aliasing amplitude for a given oversampling ratio.

Once we have chosen the interpolation method, kernel sampling density and oversampling ratio, $\varepsilon_1$ is fixed and modifying the sample values can only reduce $\varepsilon_2$. Because of this, the problem can be written as $\min_{C_s(k_x)} \max(\varepsilon_2)$, where $C_s(k_x)$ is the sampled kernel. Equation (12) in Appendix I gives the following expression for $\varepsilon_2$ in terms of the inverse Fourier transform of $C_s(k_x)$

$$\varepsilon_2[i] = \sqrt{\sum_{p=1}^{S-1} \left[ \frac{\hat{h}(i+Gp)c_s(i+Gp)}{h(i)c_s(i)} \right]^2}$$

where $\hat{h}(x)$ is a simple function of $h(x)$ as shown in Appendix I. From this equation, we can write the following second-order cone program (SOCP) [16]

$$\begin{aligned} &\text{find } C_s(k_x) \\ &\text{subject to } \sqrt{\sum_{p=1}^{S-1} [\hat{h}(i+Gp)c_s(i+Gp)]^2} \le \varepsilon_2^{\max} h(i)c_s(i) \\ &\text{and} \quad c_s(i) > 0 \\ &\qquad \text{for } i \in \left\{-\tfrac{N}{2}, -\tfrac{N}{2}+1, \ldots, N2-1\right\}. \end{aligned}$$

This SOCP finds a kernel $C_s(k_x)$ if it exists, such that the maximum value of $\varepsilon_2[i] \le \varepsilon_2^{\max}$. See Appendix III for a MATLAB implementation of this SOCP. By decreasing $\varepsilon_2^{\max}$ until the smallest value is found where a solution exists, we are able to obtain the optimal sampled kernel.

The disadvantage of this method is that a series of SOCP problems must be solved for every $\alpha, W$, and $S$ that is to be used. Each solution can take a couple of minutes to compute. However, once the solution is computed, the sample values can be stored in a file for future use. The advantage of this method is that it does provide a slightly lower maximum aliasing amplitude than the Kaiser–Bessel window. As well, it tends to distribute the aliasing amplitude more evenly over the entire image as shown in Figs. 2 and 10.

For the examples given in Fig. 10, the maximum reconstruction error of the optimal sampled kernel Fig. 10(d), (h) is 75% of the maximum reconstruction error of the comparable *unsampled* Kaiser–Bessel kernel Fig. 10(b), (f). This leads to the surprising notion that sampling the kernel can actually lead to lower errors in practice than if we had not sampled the kernel. This is not an inherent advantage of sampled kernels, since a sampled and interpolated kernel is a special type of continuous kernel. Rather, sampling the kernel has allowed us to use the *design* method developed above, which gives a kernel design not readily available when a continuous approach is taken.

## VII. Conclusion

In this paper, we bring together a number of technical refinements to the basic gridding algorithm. These include using a minimal oversampling ratio, presampling the kernel, and two methods for designing high performance gridding kernels. We have placed all of these developments in a consistent framework and developed the aliasing amplitude as a useful metric for evaluating the accuracy of the gridding algorithm.

These technical improvements complement each other, giving an accurate and efficient method for reconstructing MR images when the data is acquired using non-Cartesian trajectories. The computation time and memory savings are especially significant in 3-D imaging, where we see an eightfold reduction in computation time and a threefold reduction in memory requirements.

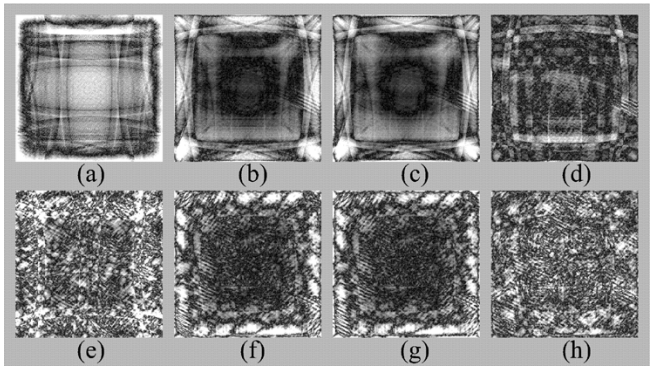


Fig. 10. Comparison of the image reconstruction error for four different kernels. (a)–(d) used a radial acquisition and (e)–(h) used a spiral acquisition; the object sampled is shown in Fig. 2(a). In the first column, a width four Kaiser–Bessel kernel was used on a grid with an oversampling ratio of two. In the second column a width five Kaiser–Bessel kernel was used on an $\alpha = 1.375$ grid, showing a similar error level. In the third column, the kernel used in (b), (f) was sampled with 300 points and linear interpolation was used, not changing the error noticeably. In the last column, the kernel was also of width five, sampled with 300 points and linearly interpolated, but the optimal sampled kernel was used (Section VI). The optimal sampled kernel distributes the errors more evenly over the image and results in a lower maximum error. All reconstruction error images are windowed up 5000 *times* relative to Fig. 2(a).

## Appendix I
## Aliasing Amplitude of a Sampled Kernel

When the kernel is sampled and interpolated in $k$-space, its inverse Fourier transform in image space can be written as a periodic term, $c_s(x)$, of period $SG$ modulated by an envelope function, where the envelope function is the inverse Fourier transform of the interpolation function. This is shown in Fig. 11. The periodicity of $c_s(x)$ allows us to write the infinite sum in (3) as a finite sum and thus quickly calculate the aliasing amplitude for a sampled kernel.

Starting from (3), we derive the aliasing amplitude for a sampled interpolated kernel, $c(i) = c_s(i)h(i)$, as follows:

$$\begin{aligned} \varepsilon[i] &= \sqrt{\frac{1}{c(i)^2} \sum_{p \ne 0} c(i+Gp)^2} \\ &= \sqrt{\sum_{p} \left[ \frac{c(i+Gp)}{c(i)} \right]^2 - 1}. \end{aligned}$$

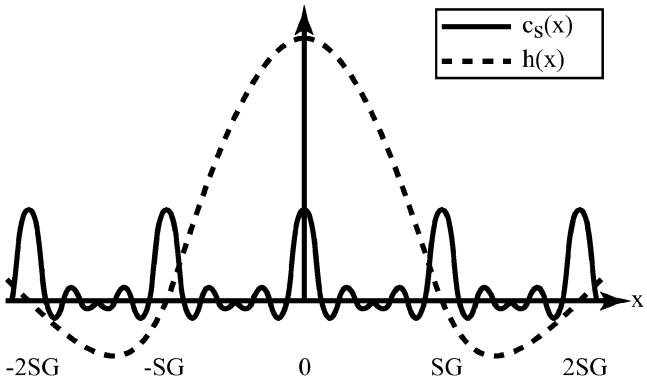


Fig. 11. The sampled and interpolated kernel, $\hat{c}(x)$, can be viewed as the periodic function $c_s(x) = c(x) * \text{Ш}(SGx)$, with period $SG$, modulated by the envelope function $h(x)$, which is a sinc function for nearest-neighbor interpolation and a sinc-squared function for linear interpolation.

We now group the terms of $c(i)$ separated by $SG$; the periodicity of $c_s(x)$ allows us to simplify $c(i+SG) = c_s(i+SG)h(i+SG) = c_s(i)h(i+SG)$

$$\varepsilon[i] = \sqrt{\sum_{p=0}^{S-1}\left[\frac{\hat{h}(i+Gp)c_s(i+Gp)}{h(i)c_s(i)}\right]^2 - 1} \tag{9}$$

where

$$\hat{h}(i) = \sqrt{\sum_q [h(i+SGq)]^2}.$$

Computing $\hat{h}(x)$ is straightforward, realizing that

$$\hat{h}(i) = \sqrt{\text{FT}\left\{[h(x)]^2 \text{Ш}\left(\frac{x-i}{SG}\right)\right\}(0)}. \tag{10}$$

Noting that the $p = 0$ term in (9) is independent of $c_s(x)$, we can separate $\varepsilon(i)$ into the quadrature sum of two terms as in (6), where

$$\varepsilon_1[i] = \sqrt{\left[\frac{\hat{h}(i)}{h(i)}\right]^2 - 1} \tag{11}$$

$$\varepsilon_2[i] = \sqrt{\sum_{p=1}^{S-1}\left[\frac{\hat{h}(i+Gp)c_s(i+Gp)}{h(i)c_s(i)}\right]^2}. \tag{12}$$

### A. Nearest-Neighbor Interpolation

For nearest-neighbor interpolation, $h(x) = \text{sinc}(x/SG)$. Substituting this into (10) and (11), $\hat{h}(i) = 1$ and

$$\varepsilon_1[i] = \sqrt{\frac{1}{\text{sinc}^2(i/SG)} - 1}.$$

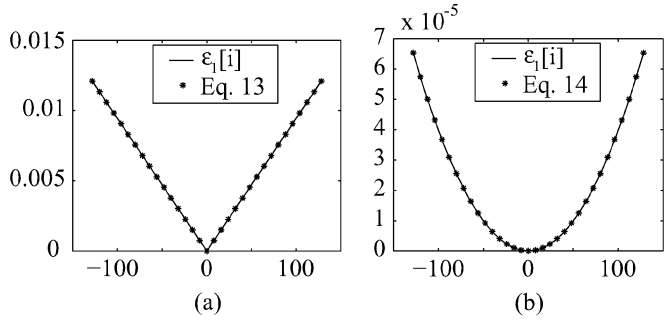


Fig. 12. Although (13) and (14) are approximations for $\varepsilon_1[i]$, they are very accurate approximations. In this example, $N = 256, \alpha = 1.25$, and $S = 60$. (a) Agreement for nearest-neighbor interpolation. Equation (13) is within 44 parts-per-million of $\varepsilon_1[i]$ in this case. (b) Agreement for linear interpolation. Equation (14) is within 209 parts-per-million of $\varepsilon_1[i]$ in this case.

Since $i \ll SG$, we can approximate $1/\text{sinc}^2(i/SG) - 1$ as $\pi^2 i^2/3S^2G^2$, using the first nonzero term of the Taylor series about $i = 0$. Thus

$$\text{nearest-neighbor } \varepsilon_1[i] \approx \frac{\pi|i|}{\sqrt{3}SG}. \tag{13}$$

When $i = -N/2, \varepsilon_1 \approx 0.91/\alpha S$ as in (7).

### B. Linear Interpolation

For linear interpolation, $h(x) = \text{sinc}^2(x/SG)$. Once again, we substitute this into (10) and (11) to obtain $\hat{h}(i) = \sqrt{(2/3) + (1/3)\cos(2\pi(i/SG))}$ and

$$\varepsilon_1[i] = \sqrt{\frac{\frac{2}{3} + \frac{1}{3}\cos\left(2\pi\frac{i}{SG}\right)}{\text{sinc}^4(i/SG)} - 1}.$$

Since $i \ll SG$ we can approximate $\varepsilon_1(i)$ using the first nonzero term of the Taylor series about $i = 0$, giving

$$\text{linear } \varepsilon_1[i] \approx \frac{\pi^2}{3\sqrt{5}}\left(\frac{i}{SG}\right)^2. \tag{14}$$

When $i = -N/2, \varepsilon_1 \approx 0.37/(\alpha S)^2$ as in (8).

Equations (13) and (14) provide simple expressions for $\varepsilon_1[i]$, the part of the aliasing amplitude due to nearest-neighbor and linear interpolation, respectively. Fig. 12 demonstrates that these approximations are also highly accurate. Equation (12) gives a finite sum solution for $\varepsilon_2[i]$, the part of the aliasing amplitude due to the kernel sample values. These simple and accurate expressions for the aliasing amplitude allow us to quickly evaluate the expected performance of a given gridding kernel and to generate comparisons of kernels as in Fig. 3.

## APPENDIX II
## APODIZATION CORRECTION FUNCTION OF A SAMPLED KERNEL

With a continuous kernel, obtaining the exact apodization correction function requires that we have an analytic solution for the inverse Fourier transform of the kernel. With the sampled kernel we can use the FFT to obtain the apodization correction function as follows.

1) Zero-pad the sampled kernel to length $SG$.
2) Compute the IFFT.
3) Select the $N$ values from $x = -N/2$ to $N/2 - 1$.
4) Multiply by the Inverse Fourier transform of the interpolation function, $h(x)$.
5) Take the reciprocal of these $N$ values.

## APPENDIX III
## CALCULATING THE OPTIMAL SAMPLED KERNEL IN MATLAB

The SOCP in Section VI finds a kernel, $C_s(k_x)$, such that $\varepsilon_2[i] < \varepsilon_2^{\max}$. Although this SOCP is easy to understand, it is not well suited for implementation. As such, we implement the following SOCP which, though slightly more complicated, gives the same result and is better suited for implementation.

$$\begin{aligned} &\text{minimize } t \\ &\text{subject to } \sqrt{\sum_{p=1}^{S-1}[\hat{h}(i+Gp)c_s(i+Gp)]^2} \le \varepsilon_2^{\max} h(i)c_s(i) + t \\ &\qquad C_s(0) = 1 \\ &\qquad C_s(-k_x) = C_s(k_x) \quad \text{for } i \in \left\{0, 1, \ldots, \frac{N}{2}\right\}. \end{aligned}$$

In this SOCP, when $t$ is less than zero, a kernel exists such that $\varepsilon_2[i] < \varepsilon_2^{\max}$.

Many optimization packages exist which will solve second-order cone programs. Lobo *et al.* have written a package specifically for SOCP problems [17]. In addition, the SeDuMi package is a freely available package that will solve many optimization problems, including SOCP problems [18], [19]. Commercial packages, such as MOSEK (www.mosek.com), are also available. Here, we provide a MATLAB program which implements the SOCP shown above, using the yalmip MATLAB toolbox and SeDuMi [20]. $N, G, W$, and $S$ are as defined in Tables I and II.

```
% function Cs = findKernel(N, G, W, S, e2max)
% Find sampled kernel, Cs(kx), such that
% \ epsilon_2[i] < e2max.
% If no such kernel can be found,
% Cs is set to 0.
function Cs = findKernel(N, G, W, S, e2max)
F = set(');
t = sdpvar(1,1);
Cs = sdpvar(W*S/2-1,1);
kx = [1:W*S/2-1]/S/G;
p = [1:S-1]';
for i = 0:(N/2),
  h_hat = sqrt(2/3+1/3*...
          cos(2*pi/S/G*(i+G*p)));
  h = sinc(i/S/G).^2;
  % Use the discrete cosine transform
  % to get cs(x) from Cs(kx)
  cAlias = 2*cos(2*pi*(i+G*p)*kx)*...
           Cs+1;
  cApodize = 2*cos(2*pi*i*kx)*Cs+1;
  F = F+set(cone(h_hat.*cAlias,...
      e2max*h*cApodize+t));
end
solvesdp(F, t);
if (double(t)<=0)
  Cs = [flipud(double(Cs)); 1; double(Cs)];
else
  Cs = 0;
end
```

## ACKNOWLEDGMENT

The authors wish to thank B. Hargreaves for his helpful feedback on this manuscript and for the use of his spiral SSFP knee sequence.

## REFERENCES

[1] J. O'Sullivan, "A fast sine function gridding algorithm for Fourier inversion in computer tomography," *IEEE Trans. Med. Imag.*, vol. MI-4, pp. 200–207, 1985.
[2] H. Schomberg and J. Timmer, "The gridding method for image reconstruction by Fourier transformation," *IEEE Trans. Med. Imag.*, vol. 14, no. 3, pp. 596–607, Sep. 1995.
[3] S. Schaller, T. Flohr, and P. Steffen, "An efficient Fourier method for 3-D radon inversion in exact cone-beam CT reconstruction," *IEEE Trans. Med. Imag.*, vol. 17, no. 2, pp. 244–250, Apr. 1998.
[4] M. Bronstein, A. Bronstein, M. Zibulevsky, and H. Azhari, "Reconstruction in diffraction ultrasound tomography using nonuniform FFT," *IEEE Trans. Med. Imag.*, vol. 21, no. 11, pp. 1395–1401, Nov. 2002.
[5] A. Dutt and V. Rokhlin, "Fast Fourier transforms for nonequispaced data," in *SIAM J. Sci. Comput.*, vol. 14, 1993, pp. 1369–1393.
[6] J. G. Pipe and P. Menon, "Sampling density compensation in MRI: Rationale and an iterative numerical solution," *Magn. Reson. Med.*, vol. 41, no. 1, pp. 179–186, 1999.
[7] V. Rasche, R. Proksa, R. Sinkus, P. Bornert, and H. Eggers, "Resampling of data between arbitrary grids using convolution interpolation," *IEEE Trans. Med. Imag.*, vol. 18, no. 5, pp. 385–392, May 1999.
[8] D. Rosenfeld, "New approach to gridding using regularization and estimation theory," *Magn. Reson. Med.*, vol. 48, no. 1, pp. 193–202, 2002.
[9] J. Fessler and B. Sutton, "Nonuniform fast Fourier transforms using min-max interpolation," *IEEE Trans. Signal Process.*, vol. 51, no. 2, pp. 560–574, Feb. 2003. [*see also IEEE Trans. Acoust., Speech, Signal Process.*].
[10] H. Moriguchi and J. L. Duerk, "Iterative next-neighbor regridding (INNG): Improved reconstruction from nonuniformly sampled k-space data using rescaled matrices," *Magn. Reson. Med.*, vol. 51, no. 2, pp. 343–352, 2004.
[11] J. Jackson, C. Meyer, D. Nishimura, and A. Macovski, "Selection of a convolution function for Fourier inversion using gridding," *IEEE Trans. Med. Imag.*, vol. 10, no. 3, pp. 473–478, Sep. 1991.
[12] L. Greengard and J.-Y. Lee, "Accelerating the nonuniform fast Fourier transform," in *SIAM Rev.*, vol. 46, 2004, pp. 443–454.
[13] M. Frigo and S. G. Johnson. (2004) The FFTW Web Page. [Online]. Available: http://www.fftw.org
[14] P. Irarrazabal and D. G. Nishimura, "Fast three dimensional magnetic resonance imaging," *Magn. Reson. Med.*, vol. 33, no. 5, pp. 656–662, 1995.
[15] F. Wajer, E. Woudenberg, R. de Beer, M. Fuderer, A. Mehlkopf, and D. van Ormondt, "Simple equation for optimal gridding parameters," in *Proc. 7th Ann. Meeting of ISMRM*, Philadelphia, PA, 1999, p. 663.
[16] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret, "Applications of second-order cone programming," *Linear Algebra Its Applicat.*, vol. 284, no. 1–3, pp. 193–228, 1998.
[17] ——, (1998) Second-order cone programming (SOCP). [Online]. Available: http://www.stanford.eduTboyd/SOCP.html
[18] J. F. Sturm, "Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones," *Optimiz. Meth. Softw.*, no. 11–12, pp. 625–653, 1999.
[19] ——, (2003) SeDuMi 1.05. [Online]. Available: http://fewcal.kub.nl/sturm/software/sedumi.html
[20] J. Löfberg. (2004) YALMIP 3. [Online]. Available: http://control.ee.ethz.ch/~joloef/yalmip.msql