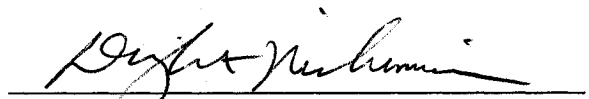
RECONSTRUCTION METHODS FOR FAST MAGNETIC RESONANCE IMAGING

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

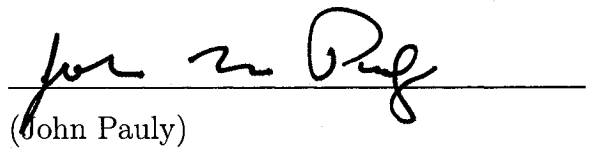
Philip James Beatty
December 2006

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.



(Dwight Nishimura) Principal Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.



(John Pauly)

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Brad Osgood)

Approved for the University Committee on Graduate Studies.

Abstract

Magnetic Resonance Imaging (MRI) is a very successful method for imaging the body, due in large part to the excellent soft tissue contrast that can be obtained. A major challenge for MRI is reducing the long scan times that can be required to obtain an image. Fast Magnetic Resonance Imaging uses sophisticated encoding techniques, such as non-Cartesian k-space trajectories and parallel imaging, to reduce the scan times required for MRI and requires advanced MRI scanner hardware and reconstruction methods.

This work focuses on reconstruction methods for fast MRI. In this dissertation, improvements that can be made to the gridding method for reconstructing MR images encoded using non-Cartesian k-space trajectories are described. In addition, a new method called Anti-aliasing Partially Parallel Encoded Acquisition Reconstruction (APPEAR) is introduced and developed for reconstructing magnetic resonance images encoded using non-Cartesian k-space trajectories and parallel imaging.

The improvements to the gridding method described in this work include using a minimal oversampling ratio, improved design for a sampled convolution kernel, reduced field-of-view reconstruction and using block grid storage. Used together, these improvements can result in a three-fold reduction in computation memory requirements and can reduce the reconstruction time by a factor of approximately thirty times for three-dimensional (3-D) image reconstruction, compared to the use of a Kaiser-Bessel convolution kernel on a 2X oversampled grid using conventional line-by-line and slice-by-slice grid storage.

The APPEAR method is a parallel imaging reconstruction method that can be used with arbitrary k-space trajectories, is non-iterative and does not need to estimate

coil sensitivity functions. In this work, the mathematical framework for parallel imaging reconstruction is extended and this extended framework is used to develop and justify the APPEAR method. The concept of correlation values is introduced and used to improve the efficiency of the APPEAR method. Phantom and in-vivo results are shown for 1-D non-Cartesian k-space trajectories and variable-density spiral k-space trajectories.

Acknowledgements

This dissertation is the final work of my graduate school career, which has spanned a five year period of great change and growth in my life. These years have been very good to me, in large part because of the help and support of many people.

Supporting oneself financially while pursuing graduate studies can be challenging and so I am very grateful for the generous financial support I have received, making it possible for me to come to Stanford and focus on my studies. The financial aid for my first year at Stanford was through the Department of Electrical Engineering and was provided by the Rose M. Chappelear Memorial Fund. I was subsequently named a William R. Hewlett Fellow and the financial aid for my second, third and fourth years was provided by the William R. Hewlett Stanford Graduate Fellowship. In my fifth year, I was supported through a research associateship under my advisor, Professor Dwight Nishimura.

Professors Dwight Nishimura, John Pauly and Brad Osgood, who form my dissertation reading committee and sat on my oral defense committee, have spent a lot of time reading various drafts of this dissertation and discussing this work with me and have helped to make this dissertation much stronger than when the first draft was ejected from the printer. I would also like to thank Professor Donald Cox, who was the chair of my oral defense committee, for his helpful feedback.

This dissertation and the research behind it would not have been possible without the continued support and encouragement of Professor Dwight Nishimura. I met Dwight in the Spring of 2001, when I visited Stanford to explore the possibility of graduate studies at Stanford. That meeting had a large influence on my decision to attend Stanford. During my time at Stanford, Dwight has been my principal

advisor in title as well as in action. Dwight is a clear thinker and this manifests itself in his accessible teaching style, his germane questions and his ability to read a long manuscript and quickly identify the weak points. I have learned a great deal from him, both in and out of the classroom and I would like to thank him for his encouragement, support and commitment to me, my education and this work.

It was through the Medical Image Reconstruction class taught by Professor John Pauly, my associate advisor, that I started on the first project that is a part of this dissertation: using a minimally oversampled grid in gridding reconstruction. John has been a great help in shaping the ideas contained in this dissertation. John is enthusiastic, friendly and most importantly, patient. There have been many times when it has taken me weeks of thinking to figure out how insightful John's comments had been in a previous discussion. Both Dwight and John have worked hard for many years to build a strong research group with the right people and equipment for MR research and without the infrastructure of the Magnetic Resonance Systems Research Laboratory (MRSRL), my work could not have started.

There are many people in the MRSRL that have helped me with problems both big and small. For helpful discussions full of great advice that have helped me to find my way, I would like to thank Krishna Nayak, Brian Hargreaves, Greig Scott, and Steve Conolly. In addition to great advice, Brian has directly and significantly helped my work, providing me with pulse sequence code and Matlab scripts and has come in early in the morning to scan with me on more than one occasion.

For their friendship and all of their research related and non-research related help I would like to thank my office mates and colleagues: Jin-Hyung Lee, Sharon Ungersma, Paul Gurney, Jongho Lee, Logi Viarsson, Michael Lustig, Peter Larson, Julie DiCarlo, Juan Santos, Bob Schaffer, Jong Park, Nathaniel Matter and Ross Venook. I would like to especially thank Paul Gurney, with whom I shared an adjoining cubicle for the majority of my time at Stanford and hotel rooms during the ISMRM conferences. Paul has been the first person that I turn to to discuss a new idea or problem and he has always been very generous with his time and brilliant with his ideas.

In addition to the aforementioned professors, mentors and colleagues at Stanford,

I would like to acknowledge and thank Professor R. Mark Henkelman of the University of Toronto for his advice over the years and for engaging me in the problems of MR image reconstruction during my internship with the Mouse Imaging Centre (MICe) in Toronto in the summer of 2002. As well, I would like to thank Jean Brittain for inviting me to work on parallel imaging reconstruction at General Electric's Applied Science Laboratory in Menlo Park during the summer of 2005. Working with Anja Brau, a lot of progress was made during this summer in understanding parallel imaging reconstruction methods and this has greatly influenced my work.

Lastly, I would like to acknowledge my family for their support and encouragement. I would especially like to thank my loving wife, Tara Tappuni, who has made more sacrifices than anyone, including myself, to give me the opportunity to come to Stanford and accomplish this work.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Magnetic Resonance Imaging (MRI) is a very successful method for imaging the body, due in large part to the excellent soft tissue contrast that can be obtained. However, a major drawback of MRI is the long scan times that can be required to obtain an image.

In order to reduce the time needed to acquire sufficient information to reconstruct an image, fast MRI uses sophisticated encoding techniques, such as non-Cartesian k-space trajectories and encoding using receiver-coil sensitivities. Accurate and efficient image reconstruction from signals acquired using these sophisticated encoding techniques is a challenging problem and the focus of this work. A brief outline describing the focus of each chapter is provided below.

1.1 Thesis Outline

Chapter two provides a brief introduction to MRI, showing how the MR signal is produced, acquired and modeled. The graphical and mathematical perspective used in this chapter sets the stage for the later chapters, which develop and describe methods for constructing images from the acquired MR signal.

The first method that I develop in detail is the gridding method, the topic of chapter three. The gridding method has been used in medical imaging reconstruction

for over twenty years [1], as an efficient way to reconstruct images encoded with non-Cartesian k-space trajectories. After describing the basic gridding method, I describe a number of improvements that can be made to the method that result in significant reductions in computation memory and time without degrading the quality of the reconstructed image. These improvements include: using a minimal oversampling ratio, improved design for a sampled convolution kernel, reduced field-of-view reconstruction and using block grid storage. Used together, these improvements can reduce gridding reconstruction time by a factor of approximately thirty times for three-dimensional (3-D) image reconstructions.

Chapters four and five develop and explore a new method for parallel imaging reconstruction—reconstruction of images that have been encoded by a combination of gradient encoding and receiver coil sensitivities. This new method, called Anti-aliasing Partially Parallel Encoded Acquisition Reconstruction (APPEAR), can be used with arbitrary k-space trajectories, is non-iterative and does not need to estimate coil sensitivity functions.

Chapter four gives a brief overview of previous work on parallel imaging and then extends the theory of multiple receiver coil reconstruction. The APPEAR method is developed and described using the extended theory introduced in this chapter. Phantom and *in vivo* results for a 2D trajectory with variable density sampling in the phase-encode direction show that the APPEAR method is able to remove the aliasing artifacts caused by reduced gradient encoding.

While chapter four develops and describes the APPEAR method and shows that the method is able to achieve good image quality, chapter five deals with the computation required by the APPEAR method. Insight into the computation required by APPEAR is given by a new concept: correlation values. In chapter five, the concept of correlation values is defined and developed. While the proposed implementation of the APPEAR method given in chapter four is pedagogically useful, chapter five shows that correlation values lead to an implementation of the APPEAR method that improves the computational efficiency ten-fold. As well as being significant from a computation point of view, the concept of correlation values can give helpful insight into the APPEAR method in particular and MR image reconstruction in general.

Significant portions of chapter five are devoted to developing this insight.

Chapter six concludes this work, summarizing what I have accomplished in the area of reconstruction methods for fast magnetic resonance imaging and giving my thoughts on future directions in this area.

Chapter 2

Setting the Stage

When an MRI scanner images a part of the body, it obtains an image of the transverse magnetization at each spatial location. Unlike the X-ray attenuation coefficient, which is a simple material property that can be imaged using computed tomography (CT), the transverse magnetization imaged by MRI is not a simple material property.

Magnetic fields are used by an MRI scanner to produce, or excite, transverse magnetization in the tissue being imaged. The amount of transverse magnetization excited in the tissue depends on material properties of the tissue as well as the magnetic fields that are applied to the tissue. Moreover, by controlling the applied magnetic fields, it is possible to vary how the transverse magnetization relates to the material properties of the tissue, allowing MRI scanners to achieve different types of contrast between tissue types as shown in Fig. 2.1.

An MRI scanner works by first creating transverse magnetization that provides a useful contrast in the body part to be imaged, and then collecting information to form an image of the transverse magnetization that has been created. This work is concerned with the latter stage: understanding the information that is collected about the transverse magnetization and how this information can be used to form an image. In this chapter, a model that describes the collected information is developed, providing a foundation for the following chapters, which focus on methods for efficiently transforming the collected information into images. The model used in this chapter is based on a classical description of MR. Although not as rigorous as a

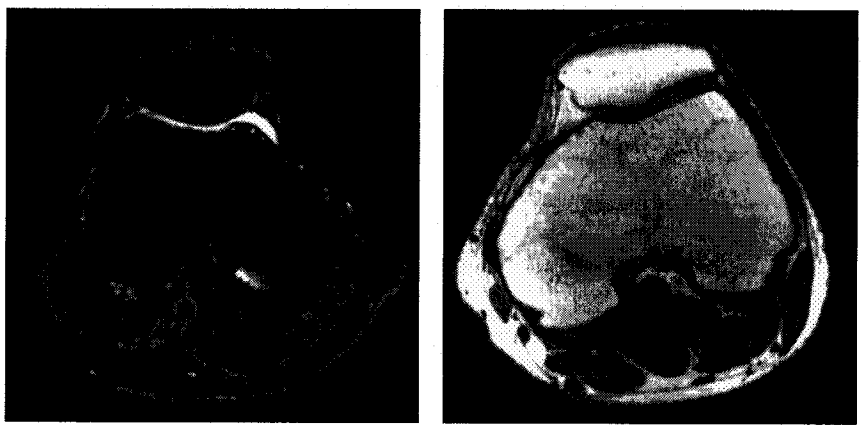


Figure 2.1: Axial knee images shown with two different types of contrast that can be obtained using an MRI scanner. (a) T2-weighted image with suppression of signal from fat tissue. (b) T1-weighted image.

quantum mechanical description, the classical description leads to an intuitive model that is sufficient for the purposes of this work.

2.1 Creating Transverse Magnetization

To create transverse magnetization, a typical MRI scanner uses three complimentary systems: 1) a magnet that creates a strong homogeneous magnetic field B_0, 2) a radio-frequency pulse generator and transmit coil that can produce a time-varying magnetic field at the resonant frequency of the magnetization and 3) a gradient system with x, y and z-axis gradient coils that can produce a magnetic field that varies in time and linearly in space.

The B_0 field is perpetually existent within an MRI scanner, its direction defining the longitudinal direction, typically plotted along the z-axis. When an object is placed in the scanner, the B_0 field causes the magnetic moments of certain nuclei to tend to align along the longitudinal direction, producing a net magnetic moment, or

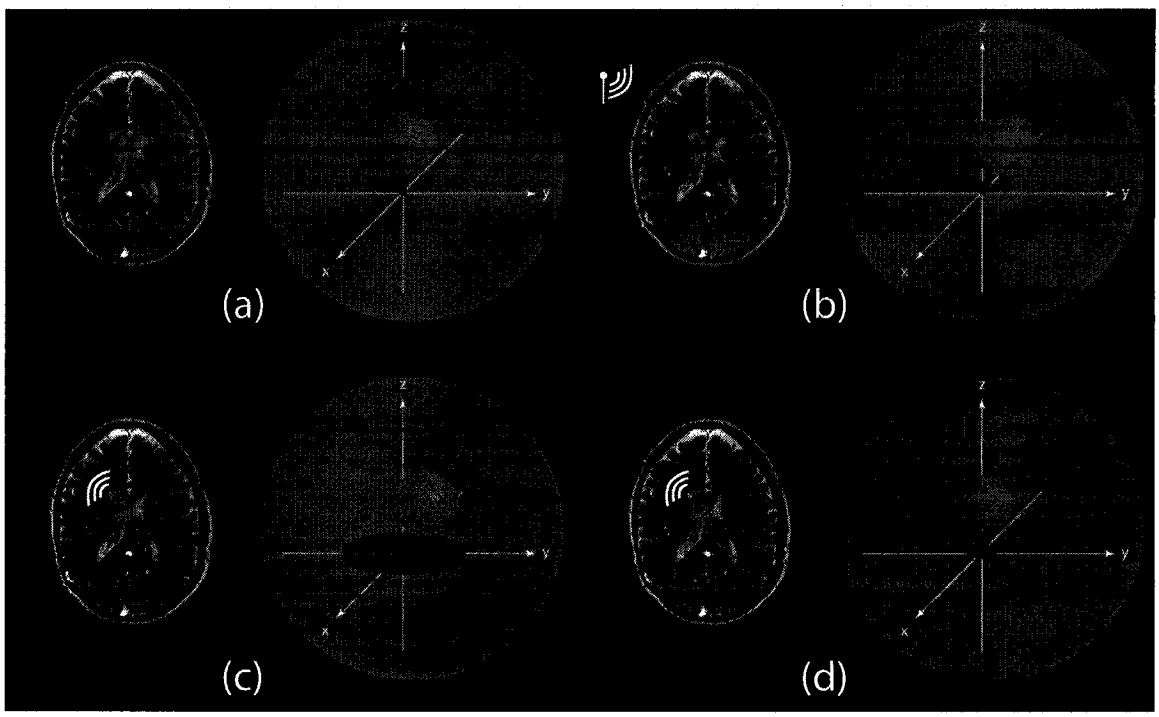


Figure 2.2: (a) Strong external magnetic field aligns magnetic moments, producing a net magnetization in the direction of the field. (b) Radio-frequency pulse tips the magnetization out of equilibrium. (c) Tipped out of equilibrium, the magnetization precesses in the transverse (xy) plane, emitting a radio-frequency signal at a frequency proportional to the magnetic field. (d) In addition to precessing, the magnetization also returns to equilibrium.

magnetization, in the longitudinal direction as shown in Fig. 2.2(a).

In the body, the most abundant, by far, of the affected nuclei are hydrogen nuclei, found in both water and lipids. Each of the nuclei has a resonant frequency proportional to the surrounding magnetic field. The constant of proportionality γ is called the gyromagnetic ratio and for hydrogen nuclei, $\gamma/2\pi = 4257\,\mathrm{Hz}\,/\,\mathrm{Gauss}$. Thus, in a 1.5T magnetic field, a common value for B_0, hydrogen nuclei will have a resonant frequency of about 64 MHz. The resonant frequency is also called the Larmor frequency.

The radio-frequency (RF) pulse generator and transmit coil are used to tip the magnetization out of equilibrium, creating transverse magnetization as shown in Fig.

2.2(b). While the magnitude of the RF magnetic field is many orders of magnitude smaller than the B_0 field, it succeeds in tipping the magnetization by operating at the resonant frequency of the magnetization. Because only magnetization whose resonant frequency is the same as the frequency of the RF pulse is tipped, the gradient system can be used to limit the locations where transverse magnetization is created. For example, applying a gradient field along the x-axis will cause the resonant frequency to vary linearly in the x direction. Applying an RF pulse while this gradient field is applied will create transverse magnetization in a yz plane at a single x location. Together, the RF system and gradient system allow for great flexibility in creating transverse magnetization.

Mathematically, the transverse magnetization is modeled as a complex-valued function $m(\mathbf{r})$, where $\mathbf{r}$ is an (x, y, z) three-tuple that gives the spatial location. By convention, the real part of $m(\mathbf{r})$ corresponds to the x-component of the transverse magnetization and the imaginary part to the y-component of the transverse magnetization. As shown in Fig. 2.2(c), when the magnetic moments are tipped out of equilibrium, they precess in the transverse (xy) plane at the Larmor frequency, $f(\mathbf{r})$. As previously noted, the Larmor frequency at a spatial location $\mathbf{r}$ is proportional to the magnetic field at that location:

$$f(\mathbf{r}) = \frac{\gamma}{2\pi} B(\mathbf{r}), \tag{2.1}$$

where $B(\mathbf{r})$ is the magnetic field and γ is the gyromagnetic ratio. As the magnetization precesses, it emits a radio-frequency signal at the Larmor frequency with a magnitude proportional to the magnitude of the transverse magnetization. It is this emitted signal that is detected, providing information about the transverse magnetization.

In addition to precessing, the magnetization also returns to equilibrium, decaying in the transverse plane with a time constant of T2 and re-growing in the longitudinal direction with a time constant of T1. As the magnetization returns to equilibrium, the net magnetization does not remain constant, as the coherency of the magnetic moments of the nuclei can change during this time. The time constants T1 and T2 are material properties that differ from tissue to tissue and are the most common

material properties used to provide contrast in MR imaging.

In a typical MR scan, the magnetization will be repeatedly excited, or tipped into the transverse plane, with some signal detected after each excitation. By adjusting the time between repetitions and the time at which the signal is acquired, T1-weighted and T2-weighted contrasts can be achieved.

2.2 The Emitted Signal

As discussed in the previous section, an MRI scanner can be programmed to cause each spatial location in an object to emit a radio-frequency signal with a magnitude proportional to the transverse magnetization at that particular spatial location. The difficulty in forming an image arises from the fact that all excited spatial locations emit a signal at the same time, making it challenging to differentiate the signal at one spatial location from another spatial location. In this section, I describe how the gradient system can be used to warp the phase of the emitted signal by location. This procedure, called gradient encoding, can be used to encode spatial location information.

The emitted signal, $m(\mathbf{r}, t)$ is the signal emitted from location $\mathbf{r}$ at time t, after being demodulated around the carrier frequency $f_0 = \frac{\gamma}{2\pi} B_0$. In the absence of the gradient system, $m(\mathbf{r}, t) = m(\mathbf{r})$.

Gradient encoding uses the magnetic field produced by the gradient system to change the Larmor frequency during the scan. The magnetic field produced by the gradient system varies linearly in space and is expressed in terms of the gradient field strength $\mathbf{G}(t)$, which has units of Gauss/cm. The magnetic field produced by the gradient system is $\mathbf{G}(t) \cdot \mathbf{r}$, changing the Larmor frequency by

$$\Delta f(\mathbf{r}, t) = \frac{\gamma}{2\pi} \mathbf{G}(t) \cdot \mathbf{r}. \tag{2.2}$$

By integrating the frequency over time, it is possible to obtain the change in the

phase of the signal caused by the gradient system:

$$\begin{aligned} \Delta\phi(\mathbf{r},t) &= 2\pi \int_0^t -\Delta f(\mathbf{r},\tau)\, d\tau && (2.3)\\ &= 2\pi \int_0^t -\frac{\gamma}{2\pi}\mathbf{G}(\tau)\cdot\mathbf{r}\, d\tau && (2.4)\\ &= -2\pi\mathbf{k}(t)\cdot\mathbf{r}, && (2.5) \end{aligned}$$

where

$$\mathbf{k}(t) = \int_0^t \frac{\gamma}{2\pi}\mathbf{G}(\tau)d\tau. \quad (2.6)$$

The negative sign in Eq. 2.3 is the result of the precession direction of protons. The change in phase expressed in Eq. 2.5 can be included in the equation for the emitted signal:

$$\begin{aligned} m(\mathbf{r},t) &= m(\mathbf{r})e^{i\Delta\phi(\mathbf{r},t)} && (2.7)\\ &= m(\mathbf{r})e^{-i2\pi\mathbf{k}(t)\cdot\mathbf{r}}. && (2.8) \end{aligned}$$

The next section, which looks at how the emitted signals are received by a receiver coil, completes the encoding picture and shows how gradient encoding can be used to encode spatial location information.

2.3 The Received Signal

At a given time, a radio-frequency receiver coil measures a single value: the voltage induced in the coil as a result of the magnetic flux through the coil. The transverse magnetization at each location in the excited object will make some contribution to the magnetic flux through the coil and hence some contribution to the induced voltage. The measured voltage is the sum of the contributions from all locations. The contribution of the transverse magnetization at a spatial location to the measured voltage depends on the coil geometry and loading as well as the spatial location of the magnetization. Using j to index the receiver coils, the receiver-coil sensitivity function for coil j, $s_j(\mathbf{r})$, is a complex-valued function that characterizes the contribution to

the coil measurement from the emitted signal $m(\mathbf{r},t)$. Summing the contributions from all locations of the magnetization expressed in Eq. 2.8, the following equation for the received MR signal $d_j(t)$ is derived:

$$d_j(t) = \int_V s_j(\mathbf{r}) m(\mathbf{r},t)\, dV \tag{2.9}$$

$$= \int_V s_j(\mathbf{r}) m(\mathbf{r}) e^{-i2\pi \mathbf{k}(t)\cdot\mathbf{r}}\, dV. \tag{2.10}$$

Although this simple signal equation does not take into account many things, including the decay of the transverse magnetization and factors such as B_0 inhomogeneity and changes in magnetic susceptibility that can affect the rate of precession, it provides in many cases an adequate model for the received signal on an MR system and is the model that will be used for the remainder of this work.

In many imaging situations only one receiver coil is used: the so called 'body coil' which is, ideally, uniformly sensitive to all spatial locations ($s_j(\mathbf{r}) = 1$). In this case, Eq. 2.10 simplifies to

$$d(t) = \int_V m(\mathbf{r}) e^{-i2\pi \mathbf{k}(t)\cdot\mathbf{r}}\, dV. \tag{2.11}$$

In Eq. 2.11 an acquired datum $d(t)$ is the result of multiplying the magnetization by a complex exponential and summing the result. At each point in time the magnetization is multiplied by a different complex exponential as shown on the left column of Fig. 2.3. Recognizing $e^{-i2\pi \mathbf{k}(t)\cdot\mathbf{r}}$ as a Fourier kernel, an acquired datum can also be interpreted as a sample in spatial-frequency or k-space, as shown on the right column of Fig. 2.3. Equation 2.6 gives the relationship between gradient field strength $\mathbf{G}(t)$ and k-space location. After transverse magnetization is excited, waveforms are played out on the gradient system, Fig. 2.4(a), that result in a desired k-space trajectory as shown in Fig. 2.4(b).

Expressing the received signal in Eq. 2.11 in terms of k-space location, Eq. 2.11 can be rewritten as the Fourier transform:

$$d(\mathbf{k}) = \int_V m(\mathbf{r}) e^{-i2\pi \mathbf{k}\cdot\mathbf{r}}\, dV. \tag{2.12}$$

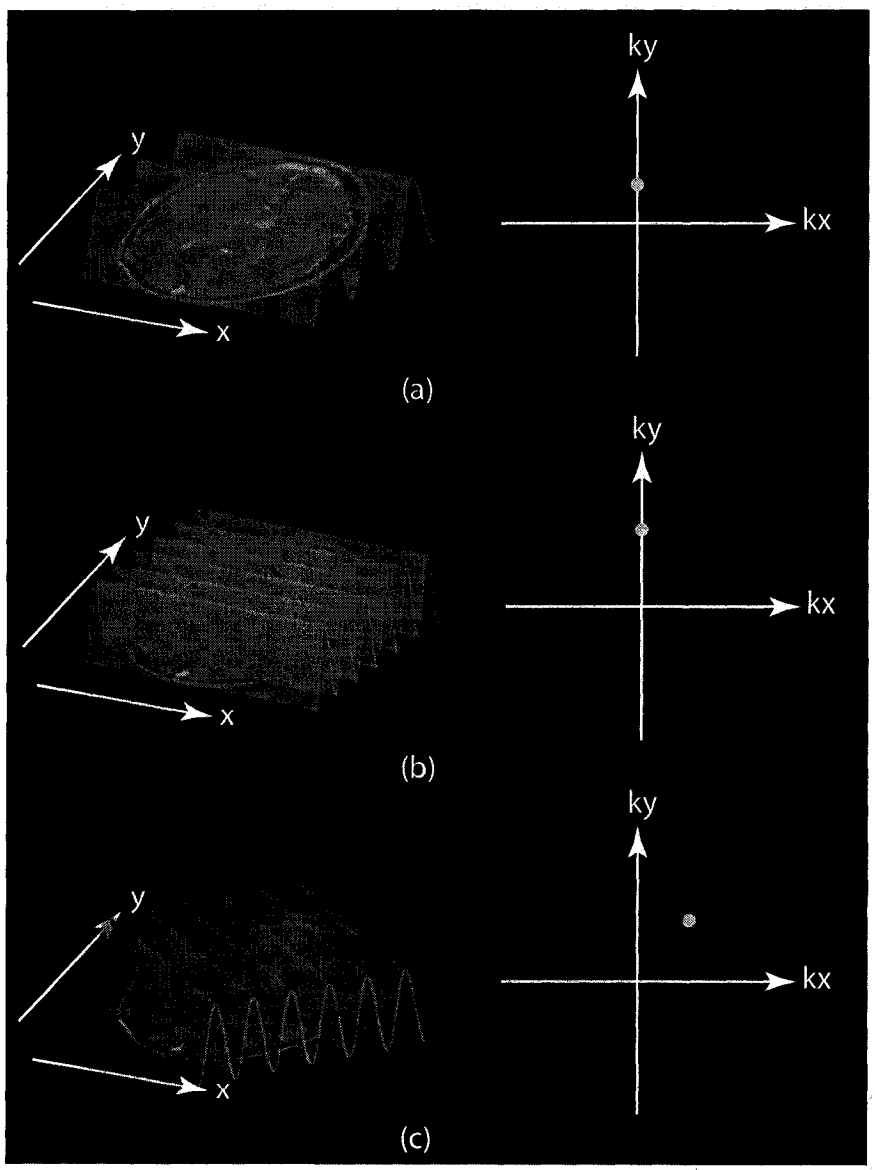


Figure 2.3: When only gradient encoding is considered, an acquired datum can be modeled as the result of multiplying the magnetization by a complex exponential image and summing the result. Alternatively, each datum can be viewed as acquiring a sample of k-space. Shown in (a)-(c) are these two perspectives for three different encoding locations.

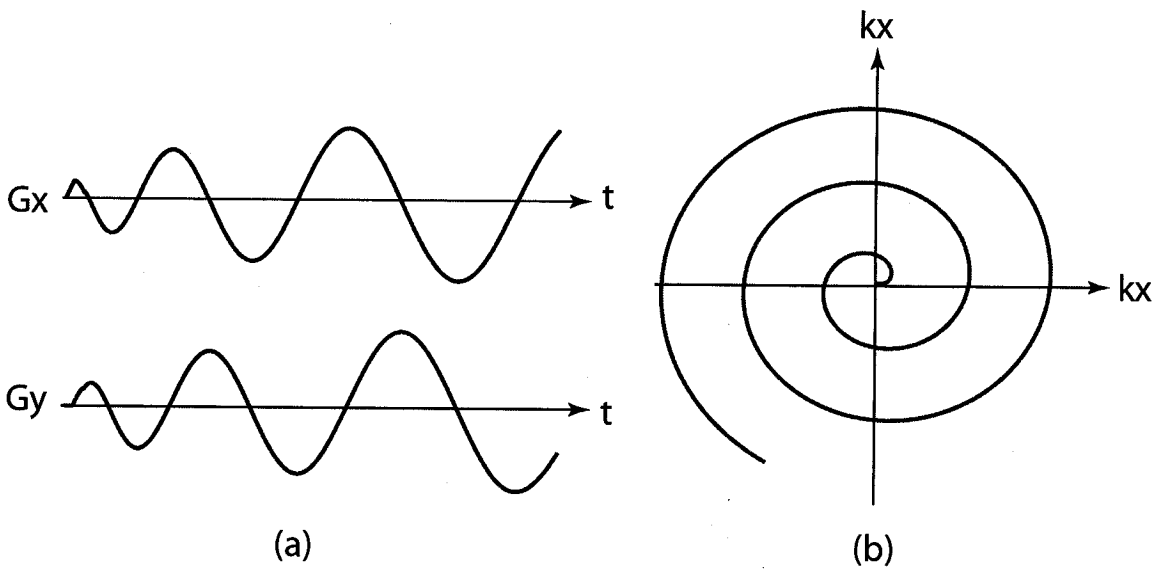


Figure 2.4: Playing gradient waveforms (a) trace a trajectory through k-space (b). The relationship between the gradient field strength and k-space trajectory is given by Eq. 2.6.

Likewise, the magnetization can be expressed in terms of the emitted signal function $d(\mathbf{k})$ using the inverse Fourier transform:

$$m(\mathbf{r}) = \int_{k\text{-space}} d(\mathbf{k}) e^{i2\pi \mathbf{k}\cdot\mathbf{r}}\, dk_x dk_y dk_z. \tag{2.13}$$

Equation 2.13 provides useful insight into MR image reconstruction, even if it is an unusable equation since an MR scanner can only collect a finite number of samples of $d(\mathbf{k})$ along a k-space trajectory and the entire $d(\mathbf{k})$ function is never known.

Due to the flexibility of modern gradient systems, many different k-space trajectories can be achieved, four of which are shown in Fig. 2.5. One advantage of the DFT trajectory shown in Fig. 2.5(a) is that image reconstruction can be performed with a two-dimensional Fast Fourier Transform (2-D FFT), since the data falls on a Cartesian grid. The 2-D FFT computes evenly-spaced samples of $m_s(x, y)$:

$$m_s(x, y) = \sum_{j=1}^{N_s} d\left(k_{x,j}, k_{y,j}\right) e^{i2\pi(k_{x,j}x + k_{y,j}y)}, \tag{2.14}$$

where$(k_{x,j}, k_{y,j})$ select the N_s evenly-spaced k-space locations that have been acquired along k_x and k_y. Equation 2.14 can be related to a 2-D version of $m(\mathbf{r}) = m(x, y)$ as

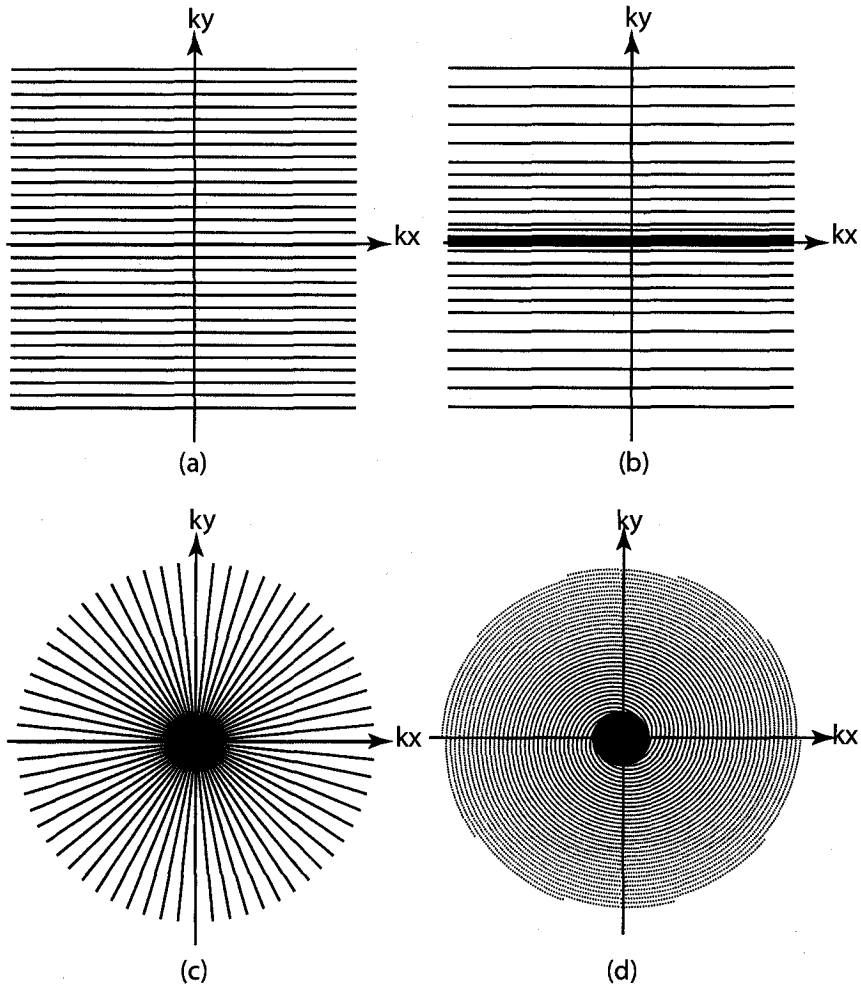


Figure 2.5: Four possible k-space trajectories: (a) DFT, (b) variable-density in the phase-encode direction, (c) radial, (d) variable-density spiral.

follows:

$$\begin{aligned} m_s(x,y) &= \sum_{j=1}^{N_s} d\left(k_{x,j},k_{y,j}\right) e^{i2\pi(k_{x,j}x+k_{y,j}y)} \\ &= \sum_{j=1}^{N_s} \int_{k\text{-space}} d\left(k_x,k_y\right) \delta\left(k_x-k_{x,j},k_y-k_{y,j}\right) e^{i2\pi(k_x x+k_y y)} dk_x dk_y \\ &= \int_{k\text{-space}} d\left(k_x,k_y\right) \sum_{j=1}^{N_s} \delta\left(k_x-k_{x,j},k_y-k_{y,j}\right) e^{i2\pi(k_x x+k_y y)} dk_x dk_y \\ &= \mathrm{IFT}\left\{ d\left(k_x,k_y\right) \sum_{j=1}^{N_s} \delta\left(k_x-k_{x,j},k_y-k_{y,j}\right) \right\}(x,y) \\ &= \mathrm{IFT}\left\{d\left(k_x,k_y\right)\right\}(x,y) * \mathrm{IFT}\left\{ \sum_{j=1}^{N_s} \delta\left(k_x-k_{x,j},k_y-k_{y,j}\right) \right\}(x,y) \\ &= m(x,y) * \mathrm{PSF}(x,y), \end{aligned} \tag{2.15}$$

where

$$\mathrm{PSF}(x,y) = \mathrm{IFT}\left\{ \sum_{j=1}^{N_s} \delta\left(k_x-k_{x,j},k_y-k_{y,j}\right) \right\}(x,y). \tag{2.16}$$

The point-spread-function $\mathrm{PSF}(x,y)$ describes how the k-space sampling of the trajectory affects the reconstructed image. As shown in Fig. 2.6, the point-spread-function for the DFT trajectory consists of equally-spaced points. The sharpness of the points determines the resolution of the reconstructed image and the distance between points determines the size of object that can be imaged without introducing aliasing artifacts, shown in Fig. 2.6(b).

An advantage of the variable-density spiral trajectory shown in Fig. 2.5(d) is that k-space can be sufficiently sampled with fewer magnetization excitation repetitions, resulting in faster data acquisition. Such an advantage is very important in fast MRI. A disadvantage of such a non-Cartesian k-space trajectory is that the reconstruction becomes more complicated. Just as the FFT is an efficient method for calculating the sum in Eq. 2.14 for evenly spaced values of x and y, gridding, covered in detail

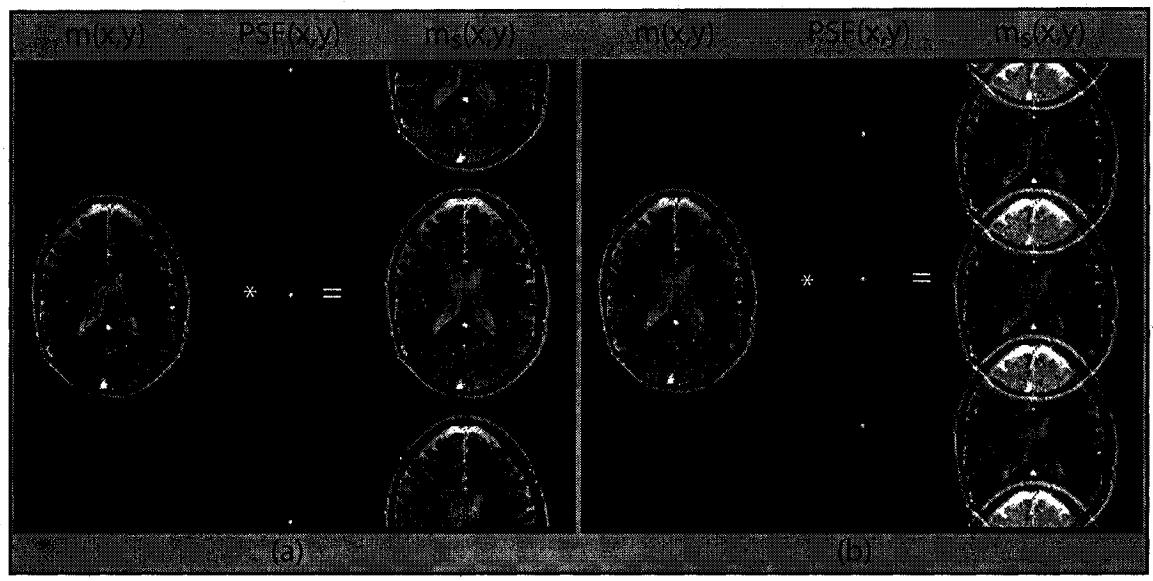


Figure 2.6: Graphical illustration of Eq. 2.15 with (a) sufficient gradient encoding and (b) insufficient gradient encoding, leading to aliasing artifacts.

in the next chapter, is an efficient method for calculating

$$m_s(x,y) = \sum_{j=1}^{N_s} d\left(k_{x,j}, k_{y,j}\right) w\left(k_{x,j}, k_{y,j}\right) e^{i2\pi(k_{x,j}x + k_{y,j}y)} \tag{2.17}$$

for evenly spaced values of x and y when $(k_{x,j}, k_{y,j})$ do not fall on a Cartesian grid. The density-compensation weight values $w(k_{x,j}, k_{y,j})$ allow a more ideal point-spread-function to be achieved. An ideal point-spread-function has a sharp point at $x = y = 0$ and no or very little content within a radius of the desired field-of-view around the central point. Starting with Eq. 2.17 and imitating the derivation of Eq. 2.15, an image reconstructed using gridding can be expressed in the same form as Eq. 2.15 where

$$\mathrm{PSF}(x,y) = \mathrm{IFT}\left\{\sum_{j=1}^{N_s} w\left(k_{x,j}, k_{y,j}\right) \delta\left(k_x - k_{x,j}, k_y - k_{y,j}\right)\right\}(x,y). \tag{2.18}$$

Figure 2.7 illustrates convolution with a spiral trajectory point-spread-function. In (a)

sufficient gradient encoding is performed such that convolving with the point-spread-function does not lead to aliasing artifacts. When insufficient gradient encoding is performed, as shown in (b), the aliasing artifacts produced by convolving with the spiral point-spread-function have a very different structure compared to the aliasing artifacts associated with the DFT trajectory, shown in Fig. 2.6(b).

Expressing $m_s(x, y)$ as the magnetization function $m(x, y)$ convolved with a point-spread-function $\mathrm{PSF}(x, y)$ as in Eq. 2.15 gives a useful visualization of the operation performed by gridding. However it is more useful for the development of the gridding method to view $m_s(x, y)$ as the inverse Fourier transform of the sampled, density-compensated k-space data function $M_s(k_x, k_y)$:

$$m_s(x, y) = \mathrm{IFT}\left\{M_s(k_x, k_y)\right\}(x, y), \tag{2.19}$$

where

$$M_s(k_x, k_y) = \sum_{j=1}^{N_s} d\left(k_x, k_y\right) w\left(k_{x,j}, k_{y,j}\right) \delta\left(k_x - k_{x,j}, k_y - k_{y,j}\right). \tag{2.20}$$

The equivalence of Eq. 2.19 to Eq. 2.17 can be shown by substituting Eq. 2.20 into Eq. 2.19 and simplifying. Calculating the inverse Fourier transform of $M_s(k_x, k_y)$ using a direct implementation of Eq. 2.17 is computationally prohibitive; the next chapter discusses how the gridding method efficiently computes this transform.

As discussed above, one way to reduce the encoding time necessary to fully encode an image is by using non-Cartesian trajectories. The encoding time can also be reduced by using multiple receiver coils. When more than one coil is present, the receiver-coil sensitivity functions can be used to encode spatial location information, allowing an image to be reconstructed without being fully gradient encoded. Such encoding schemes are often called 'parallel imaging', since at any one time N_c separately encoded data values are acquired, where N_c is the number of receiver coils. To see more clearly how the magnetization is encoded in such a scheme, Eq. 2.10 can be

Figure 2.7: Graphical illustration of convolving with a spiral k-space trajectory point-spread-function with (a) sufficient gradient encoding and (b) insufficient gradient encoding, leading to aliasing artifacts.

rewritten in terms of an encoding function $e_j(\mathbf{k}, \mathbf{r})$:

$$d_j(t) = \int_V s_j(\mathbf{r}) m(\mathbf{r}) e^{-i2\pi \mathbf{k}(t)\cdot\mathbf{r}}\, d\mathbf{r} \tag{2.21}$$

$$= \int_V m(\mathbf{r}) e_j(\mathbf{k}, \mathbf{r})\, d\mathbf{r} \tag{2.22}$$

where

$$e_j(\mathbf{k}, \mathbf{r}) = s_j(\mathbf{r}) e^{-i2\pi \mathbf{k}(t)\cdot\mathbf{r}} \tag{2.23}$$

In this case, each acquired datum can be viewed as multiplying the magnetization by an encoding function and summing the result. The encoding function $e_j(\mathbf{k}, \mathbf{r})$ is a product of both the receiver-coil sensitivity function and the gradient encoding complex exponential, as shown in Fig. 2.8. The methods described in chapters 4 and 5 take advantage of both gradient encoding and receiver-coil sensitivity encoding, allowing for faster acquisitions.

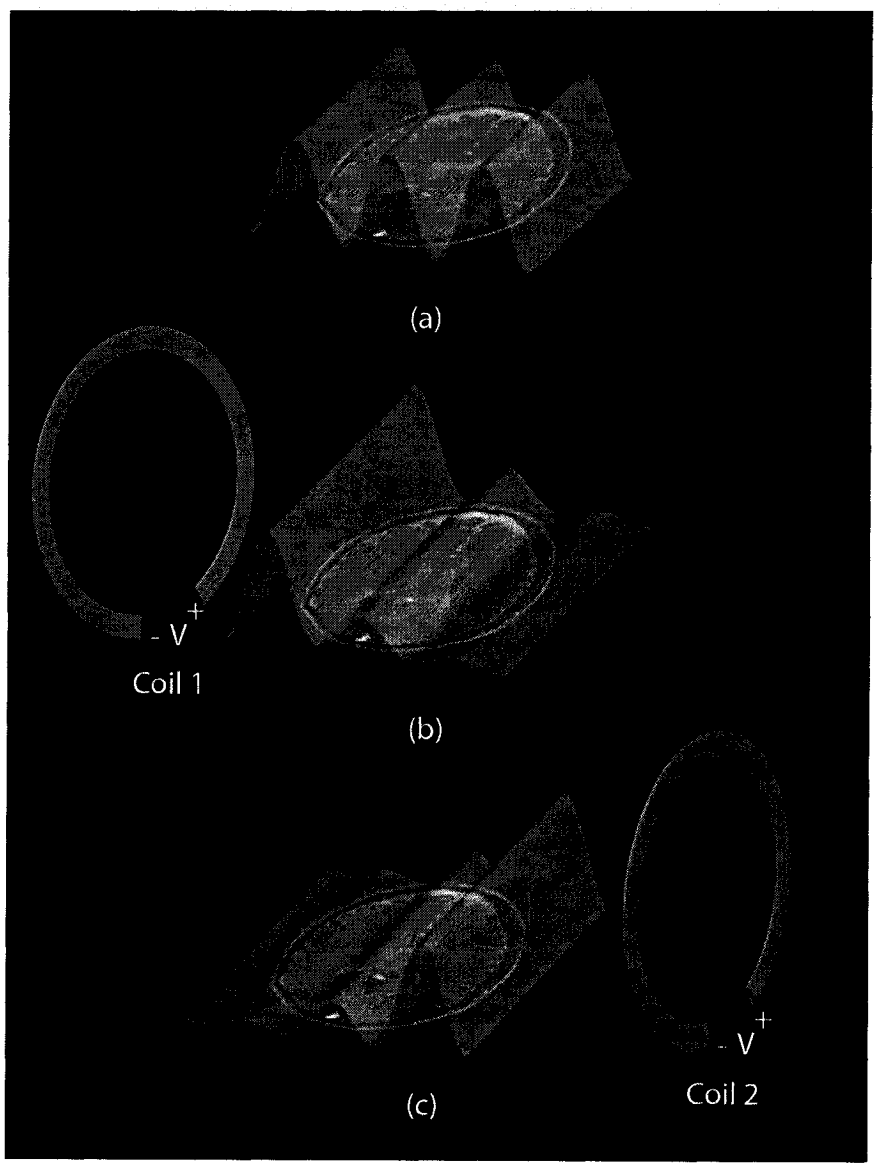


Figure 2.8: (a) As in Fig. 2.3, a datum acquired using the body coil can be modeled as multiplying the magnetization by a complex exponential image and summing the result. (b), (c) For coils with non-uniform coil sensitivity, an acquired datum can be modeled as multiplying the magnetization by an encoding function (Eq. 2.23) and summing the result.

Chapter 3

Gridding Reconstruction

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 lie 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]. 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. In this chapter, I describe a number of techniques that can be used to reduce the computation memory and time needed by the gridding method while maintaining high reconstruction accuracy. These techniques include using a minimal oversampling ratio, presampling the kernel and taking advantage of block storage. Although I focus primarily on MRI reconstruction, gridding can also be used in cone-beam CT reconstruction and ultrasound tomography [3, 4].

The gridding algorithm can be divided into two stages: sampling density compensation and inverse Fourier transform; in this chapter I 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 a fast Fourier transform 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, 9, 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 2D imaging, such methods are appropriate. The work in this chapter 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 one is 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 (1.125-1.375), 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 3.3 and 3.5 I develop two kernel design methods.

On the one hand, memory constraints make undesirable the use of precomputed weights for each data sample; 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]. In this chapter, I show that a similar efficiency can be achieved 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 easily fit in the computer cache. I 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 the gridding method is implemented such that the reconstruction computation time is as short as possible while being confident that no reconstruction artifacts are noticeable in the final image. I define the aliasing amplitude, a metric for gridding accuracy, which allows one 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, one can have great confidence in the fidelity of the reconstruction. For the aliasing amplitude requirements of MR image reconstruction, parameter combinations with minimal oversampling ratios are much more computationally expedient than combinations that 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.

When a minimal oversampling ratio and presampled kernel are used to perform a 3-D reconstruction, the computation time required for both gridding and FFT is limited, not by the ability of the processor to accomplish the computation, but by the time it takes to get the appropriate data from the memory to the processors for the computation. By organizing the grid in a block storage format, data likely to be accessed concurrently is put in the same page of memory. The block storage format reduces processor-to-memory traffic and can lead to a factor of four reduction in computation time. By combining the use of a minimal oversampling ratio, a presampled kernel and block storage, the computation time for gridding can be reduced by a factor of approximately thirty, compared to a less sophisticated approach that does not take advantage of any of these techniques.

In addition to being simple and robust, gridding also allows for a certain amount of flexibility since the resolution and field-of-view of the final image can be chosen independently of the resolution of the acquired data and the spatial extent of the

object being imaged. In this chapter, I show some practical situations that take advantage of this flexibility of the gridding method, including the ability to reconstruct an image with a field-of-view smaller than the extent of the object being imaged.

3.1 The Gridding Algorithm

For the rest of this chapter, I shall assume that the data has already been density compensated and I 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 (I use fftw) [13].
4. Multiply by an apodization correction function.

Throughout this chapter I treat the one dimensional case for pedagogical simplicity; it is easy to extend the result to two and three dimensions when separable gridding kernels are used. I define the variables and functions used throughout this chapter in Tables 3.1 and 3.2. In addition, I define the 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 I sample k-space at G evenly spaced points in the range $[-0.5, 0.5]$, since I measure k-space in *cycles* per unit distance. With these conventions, the gridding algorithm can be expressed mathematically as

$$\widehat{m}_s[i] = \mathrm{IFT}\left\{\left[M_s(k_x) * C(k_x)\right] \text{Ш}\left(Gk_x\right)\right\}(i)\frac{1}{c(i)}. \tag{3.1}$$

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. One would like gridding to perform the mathematical operations given in Eq. 3.1, however in practice the FFT algorithm is used, not a continuous inverse Fourier

Table 3.1: 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, $i \in \left\{-\frac{N}{2}, -\frac{N}{2}+1, \ldots, \frac{N}{2}-1\right\}$.
$M_s(k_x)$	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, $\text{Ш}(x) = \sum_{p=-\infty}^{\infty} \delta(x-p)$.
$\text{FT}\{f(x)\}(k_x)$	Fourier transform. $\text{FT}\{f(x)\}(k_x) = \int_{-\infty}^{\infty} f(x) \exp(-j2\pi k_x x) dx$
$\text{IFT}\{f(k_x)\}(x)$	Inverse Fourier transform. $\text{IFT}\{f(k_x)\}(x) = \int_{-\infty}^{\infty} f(k_x) \exp(j2\pi k_x x) dk_x$

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

Table 3.2: 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.
	$C_s(k_x) = C(k_x) \amalg (SGk_x)$
$c_s(x)$	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 $\widehat{C}(k_x)$.
	$\widehat{c}(x) = c_s(x)h(x)$

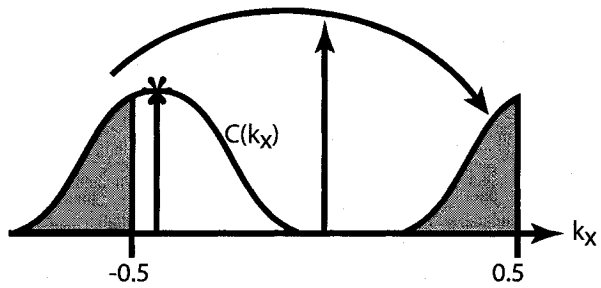


Figure 3.1: The convolution with the gridding kernel is a circular convolution. Thus, when the gridding kernel goes beyond the extent of the grid, it wraps around to the other side of the grid.

transform. Because of this, one only computes $\widehat{m}_s[i]$ for integer values of i—that is, one computes image values at discrete pixel locations. As well, there is the following 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 as shown in Fig. 3.1. 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 2D images and DW^3 for 3D images, where D is the number of data points and W is the width of the kernel on the oversampled grid. This term can be reduced by shrinking W and by reducing the constant of proportionality. The second term is computing the FFT. For grid oversampling ratio α and image width N, it is proportional to $\alpha^2 N^2 \log \alpha N$ for 2D 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 2D images and $\alpha^3 N^3$ for 3D images, a smaller α also reduces the memory requirements.

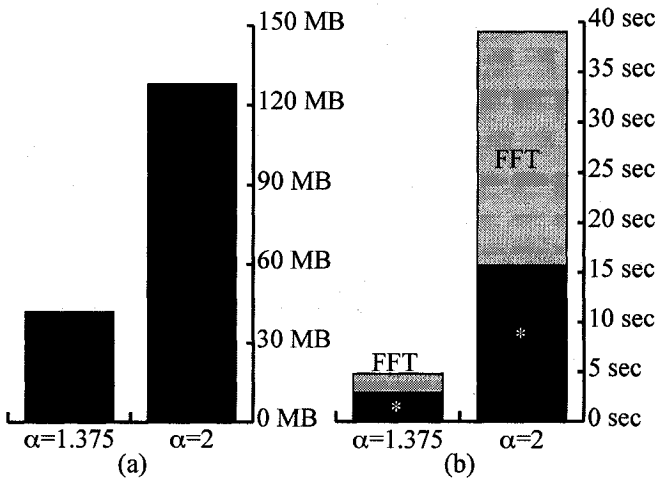


Figure 3.2: Reconstruction of a $128 \times 128 \times 128$ image from the 3-D cones trajectory with 2,304,000 points [14]. With a minimal oversampling ratio of 1.375, I used a sampled Kaiser-Bessel kernel, $W = 5$, with linear interpolation. With the typical oversampling ratio of two, I used a Kaiser-Bessel kernel, $W = 4$. Both parameter choices have similar aliasing amplitudes, resulting in similar reconstruction errors (Fig. 3.11). (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, but without the advantage of the block storage format.

When one moves to a minimal oversampling ratio to decrease the computation time of the FFT, if one wishes to maintain the same accuracy, one 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. Figure 3.2 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 α and W, although there is a limit, since one cannot go below an oversampling ratio of one and interpolation must take some positive amount of time.

3.2 Aliasing Amplitude: A Metric for Gridding Accuracy

As noted in the previous chapter, the goal of the gridding algorithm is to find the inverse Fourier transform of $M_s(k_x)$. Put simply, the goal is to have $\widehat{m}_s[i] = m_s(x)$ 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, one can calculate the image reconstruction error, $E[i] = |m_s(i) - \widehat{m}_s[i]|$, where $m_s(i)$ is calculated directly using the sum in Table 3.1 and $\widehat{m}_s[i]$ is found by performing the gridding reconstruction. It should be noted that while 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, I rewrite Eq. 3.1 completely in the image domain:

$$\widehat{m}_s[i] = \left\{ [m_s(i)c(i)] * \left[\frac{1}{G} \text{Ш}\left(\frac{i}{G}\right) \right] \right\} \frac{1}{c(i)}. \tag{3.2}$$

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 Eq. 3.2, one can rewrite the image reconstruction error as

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

Since the error depends on $m_s(x)$, it depends on the k-space sampling pattern and the object being imaged. As one tries 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 one 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, one does not get an idea of how the error will vary from position to position in the image. As well, it is desirable to have a metric which will predict, at least in terms of the order of magnitude, what one 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, I obtain a metric that is both a function of position and can be used to estimate $E[i]$. I call this metric the *aliasing amplitude*, $\varepsilon[i]$:

$$\varepsilon[i] = \sqrt{\frac{1}{[c(i)]^2} \sum_{p \neq 0} [c(i + Gp)]^2}. \tag{3.3}$$

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 $\widehat{m}_s[i]$ is $m_s(i)$ and the standard deviation of location i in the image is identical to the aliasing amplitude at this pixel location.

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, one can reliably predict the order of magnitude and profile of the error from the aliasing amplitude, as shown in Fig. 3.3. This allows one to design kernels using the aliasing amplitude as a metric for accuracy.

Figure 3.4 provides a reference of the maximum aliasing amplitude of an image for various oversampling ratios and kernel widths. With the typical signal-to-noise ratio (SNR) of MR images, an aliasing amplitude of 0.1 will likely suppress visible artifacts, allowing the use of 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 reconstruction field-of-view 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 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. 3.5, I 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.

3.3 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, α, 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. 3.6. The error images in (b) and (d)

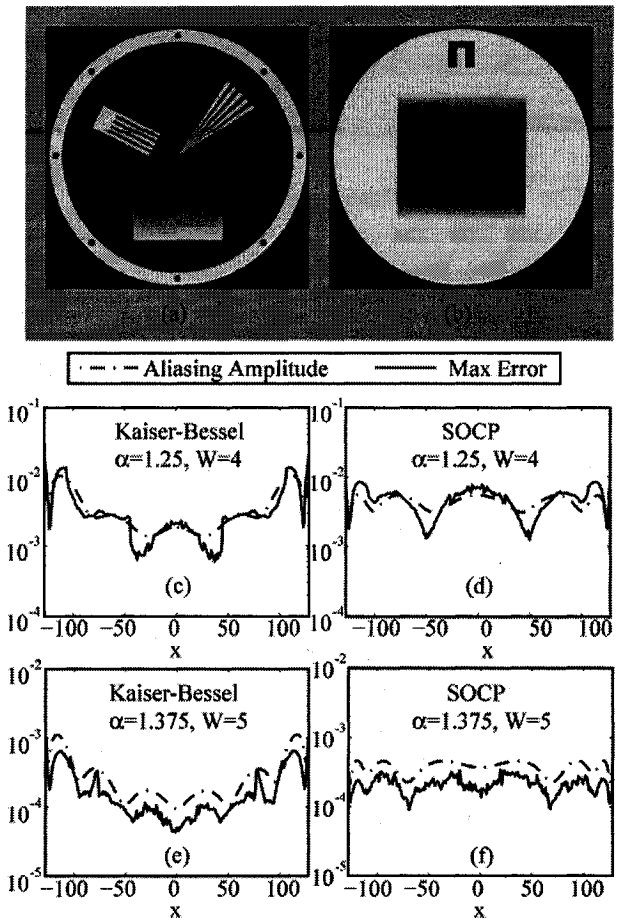


Figure 3.3: The aliasing amplitude, $\varepsilon[i]$, is a good predictor for the image reconstruction error, $E[i]$. Sampling the two numerical phantoms (a) and (b) with both radial and spiral k-space trajectories, produces four data sets. In order to look at the 1D error pattern, I used a very low error method in the y-direction (oversampling ratio $\alpha = 2$, with a Kaiser-Bessel kernel of width $W = 8$). In the x-direction I 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, I reconstructed four 256×256 images (two phantoms $\times$ two trajectories) and subtracted them from the ideal reconstructions using the exact inverse Fourier transform. Taking the absolute value I obtained the image reconstruction error, $E[i]$. This gave 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 one is likely to see.

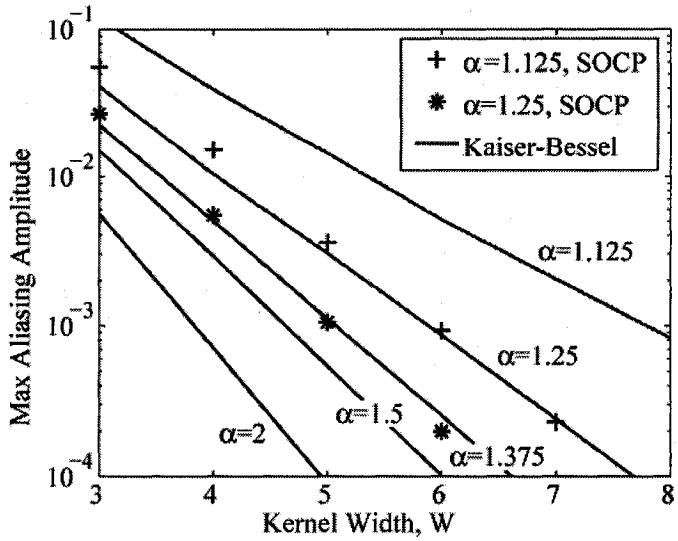


Figure 3.4: Maximum aliasing amplitude for a selection of kernels. The shape parameter for the Kaiser-Bessel kernels is found using Eq. 3.5. SOCP refers to the optimal sampled kernels found using second-order cone programming (SOCP), a technique that is discussed in detail in section 3.5.

show that the large errors in (a) and (c) can be avoided by designing the kernel for the oversampling ratio used. In the case that the errors in (a) and (c) are acceptable, one would do better to design for this target accuracy and decrease the 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 3.3. The shape parameter, β, determines the shape of the Kaiser-Bessel window. Here, I derive a simple equation which gives near optimal β for any $\alpha \in [1, 2]$. A similar equation for β 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 β values in Table 3.3. Jackson searched for the kernel that gave the lowest *average* aliasing energy over the image whereas I search for the kernel that

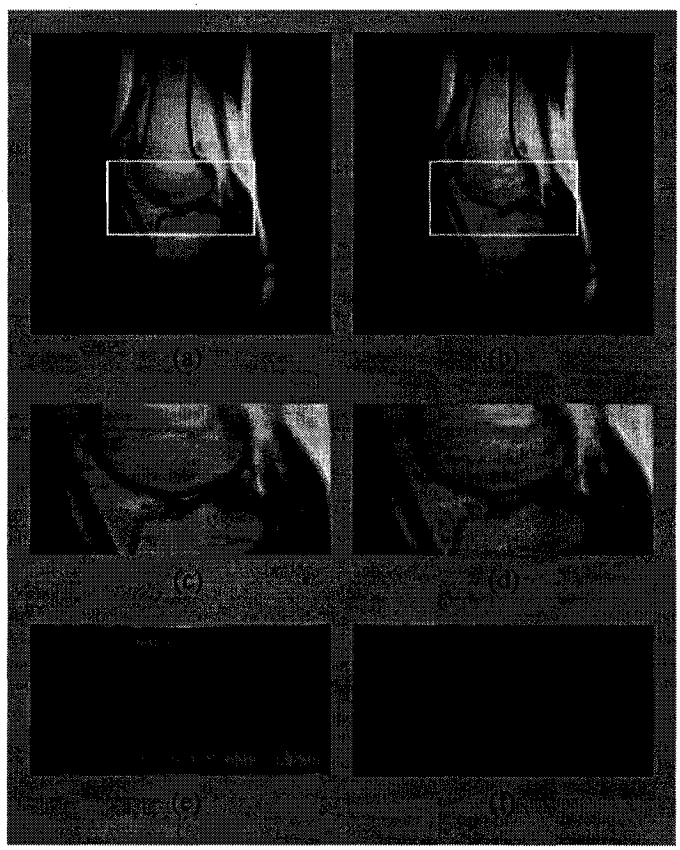


Figure 3.5: Sagittal knee images obtained with a 2D spiral sequence. (a),(c) were reconstructed using $\alpha = 1.125$ and $W = 3$, giving a maximum aliasing amplitude of about 0.1. (b),(d) were reconstructed using $\alpha = 1.25$ and $W = 4$, giving a maximum aliasing amplitude of about 0.01. When the 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.

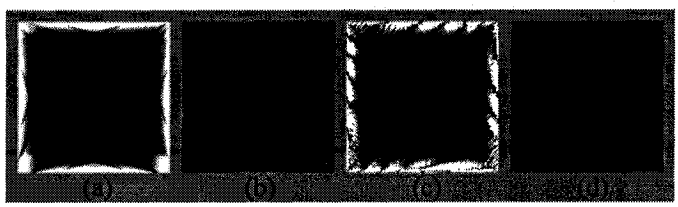


Figure 3.6: The k-space data for the numerical phantom in Fig. 2(a) was acquired using a radial trajectory in (a) and (b) and a spiral trajectory in (c) and (d). These two data sets were reconstructed on a grid with grid oversampling ratio $\alpha = 1.375$ using two different Kaiser-Bessel kernels, both of width $W = 5$. In (a) and (c), the shape parameter, β, 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 $\alpha = 1.375$ (Eq. 3.5). 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.

gives the lowest *maximum* aliasing amplitude at any point in the image. I refer to this as *optimal* in the rest of this chapter. I choose this metric because I am 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.

A good convolution kernel acts as a multi-bandpass filter in the image domain with passband $x \in [-N/2, N/2]$ and stop-bands $x \in [Gp - N/2, Gp + N/2]$, 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

$$\begin{aligned} C(k_x) &= \frac{G}{W} I_0\left(\beta\sqrt{1-(2Gk_x/W)^2}\right) \text{ for } |k_x| \leq \frac{W}{2G} \\ c(x) &= \frac{\sin\sqrt{(\pi W x/G)^2 - \beta^2}}{\sqrt{(\pi W x/G)^2 - \beta^2}} \end{aligned} \tag{3.4}$$

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

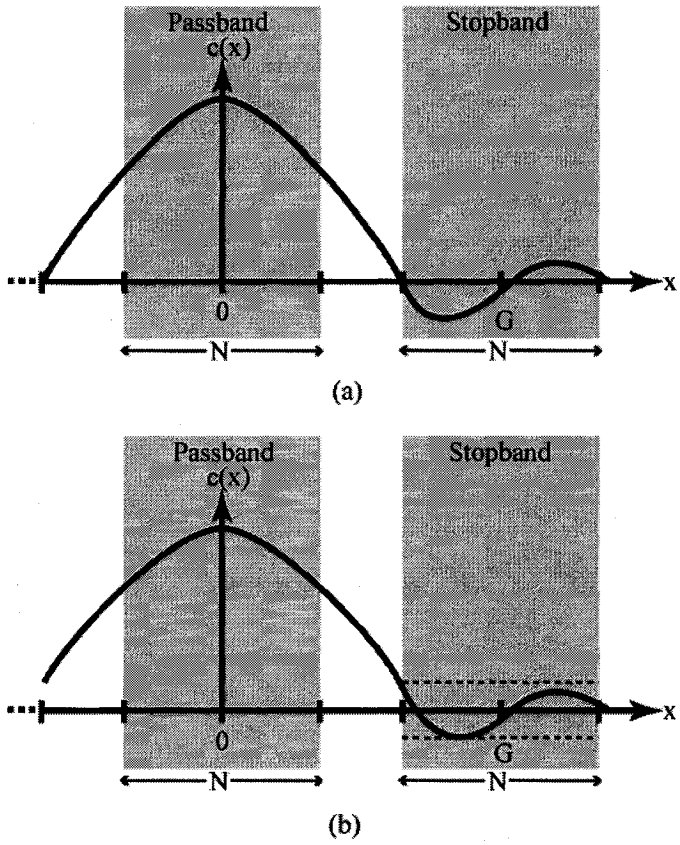


Figure 3.7: (a) Positioning the first zero crossing at the edge of the first stop-band results in a good kernel design, since it allows the main lobe to be as wide as possible without any of the main lobe contributing to the aliasing amplitude. A wide main lobe lessens apodization in the passband. Since apodization correction increases artifacts aliasing in from the stop-bands as well as the passband signal, less apodization in the passband is preferable. (b) The kernel design in (a) can be slightly improved upon by allowing the main lobe to enter slightly into the first stop-band. In this case the objective is to keep the aliasing amplitude due to the main lobe in the first stop-band equal to the aliasing amplitude caused by the first side-lobe.

Table 3.3: Comparison of β Values

	β			
	$\alpha = 1$		$\alpha = 2$	
W^a	Jackson	Eq. 3.5	Jackson	Eq. 3.5
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

[a]I use the kernel width parameter, W, which is equal to twice the window width parameter used by Jackson et al. for $\alpha = 2$.

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 stop-band, as shown in Fig. 3.7. 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 stop-band. From Eq. 3.4, the first zero of the Kaiser-Bessel window is when $\sqrt{(\pi W x/G)^2 - \beta^2} = \pi$. Setting $\beta = \pi\sqrt{W^2\left(\alpha - \frac{1}{2}\right)^2/\alpha^2 - 1}$ will position this zero crossing at $x = G - N/2$, as shown in Fig. 3.7(a). It turns out that it is possible to do slightly better if one allows some of the main lobe to enter into the first stop-band, as illustrated by Fig. 3.7(b). In essence the aliasing amplitude at the edge of the image is allowed to rise until it reaches the aliasing amplitude caused by the first side-lobe. The point at which this happens is difficult to solve analytically, however I have found that a slight modification of the equation I obtain by positioning the first zero crossing at $x = G - N/2$ works very well in practice. The modified equation is:

$$\beta = \pi\sqrt{\frac{W^2}{\alpha^2}\left(\alpha - \frac{1}{2}\right)^2 - 0.8} \tag{3.5}$$

Equation 3.5 has the advantage that it gives a β value which is very close to the optimal β value and is simple to calculate. Equation 3.5 can be used to generate a kernel that will ensure that the gridding solution is highly accurate, considering the

α and W selected. Once the α and W have been selected, the only way one has of reducing the computation time is to reduce the constant of proportionality of the DW^2 term (DW^3 term for 3D). That is, the computation time can only be reduced by decreasing 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.

3.4 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. The computer 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 $\widehat{C}(k_x)$ (Table 3.2). While it is clear that this scheme offers a significant computation time savings, it is not clear how the aliasing amplitude is affected. I demonstrate that if not done carefully, this sampling can become the dominant source of inaccuracy in the gridding algorithm. I 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 one considers the two extremes: 1) sampling with one point and using linear interpolation, the resulting kernel is a triangle kernel regardless of the kernel function being sampled; 2) 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 A provides a mathematical analysis of the aliasing amplitude of a sampled and interpolated kernel. Here I describe the results

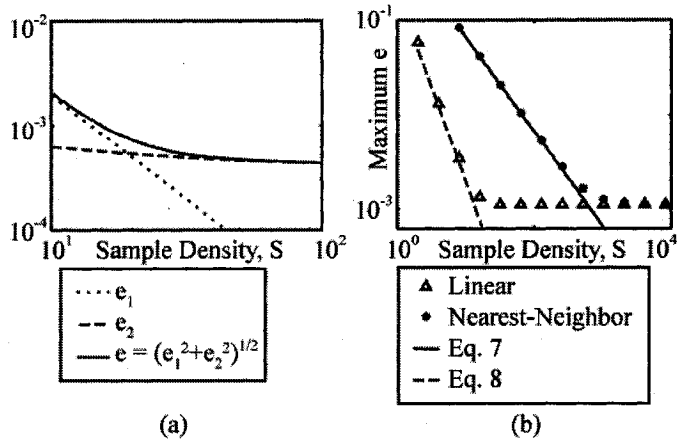


Figure 3.8: (a) Dependence of the two components, ε_1 and ε_2, of the aliasing amplitude on the kernel sampling density, S, for a sampled Kaiser-Bessel kernel with linear interpolation. The kernel width is $W = 5$ and β was obtained using Eq. 3.5. Shown here is pixel position $i = 64$, with $N = 128$, $G = 176$ ($\alpha = 1.375$). Since ε_2 has a weak dependence on the kernel sampling density, once ε_1 drops below ε_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 under-sampled, $\varepsilon \approx \varepsilon_1$ and the aliasing amplitude corresponds to Eqs. 3.7 and 3.8. 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.

of that analysis and give a simple expression for ensuring that a kernel is sampled finely enough such that its aliasing amplitude is that of the underlying kernel function being sampled.

As shown in appendix A, Eq. A.3, one useful property of the aliasing amplitude of a sampled kernel is that it can be written as the quadrature sum of two terms,

$$\varepsilon[i] = \sqrt{(\varepsilon_1[i])^2 + (\varepsilon_2[i])^2}, \tag{3.6}$$

where the first term, ε_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.

3.8(a). Roughly, one can take ε_1 to be the aliasing amplitude due to sampling of the kernel and ε_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 ε_1. Once ε_1 is an order of magnitude less than ε_2, $\varepsilon[i] \cong \varepsilon_2[i]$ and there is little benefit to further increasing the kernel sampling density, also illustrated in Fig. 3.8(a).

The fundamental difference between nearest-neighbor and linear interpolation is how quickly ε_1 decreases as the kernel sampling density is increased. For both interpolation schemes, ε_1 is greatest at the edge of the image, when $i = -N/2$. For the two interpolation schemes this maximum value of ε_1 over the image is derived in appendix A to be

$$\max(\text{nearest-neighbor } \varepsilon_1) = \frac{0.91}{\alpha S} \tag{3.7}$$

$$\max(\text{linear } \varepsilon_1) = \frac{0.37}{(\alpha S)^2}. \tag{3.8}$$

Since ε_1 is inversely proportional to the sampling density S for nearest-neighbor interpolation, a large kernel sampling density is needed to decrease ε_1 below the value of ε_2. For linear interpolation, ε_1 is inversely proportional to S^2, and requires a far smaller kernel sampling density for a comparable decrease in ε_1, as shown in Fig. 3.8(b).

This point can be illustrated with a simple design example; the objective is 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. 3.4, 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, one can 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 Eq. 7 and 8 can be used to predict the kernel accuracy, as shown in Fig.

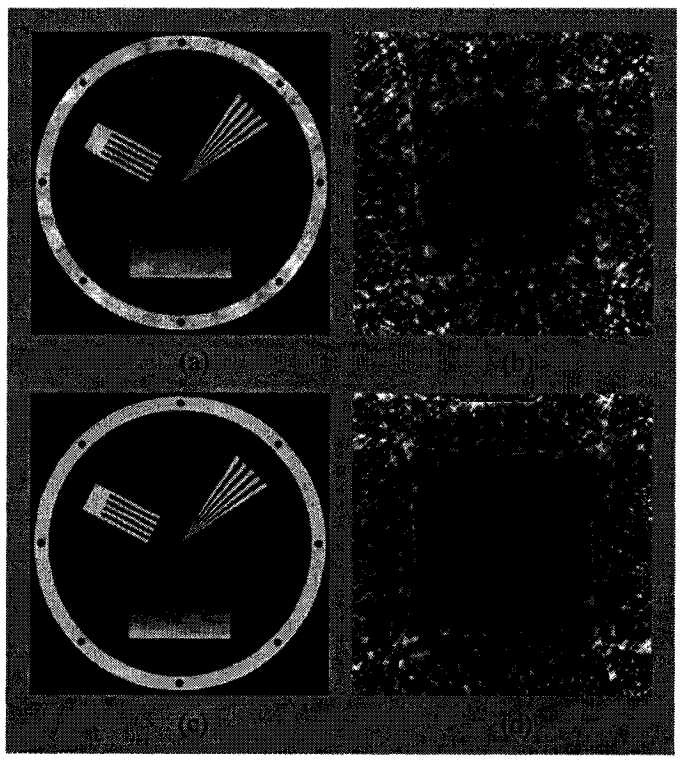


Figure 3.9: Two common convolution kernels, the nearest-neighbor kernel and the triangle kernel, are simple cases of presampled and interpolated kernels. In both of these cases, the kernel sampling density, $S = 1$. From Eq. A.5 in appendix A, when $S = 1$, $\varepsilon_2 = 0$ and so $\varepsilon = \varepsilon_1$. (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 Eqs. 3.7 and 3.8, the maximum error for the nearest-neighbor kernel should be about 5 times that of the triangle kernel. Comparing (b) and (d) shows that this is the case.

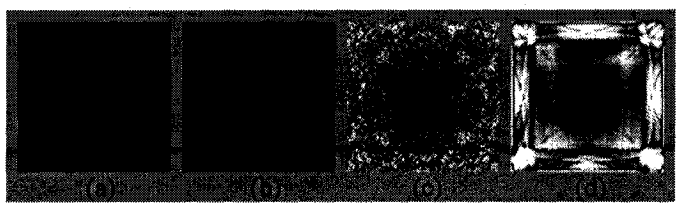


Figure 3.10: Numerical phantom in Fig. 2(a) acquired with a radial trajectory. Shown is the image reconstruction error resulting from different gridding parameters. (a) $\alpha = 2$, Kaiser-Bessel kernel, $W = 6$. (b) Same parameters as (a), but now the kernel is sampled with 600 points and linear interpolation is used. This gives no increase in the reconstruction error. (c) Same kernel as (a), but here the kernel is sampled with *3000 points* and nearest-neighbor interpolation is used. Even though far more points are sampled, 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).

3.9.

I 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. Figure 3.10 highlights this concept. With an oversampling ratio of two and a kernel width of six, sampling with 3000 points and using nearest-neighbor interpolation (c) leads to a significant increase in error over the unsampled kernel (a) and oversampling with 600 points and using linear interpolation (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, (d), since this has the same level of error.

For the Kaiser-Bessel kernel, Eq. 3.4 can be used for apodization correction. When the kernel is sampled and interpolated, the apodization correction function should change accordingly. In appendix B, I give an accurate method for determining

the apodization correction from the kernel sample values.

3.5 Choosing the Optimal Sampled Kernel

Thus far the sampled kernel has been designed 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 one can design the sample values directly to achieve the minimum value for the maximum aliasing amplitude for a given oversampling ratio.

Once the interpolation method has been chosen, the kernel sampling density and oversampling ratio ε_1 is fixed and modifying the sample values can only reduce ε_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 A.5 in appendix A gives the following expression for ε_2 in terms of the inverse Fourier transform of $C_s(k_x)$:

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

where $\widehat{h}(x)$ is a simple function of $h(x)$ as shown in appendix A. From this equation, one can write the following second-order cone program (SOCP) [16]:

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

This SOCP finds a kernel, $C_s(k_x)$, if it exists, such that the maximum value of $\varepsilon_2[i] \leq$

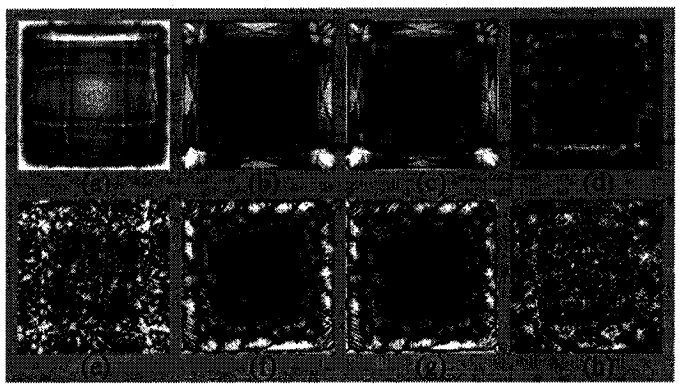


Figure 3.11: Comparison of the image reconstruction error for four different kernels. (a)-(d) used a radial acquisition and (e)-(h) used a spiral acquisition; the object sampled is shown in Fig. 2(a). In the first column, a width four Kaiser-Bessel kernel was used on a grid with an oversampling ratio 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 3.5). 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).

$\varepsilon_2^{\max}$. See appendix C for a MATLAB implementation of this SOCP. By decreasing $\varepsilon_2^{\max}$ until the smallest value is found where a solution exists, the optimal sampled kernel is found.

The disadvantage of this method is that a series of SOCP problems must be solved for every α, 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 Fig. 2 and 3.11.

For the examples given in Fig. 3.11, the maximum reconstruction error of the optimal sampled kernel (d),(h) is 75% of the maximum reconstruction error of the

comparable *unsampled* Kaiser-Bessel kernel (b),(f). This leads to the surprising notion that sampling the kernel can actually lead to lower errors in practice than if one 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 allows the use of the *design* method developed above, which gives a kernel design not readily available when a continuous approach is taken.

3.6 The Block Storage Format

In practice, the amount of time it takes to complete an image reconstruction computation on a modern computer is dependent on both the amount of computation that must be performed and the memory bandwidth and latency. Using a reduced oversampling ratio and a sampled kernel reduces the computation time by reducing the amount of computation that must be performed. These techniques are so successful that even for moderately sized 3-D images of $128 \times 128 \times 128$, the computation time is not limited by the ability of the central processing unit (CPU) of the computer to perform the necessary calculations, but by the memory bandwidth and latency: the time it takes to get data from the computer memory to the CPU for processing. This section develops the block storage format, which is a reorganization of the grid data in the computer memory. The block storage format reduces the traffic between the CPU and memory, reducing the computation time approximately four-fold when reconstructing a $256 \times 256 \times 256$ image.

In standard practice, the grid data is stored in a line-by-line storage format as shown in Fig. 3.12. In this format, the data for each line of the 3-D grid is stored contiguously in memory. Figure 3.12 also shows the block storage format, where data for a small 3-D block of the grid is stored contiguously in memory. In the current implementation $8 \times 8 \times 8$ voxel blocks are used, corresponding to one page of memory. In the line-by-line format, data is very spread out in memory when traversing the data perpendicular to the lines. By storing the data in the block storage format, data likely to be accessed concurrently is put in the same page of memory. Numerical experiments were performed using two 3-D trajectories: the spiral-PR trajectory

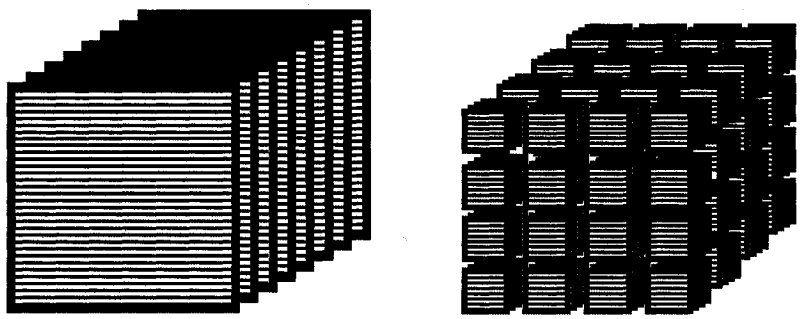


Figure 3.12: Block storage stores the same information as the common line-by-line format, but the information is organized differently in the computer memory. In the current implementation, $8 \times 8 \times 8$ blocks of grid locations are stored concurrently in memory.

and the cones trajectory, both illustrated in Fig. 3.13. Figure 3.14 shows how the number of pages accessed during the gridding process is reduced by using the block storage format, speeding up the entire reconstruction. Gridding was performed with a kernel width of four and oversampling ratio of 1.25. In addition, with the line-by-line storage format, the computation time is very sensitive to the orientation of the grid. For example, when the spirals are rotated about the contiguous-line direction to form the spiral-PR trajectory, far fewer memory pages need to be accessed than when the spirals are rotated around an axis normal to the contiguous-line direction. Block storage is immune to this effect.

In order to perform the FFT while the data is stored in the block format, one can simply copy a row of blocks to a buffer, perform a 1-D FFT, using fftw, and copy the result back, as shown in Fig. 3.15. Repeating this for all three dimensions gives a 3-D FFT. Performing the FFT in this way is 4 times faster than performing a 3-D FFT on data stored in a line-by-line format, for a $320 \times 320 \times 320$ grid. Figure 3.16 shows the FFT computation time for the grid stored using both line-by-line format and the block storage format. Less dependent on the CPU-memory traffic constraints, the computation time for the FFT with block storage format varies consistently with the

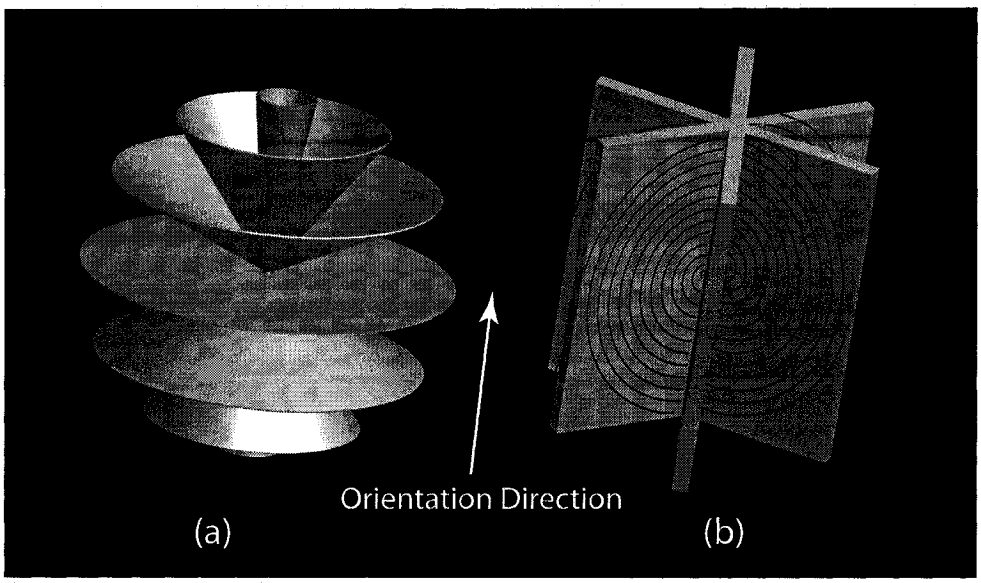


Figure 3.13: (a) Cones trajectory. (b) Spiral-PR trajectory.

amount of computation, roughly $N^3 \log N$.

3.7 FOV and Resolution Flexibility

In the gridding algorithm, data is convolved with a gridding kernel and the result is sampled onto a Cartesian grid. Since the sampling onto the grid causes a repetition in image space, the gridding kernel must be chosen to suppress these repetitions from aliasing back into the image. Usually, the FOV is set to the FOV of the object and the kernel is designed to suppress artifacts from the point-spread-function associated with the k-space trajectory. As shown in Fig. 3.17, if an FOV smaller than the object is chosen, it is possible to use a gridding kernel for this smaller FOV to suppress other parts of the *object* (in addition to artifacts from the point-spread-function) from aliasing into the smaller FOV. Choosing the appropriate part of the image to reconstruct is accomplished by adding a linear phase in k-space, corresponding to a shift in image space.

With the gridding method of reconstruction, the FOV and resolution of the final

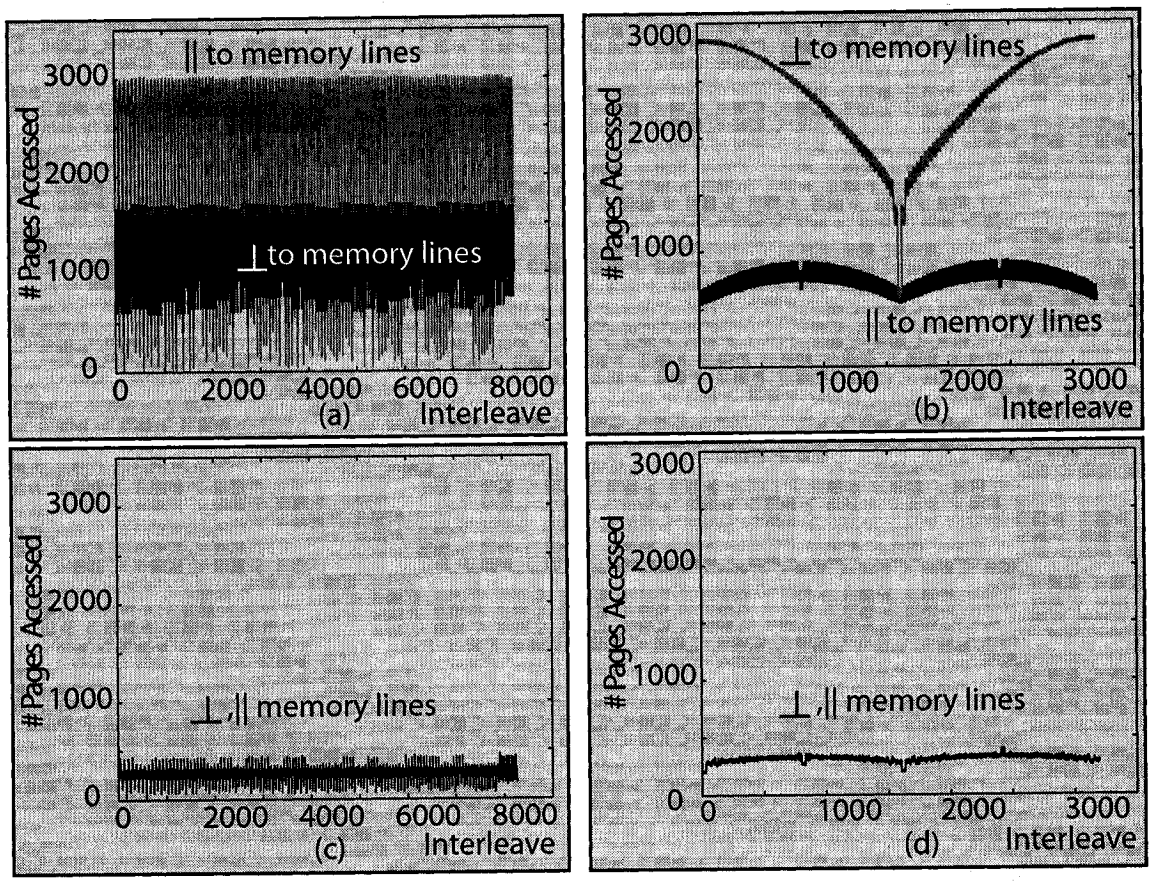


Figure 3.14: Far fewer pages of memory need to be accessed while gridding an interleave when block storage is used instead of line-by-line storage. Plots of the number of memory pages accessed for each interleave are shown for (a) cones trajectory with line-by-line storage, (b) spiral-PR trajectory with line-by-line storage, (c) cones trajectory with block storage, (d) spiral-PR trajectory with block storage. For the line-by-line format, the number of pages accessed depends on whether the orientation direction of the trajectory (Fig. 3.13) is parallel or perpendicular to the orientation of the memory lines. When block storage is used, the number of pages accessed is very immune to the orientation direction of the trajectory.

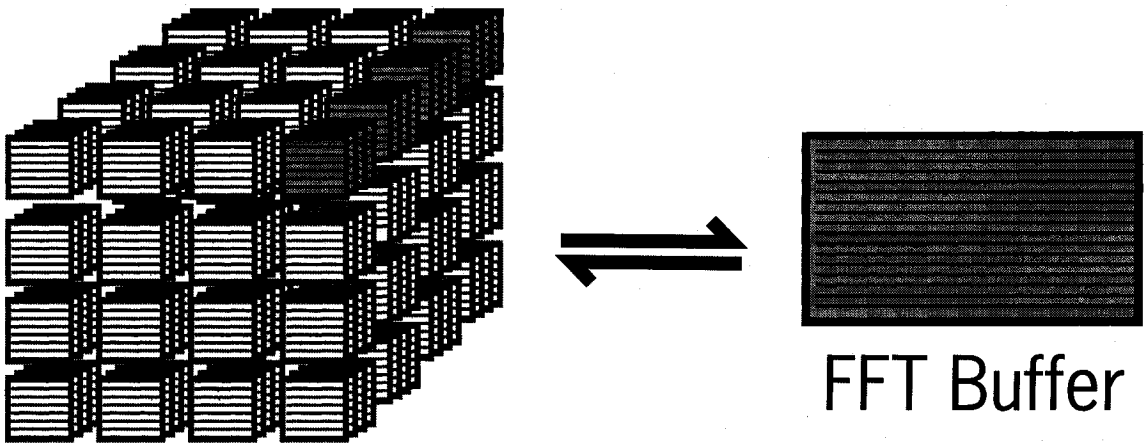


Figure 3.15: When the grid is stored in block format, the FFT is performed in each direction (x, y, z) separately. Blocks in the direction along which the FFT is to be performed are copied to an FFT Buffer, converting these blocks to line-by-line format. The FFT is performed on these lines using FFTW and then the lines are copied back to block storage.

image are not tied to the FOV or resolution of the k-space trajectory used to acquire the data. This makes gridding a very flexible reconstruction method. In applications such as cardiac imaging, where the region of interest (the heart) is smaller than the FOV of the k-space trajectory (the chest), reconstruction time can be reduced by using a reduced FOV. Since the FOV can be easily shifted by applying a linear phase shift, gridding makes it easy to implement reconstruction methods such as PILS [17] that rely on localized coil sensitivities. To balance the competing objectives of high resolution, good signal-to-noise ratio and fast imaging, often the image space resolution of the acquired data is chosen to be different along the x, y, and z dimensions. When this acquired data is reconstructed using gridding, the resolution of the grid can be chosen to be equal in all dimensions. This provides a simple way to achieve the correct aspect ratio regardless of how the data is acquired. By separating the choice of FOV and resolution for the final image from the design choices of the k-space trajectory, gridding simplifies the imaging process.

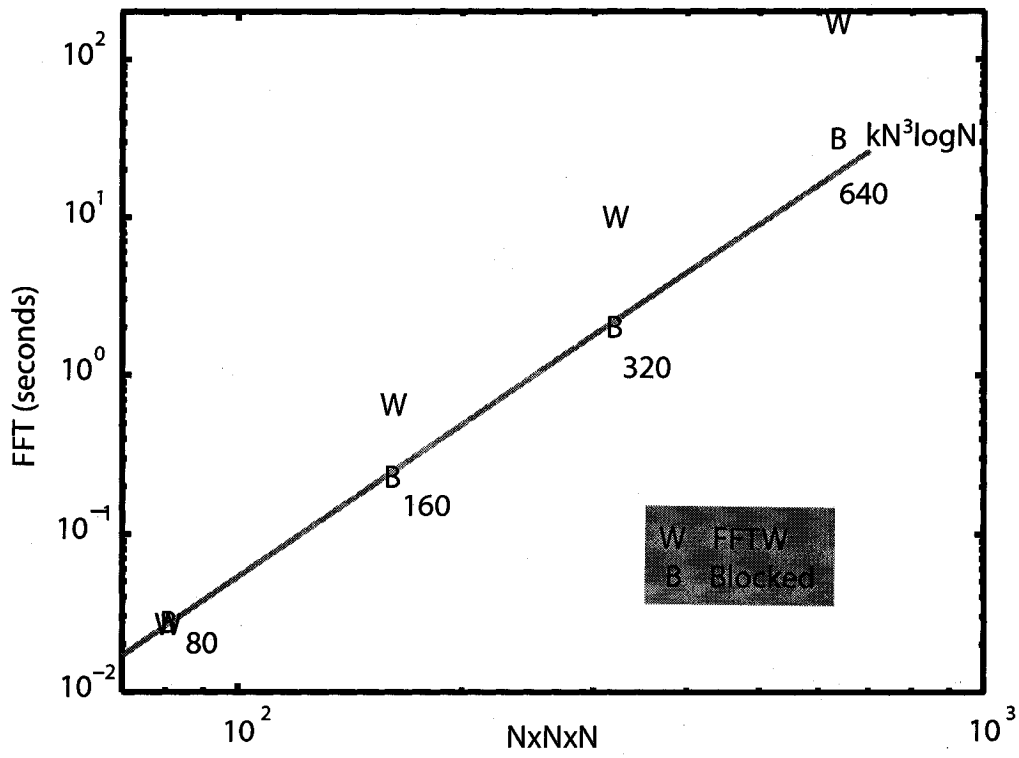


Figure 3.16: Time required to compute a 3-D FFT plotted against FFT size. The computation grows as $N^3 \log N$, where N is the size of the FFT along one dimension. When $N = 80$, the time to compute the FFT is similar between line-by-line FFTW and when block storage format is used. As the size of the FFT increases, the computation time grows as $N^3 \log N$ when the block storage format is used. When the line-by-line format is used, the computation time grows at a faster rate as memory management becomes an issue.

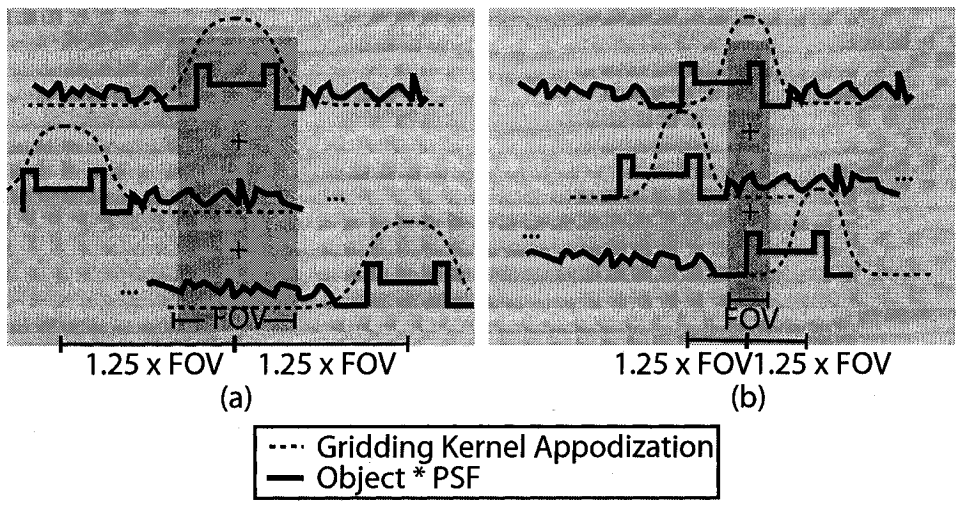


Figure 3.17: Sampling onto a grid causes the image convolved with the point-spread-function (PSF) to be repeated and summed every $1.25 \times \text{FOV}$, for a grid oversampling ratio of 1.25. The gridding kernel is designed to suppress these repetitions from entering into the FOV. Typically, the FOV is set to capture the entire object (a), but can be set to capture a part of the object (b), allowing the use of a smaller grid.

3.8 Conclusion

In this chapter, a number of technical refinements to the basic gridding algorithm are brought together. These include using a minimal oversampling ratio, presampling the kernel, two methods for designing high performance gridding kernels, the use of the block storage format and flexibility in specifying the field-of-view for the reconstructed image. All of these developments have been placed in a consistent framework and I have 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 3D imaging, where a thirty-fold reduction in computation time and three-fold reduction in memory requirements can be achieved.

Chapter 4

Anti-aliasing Partially Parallel Encoded Acquisition Reconstruction

4.1 Introduction

In magnetic resonance imaging (MRI), it is well known that multiple receiver coils can be used to reduce the gradient encoding required and, consequently, the time needed to acquire an image. This is because multiple receiver coils enable the image to be encoded in parallel—at each sample time each receiver collects a differently encoded datum. The acquired data are the result of encoding with both the gradients and the spatial sensitivity of the receiver coils (coil sensitivities) and no longer correspond to samples in a common k-space.

Over the past few years, many reconstruction methods that take advantage of sensitivity encoding have been developed and improved. The original SMASH method [18] is limited to Cartesian k-space trajectories and places requirements on the coil sensitivity functions that are difficult to achieve in practice. SENSitivity Encoding (SENSE) theory [19, 20] relaxes the requirement that the coil sensitivities have specific profiles and provides an iterative method for reconstructing arbitrary k-space trajectories. The SPACE-RIP [21] method allows non-iterative reconstruction for

k-space trajectories which do not fall on a Cartesian grid in one dimension, while the recently introduced PARS method [22] can reconstruct one, two and three dimensional arbitrary k-space trajectories without iteration. All of the above methods require the coil sensitivity functions to be found to a high degree of accuracy. Finding the coil sensitivities can be accomplished by performing a full field-of-view (FOV) initial calibration scan before acquiring the image data [19] or by designing the k-space trajectory such that a low resolution full FOV scan can be extracted from the acquired data [23]. In practice it is difficult to obtain the coil sensitivity functions without errors and, even when small errors exist in the coil sensitivity functions used in the reconstruction process, these errors can lead to visible image artifacts.

PILS [17] is a non-iterative method that does not require the coil sensitivities to be known with great accuracy and works with arbitrary k-space trajectories. However, the PILS method is only viable when the coil sensitivities are sufficiently localized in space.

GRAPPA, both single-column [24] and multi-column [25] versions, as well as predecessors AUTO-SMASH [26] and VD-AUTO-SMASH [27], uses an autocalibration technique that does not require the coil sensitivities to be known and avoids problems in coil sensitivity estimation that affect the previous methods. For Cartesian trajectories, autocalibration is able to remove the aliasing artifacts caused by reduced gradient encoding. While the autocalibration technique has been extended to specific non-Cartesian trajectories [28, 29, 30, 31, 32, 33], in doing so, it loses some of its ability to successfully remove all of the aliasing artifacts.

In this chapter, I develop a new calibration technique which I call the *local projection calibration* technique. I show that this new technique is able to remove the aliasing artifacts from arbitrary k-space trajectories, without needing to estimate the coil sensitivity functions.

In their development of SENSE theory [19], Pruessmann *et al.* provide a general formulation for encoding with coil sensitivities. In this chapter, I extend the linear algebra framework of SENSE to develop the local projection calibration technique. Using this linear algebra framework, I show that the local projection calibration technique is fundamentally different from techniques that use low resolution images to

construct coil sensitivity estimations. Moreover, the local projection calibration technique avoids the main difficulties in estimating coil sensitivities from low resolution data: Gibbs ringing distortion and an inability to deal with sensitivity maps with high frequency content.

Finally, I describe the APPEAR (Anti-aliasing Partially Parallel Encoded Acquisition Reconstruction) method for reconstructing parallel encoded acquisitions as a practical method that uses the local projection calibration technique to remove the aliasing artifacts caused by reduced gradient encoding. The non-iterative APPEAR method is unique in being able to take advantage of the local projection calibration technique in reconstructing arbitrary k-space trajectories. I show phantom and *in vivo* reconstruction results using APPEAR for trajectories which are non-Cartesian in the one phase-encode dimension.

4.2 Theory

In this section, the pertinent SENSE theory is reviewed and then extended to provide an understanding of the local projection calibration technique. The APPEAR method is introduced, demonstrating how local projection calibration can be used to reconstruct images from arbitrary k-space sampling.

4.2.1 SENSE

In MRI, data acquisition can be viewed as analyzing the spatially varying transverse magnetization, $m(\mathbf{r})$, by projecting it onto a collection of encoding functions which take the form $e_j(\mathbf{k},\mathbf{r}) = s_j(\mathbf{r})g(\mathbf{k},\mathbf{r})$ where $s_j(\mathbf{r})$ is the sensitivity of receiver coil j and $g(\mathbf{k},\mathbf{r}) = \exp(-i2\pi\mathbf{k}^T\mathbf{r})$ is the gradient encoding function for location $\mathbf{k}$, as described in Table 4.1. When only one receiver coil is used, each encoding function can be uniquely identified by a specific location in k-space. When using multiple coils, the notion of location can be extended by defining an *encoding location* to be the pair of values, $(j,\mathbf{k})$, consisting of an integer coil index j and continuous-valued three-tuple k-space location $\mathbf{k}$, that uniquely identifies an encoding function. The projection of

Table 4.1: General Variables and Functions

$\mathbf{r}$	Spatial location. 3×1
j	Coil index.
$\mathbf{k}$	k-space location. 3×1
$m(\mathbf{r})$	Magnetization being analyzed.
$s_j(\mathbf{r})$	Sensitivity of coil j.
$g(\mathbf{k}, \mathbf{r})$	Gradient encoding function for k-space location $\mathbf{k}$. $g(\mathbf{k}, \mathbf{r}) = \exp\left(-i2\pi\mathbf{k}^T\mathbf{r}\right)$
$e_j(\mathbf{k}, \mathbf{r})$	Encoding function. $e_j(\mathbf{k}, \mathbf{r}) = s_j(\mathbf{r})g(\mathbf{k}, \mathbf{r})$
$(j, \mathbf{k})$	Encoding location, coil j at k-space location $\mathbf{k}$.
$d_j(\mathbf{k})$	Data acquired at encoding location $(j, \mathbf{k})$.
$\mathcal{A}$	Acquisition set. Contains all encoding locations of data (from all coils) acquired in the scan.
$N_{\mathcal{A}}$	Number of elements in $\mathcal{A}$ (also number of data values acquired in scan).
N_v	Number of voxels in reconstructed image.
N_c	Number of receiver coils.
$\mathbf{d}_{\mathcal{A}}$	Vector containing acquired data values. $N_{\mathcal{A}} \times 1$
$\mathbf{m}$	Vector containing voxel values of image being analyzed. $N_v \times 1$
$\mathbf{e}_j(\mathbf{k})$	Encoding vector. $N_v \times 1$
$\mathbf{s}_j$	Sensitivity vector. $N_v \times 1$
$\mathbf{g}(\mathbf{k})$	Gradient encoding vector. $N_v \times 1$
$E_{\mathcal{A}}$	Encoding matrix for acquired data. $N_{\mathcal{A}} \times N_v$ Each row consists of $\mathbf{e}_j^T(\mathbf{k})$ for a $(j, \mathbf{k}) \in \mathcal{A}$.
$R_{\mathcal{A}}$	SENSE reconstruction matrix for acquired data. $N_v \times N_{\mathcal{A}}$
$\widehat{\mathbf{m}}$	Vector containing reconstructed voxel values. $N_v \times 1$
$\mathrm{DSFT}\{\mathbf{a}\}(\mathbf{k})$	Discrete space Fourier transform. $\mathrm{DSFT}\{\mathbf{a}\}(\mathbf{k}) = \mathbf{g}^T(\mathbf{k})\mathbf{a}$
$\mathbf{a} \cdot \mathbf{b}$	Hadamard or entry-wise vector product.

$m(\mathbf{r})$ onto an encoding function is accomplished by the integral

$$d_j(\mathbf{k}) = \int_V e_j(\mathbf{k}, \mathbf{r}) m(\mathbf{r})\, d\mathbf{r}, \tag{4.1}$$

where $d_j(\mathbf{k})$ is the resulting datum acquired at encoding location $(j, \mathbf{k})$.

Reconstruction can be viewed as an attempt to synthesize the magnetization image from the results of this analysis. Typically, a finite number of data points are acquired and a finite number of voxel values are reconstructed. The *acquisition set*, $\mathcal{A}$, contains the encoding locations, from all coils, acquired in the scan,

$$\mathcal{A} = \{(j, \mathbf{k}) | d_j(\mathbf{k}) \text{ was acquired in the scan}\}, \tag{4.2}$$

and the acquired data can be assembled in the vector $\mathbf{d}_\mathcal{A}$. Since the encoding is linear, a linear reconstruction is appropriate, which can be written as

$$\widehat{\mathbf{m}} = R_\mathcal{A} \mathbf{d}_\mathcal{A}, \tag{4.3}$$

where the vector $\widehat{\mathbf{m}}$ contains the reconstruction estimate of the voxel values and $R_\mathcal{A}$ is the reconstruction matrix, each row of which synthesizes an image voxel value from the acquired analysis data.

Pruessmann *et al.* showed that when the magnetization, $m(\mathbf{r})$, can be approximated with a Dirac delta function at the center of each voxel, the reconstruction matrix, $R_\mathcal{A}$, that minimizes the noise in the reconstructed image can be expressed as

$$R_\mathcal{A} = E_\mathcal{A}^\dagger, \tag{4.4}$$

where $E_\mathcal{A}^\dagger = \left(E_\mathcal{A}^H \Psi_\mathcal{A}^{-1} E_\mathcal{A}\right)^{-1} E_\mathcal{A}^H \Psi_\mathcal{A}^{-1}$ is a pseudo-inverse of the encoding matrix $E_\mathcal{A}$. I will discuss $E_\mathcal{A}$ in more detail shortly. $\Psi_\mathcal{A}$, the sample noise matrix, is included to calculate the pseudo inverse that minimizes the noise in the reconstructed image. In general, this chapter assumes the Dirac delta voxel approximation, allowing functions of $\mathbf{r}$ to be written as vectors with a total number of elements of N_v, the total number

Table 4.2: Data Synthesis Variables

$\widehat{d}_j(\mathbf{k})$	Synthesized datum at location $(j, \mathbf{k})$. Linear combination of data values.
$w_{j,j',\mathcal{A}}(\mathbf{k}, \mathbf{k}')$	Linear combination weight applied to $d_{j'}(\mathbf{k}')$ when synthesizing $\widehat{d}_j(\mathbf{k})$ from data values acquired at locations in $\mathcal{A}$.
$\mathbf{w}_{j,\mathcal{A}}(\mathbf{k})$	Vector containing all weights to synthesize $\widehat{d}_j(\mathbf{k})$ from data values acquired at locations in $\mathcal{A}$. $N_{\mathcal{A}} \times 1$
$\widehat{\mathbf{e}}_{j,\mathcal{A}}(\mathbf{k})$	Estimated encoding vector for location $(j, \mathbf{k})$. $N_v \times 1$ Linear combination of encoding vectors at locations in $\mathcal{A}$.
$\boldsymbol{\epsilon}_{j,\mathcal{A}}(\mathbf{k})$	Error in estimated encoding vector $\widehat{\mathbf{e}}_{j,\mathcal{A}}(\mathbf{k})$.

of voxels. With this discretization of the magnetization, Eq. 4.1 can be written as

$$d_j(\mathbf{k}) = \mathbf{e}_j^T(\mathbf{k})\mathbf{m}, \tag{4.5}$$

where $\mathbf{m}$ is the $N_v \times 1$ magnetization vector, containing the magnetizations of the voxels, and $\mathbf{e}_j(\mathbf{k})$ is the $N_v \times 1$ encoding vector for encoding location $(j, \mathbf{k})$. Note that the encoding vector can still be broken up into the vector representation of the sensitivity and gradient encoding terms, i.e., $\mathbf{e}_j(\mathbf{k}) = \mathbf{s}_j \cdot \mathbf{g}(\mathbf{k})$, where $\mathbf{s}_j$ is multiplied by $\mathbf{g}(\mathbf{k})$ using the Hadamard, or entry-wise, product.

Each row of the encoding matrix, $E_{\mathcal{A}}$, consists of an encoding vector in the acquisition set such that

$$\mathbf{d}_{\mathcal{A}} = E_{\mathcal{A}}\mathbf{m}. \tag{4.6}$$

SENSE theory assumes that the coil sensitivities are known, allowing the encoding matrix $E_{\mathcal{A}}$ and any encoding vector $\mathbf{e}_j(\mathbf{k})$ to be calculated.

When the magnetization is fully encoded, the set of encoding vectors comprising $E_{\mathcal{A}}$ span the space of the magnetization vector $\mathbf{m}$. In this case, Eq. 4.4 can be used to find the reconstruction matrix and the image can be reconstructed using Eq. 4.3. Since the final objective of any reconstruction method is to reconstruct the image, this is usually how SENSE theory is employed.

As shown later, the APPEAR method involves synthesis of unacquired data. Due to this property of APPEAR, it is helpful at this point to note that SENSE theory can

be employed to synthesize, from the acquired data, any datum $\widehat{d}_j(\mathbf{k})$ for an unacquired location $\mathbf{k}$ within the extent of acquired k-space. As SENSE theory assumes that the coil sensitivities are known (and hence the encoding functions are known), the magnetization vector in Eq. 4.5 can be replaced with the SENSE estimate of the magnetization from Eq. 4.3:

$$\begin{aligned} \widehat{d}_j(\mathbf{k}) &= \mathbf{e}_j^T(\mathbf{k})\widehat{\mathbf{m}} && (4.7)\\ &= \mathbf{e}_j^T(\mathbf{k})R_{\mathcal{A}}\mathbf{d}_{\mathcal{A}} && (4.8)\\ &= \sum_{(j',\mathbf{k}')\in\mathcal{A}} w_{j,j',\mathcal{A}}(\mathbf{k},\mathbf{k}')d_{j'}(\mathbf{k}'), && (4.9) \end{aligned}$$

where $w_{j,j',\mathcal{A}}(\mathbf{k},\mathbf{k}')$ is a complex scalar value formed by multiplying $\mathbf{e}_j^T(\mathbf{k})$ by the column of $R_{\mathcal{A}}$ corresponding to the encoding location $(j',\mathbf{k}')$. From Eq. 4.9 it is clear that the estimated datum, $\widehat{d}_j(\mathbf{k})$, is simply a linear combination of the acquired data. The linear combination weights can be denoted by the vector $\mathbf{w}_{j,\mathcal{A}}^T(\mathbf{k}) = \mathbf{e}_j^T(\mathbf{k})R_{\mathcal{A}}$, allowing Eq. 4.9 to be rewritten as

$$\widehat{d}_j(\mathbf{k}) = \mathbf{w}_{j,\mathcal{A}}^T(\mathbf{k})\mathbf{d}_{\mathcal{A}}. \quad (4.10)$$

While Eq. 4.10 is very useful for showing how an unacquired datum can be synthesized from the acquired data, it does not provide much intuition as to the quality of the synthesis. A better intuition can be obtained by expressing the synthesized datum in terms of an estimated encoding vector. This can be accomplished by substituting Eq. 4.6 into Eq. 4.10:

$$\widehat{d}_j(\mathbf{k}) = \mathbf{w}_{j,\mathcal{A}}^T(\mathbf{k})E_{\mathcal{A}}\mathbf{m} = \widehat{\mathbf{e}}_{j,\mathcal{A}}^T(\mathbf{k})\mathbf{m}. \quad (4.11)$$

The estimated encoding vector $\widehat{\mathbf{e}}_{j,\mathcal{A}}(\mathbf{k})$ is a linear combination of the encoding vectors of the acquired data

$$\widehat{\mathbf{e}}_{j,\mathcal{A}}^T(\mathbf{k}) = \mathbf{w}_{j,\mathcal{A}}^T(\mathbf{k})E_{\mathcal{A}}. \quad (4.12)$$

Since I use the idea of an estimated encoding vector extensively as I develop the theory for local projection calibration, it is useful to discuss it in further detail.

While both the magnetization and the encoding vector are written as vectors to take advantage of matrix notation, both are best viewed as images. Acquiring a datum as in Eq. 4.5 can then be viewed as multiplying the magnetization image by an encoding vector image and summing all of the resultant voxel values. By expressing a synthesized datum in terms of an estimated encoding vector, as in Eq. 4.11, the synthesized datum can also be viewed as the result of summing the multiplication of two images (the magnetization image and the estimated encoding vector image). If the estimated encoding vector for an encoding location is not the same as the true encoding vector for that location, the synthesized datum can be in error. This can be made more concrete by defining the error in the estimated encoding vector as the vector

$$\boldsymbol{\epsilon}^T_{j,\mathcal{A}}(\mathbf{k}) = \widehat{\mathbf{e}}^T_{j,\mathcal{A}}(\mathbf{k}) - \mathbf{e}^T_j(\mathbf{k}). \tag{4.13}$$

The error in the estimated encoding vector can also be viewed as an image and the error in the synthesized datum can then be calculated by multiplying the magnetization image by this error image and summing the voxel values. By Parseval's theorem, the total energy of the artifacts in image space is equal to the total energy of all of the data synthesis errors. Although this does not give an indication of the level of structure in the artifacts, it points out that reducing the errors of the estimated encoding vectors directly reduces the energy of the artifacts in the reconstructed image. To ensure accurate synthesis, the error in an estimated encoding vector image should be zero, or sufficiently close to zero, at any voxel locations that contain magnetization. Note, however, that an estimated encoding vector and its associated error vector are dependent solely on the linear combination weights and the encoding vectors of the acquired data; they do not depend on the magnetization itself. Thus, the error in the estimated encoding vector, which can be viewed as an image, gives us the ability to see the quality of a set of linear combination weights independently from the magnetization being imaged.

When the magnetization is fully encoded, and the coil sensitivity functions known, the linear combination weights calculated by SENSE give a perfect estimate of the encoding vector. Since typically the magnetization is over-encoded, with more data

acquired than voxels reconstructed, the SENSE weights are not the only weights that would give a perfect estimate of the encoding vector. However, when the acquired data contains noise, these SENSE weights do give the linear combination that minimizes the noise in the synthesized datum.

As can be seen from Eq. 4.10, when SENSE is used to synthesize data at unacquired k-space locations, it uses a linear combination, similar to GRAPPA, PARS and APPEAR. Since all of these methods can be expressed as procedures for synthesizing unacquired data using a linear combination, the differences in image quality between these methods can be attributed to different choices for linear combination weight values. When the coil sensitivities are known, the weights calculated by SENSE allow the encoding vectors to be perfectly estimated, resulting in removal of aliasing artifacts. However, when errors exist in the coil sensitivity estimates, these errors are propagated to the weights calculated by SENSE, leading to imperfect estimations of the encoding vectors and possible residual aliasing artifacts. In the next section I show how APPEAR uses local projection calibration to obtain linear combination weights without estimating the coil sensitivities, and that these weights give estimated encoding vectors with very little error.

4.2.2 APPEAR

APPEAR generates the linear combination weights needed to synthesize unacquired data from data acquired on all of the coils by using *local projection calibration* together with *central region interpolation*, both of which are described in this section. Using these linear combination weights, unacquired data can be synthesized—a process I refer to as *local projection interpolation*. The current implementation of APPEAR synthesizes data across a full-FOV grid for each coil; separate coil images can be obtained by computing the Fourier transform of each grid. These coil images can then be combined to form a composite image; currently this combination is done using sum-of-squares.

In the first part of this section, I develop the local projection calibration technique and show that the technique is able to find high quality linear combination weights.

While the local projection calibration technique does not assume that the coil sensitivity functions are known, it does make the assumption that within a central region of k-space, the data values are known or can be found for every encoding location $(j, \mathbf{k})$. Since invariably only a finite amount of data is acquired, one must be able to synthesize appropriate values for unacquired locations within the central region.

In the second part of this section, I develop the idea of central region interpolation, whereby a data value at any encoding location within a sufficiently sampled central region of k-space can be synthesized from only the data acquired on the same coil. The APPEAR method uses central region interpolation to synthesize data at encoding locations *within* the central region as needed to perform local projection calibration. In turn, local projection calibration generates linear combination weights used to synthesize unacquired data throughout k-space using data acquired from all coils.

Local Projection Calibration

Despite Eq. 4.12, a small subset of the acquired encoding vectors (a few rows of $E_{\mathcal{A}}$) is sufficient for estimating an unacquired encoding vector, when the chosen subset of acquired encoding vectors are local to the k-space location of the encoding vector to be estimated. By limiting the number of acquired encoding vectors used in the estimate, local projection calibration can find high-accuracy weights using the sufficiently sampled central region of k-space. When linear combination weights are found in this way, the estimated encoding vector is a projection of the unacquired encoding vector onto the subspace spanned by the local acquired encoding vectors. Self-calibrating PARS [22] also synthesizes an unacquired datum from acquired data in a limited local neighborhood in k-space, and uses a sufficiently sampled central region of k-space for calibration. However, the self-calibration technique [23] used by PARS, which uses the sufficiently sampled central region to estimate the coil sensitivities, is different from the local projection calibration technique (which does not estimate the coil sensitivities) and generates a different set of linear combination weights. In this section, I develop the local projection calibration technique, leaving a comparison between self-calibrating PARS and local projection calibration for the discussion section.

I start by introducing some notation that allows us to work with local acquired

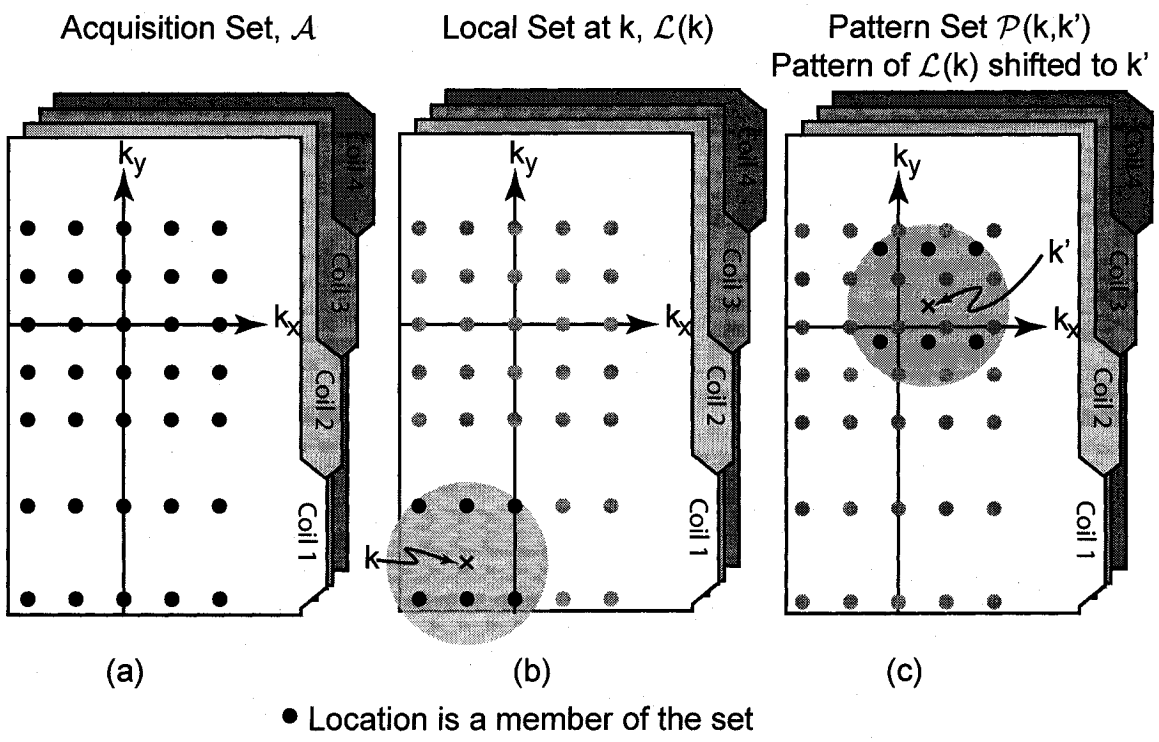


Figure 4.1: (a) The acquisition set, $\mathcal{A}$, contains all of the acquired data locations for all coils. (b) The local set, $\mathcal{L}(\mathbf{k})$, contains the acquired data locations for all coils in a radius κ around $\mathbf{k}$. (c) The pattern set, $\mathcal{P}(\mathbf{k}, \mathbf{k}')$, contains the locations in the pattern of $\mathcal{L}(\mathbf{k})$, but centered around $\mathbf{k}'$. The locations in $\mathcal{P}(\mathbf{k}, \mathbf{k}')$ are not necessarily in the acquisition set and therefore can be different from $\mathcal{L}(\mathbf{k}')$.

data. I then specify how the weights for the local neighborhood are calculated. Finally, I show that when the weights are calculated in this way, the errors in the estimated encoding vectors weighted by the magnetization are minimized.

Instead of using the entire acquisition set, $\mathcal{A}$, I define the local set, $\mathcal{L}(\mathbf{k})$, as a set of acquired locations local to k-space location $\mathbf{k}$:

$$\mathcal{L}(\mathbf{k}) = \{(j, \mathbf{k} + \mathbf{\Delta k}) | (j, \mathbf{k} + \mathbf{\Delta k}) \in \mathcal{A}, \|\mathbf{\Delta k}\| < \kappa\} . \tag{4.14}$$

Essentially, $\mathcal{L}(\mathbf{k})$, as described in Table 4.3 and illustrated in Fig. 4.1, contains all of the acquired locations from all coils within a radius κ of $\mathbf{k}$. By replacing the

Table 4.3: Local Projection Calibration Variables

$\mathcal{L}(\mathbf{k})$	Local set. Contains all the acquired locations, from all coils, within radius κ of $\mathbf{k}$.
$N_{\mathcal{L}(\mathbf{k})}$	Number of elements in $\mathcal{L}(\mathbf{k})$.
$E_{\mathcal{L}(\mathbf{k})}$	Encoding matrix for data at locations in $\mathcal{L}(\mathbf{k})$. $N_{\mathcal{L}(\mathbf{k})} \times N_v$
$\mathbf{d}_{\mathcal{L}(\mathbf{k})}$	Vector containing acquired data values local to $\mathbf{k}$. $N_{\mathcal{L}(\mathbf{k})} \times 1$
$\mathbf{w}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$	Vector containing all weights to synthesize $\widehat{d}_j(\mathbf{k})$ from data values at locations in $\mathcal{L}(\mathbf{k})$. $N_{\mathcal{L}(\mathbf{k})} \times 1$
$\widehat{\mathbf{e}}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$	Estimated encoding vector for location $(j, \mathbf{k})$. $N_v \times 1$ Linear combination of encoding vectors at locations in $\mathcal{L}(\mathbf{k})$.
$\boldsymbol{\epsilon}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$	Error in estimated encoding vector $\widehat{\mathbf{e}}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$. $N_v \times 1$
$\mathcal{P}(\mathbf{k}, \mathbf{k}')$	Pattern set. $\mathcal{L}(\mathbf{k})$ shifted by $\mathbf{k}' - \mathbf{k}$ in k-space.
$\mathbf{d}_{\mathcal{P}(\mathbf{k},\mathbf{k}')}$	Vector containing acquired data values at locations in $\mathcal{P}(\mathbf{k}, \mathbf{k}')$. $N_{\mathcal{L}(\mathbf{k})} \times 1$ When locations in $\mathcal{P}(\mathbf{k}, \mathbf{k}')$ are not part of the acquired data, $\widetilde{\mathbf{d}}_{\mathcal{P}}(\mathbf{k}, \mathbf{k}')$ is used as described in the Central Region Interpolation section.
$E_{\mathcal{P}(\mathbf{k},\mathbf{k}')}$	Encoding matrix for data at locations in $\mathcal{P}(\mathbf{k}, \mathbf{k}')$. $N_{\mathcal{L}(\mathbf{k})} \times N_v$
$\widehat{\mathbf{e}}_{j,\mathcal{P}(\mathbf{k},\mathbf{k}')}(\mathbf{k}')$	Estimated encoding vector for location $(j, \mathbf{k}')$. $N_v \times 1$ Linear combination of encoding vectors at locations in $\mathcal{P}(\mathbf{k}, \mathbf{k}')$.
$\boldsymbol{\epsilon}_{j,\mathcal{P}(\mathbf{k},\mathbf{k}')}(\mathbf{k}')$	Error in estimated encoding vector $\widehat{\mathbf{e}}_{j,\mathcal{P}(\mathbf{k},\mathbf{k}')}(\mathbf{k}')$. $N_v \times 1$

acquisition set, $\mathcal{A}$, with the local set, $\mathcal{L}(\mathbf{k})$, in Eq. 4.10, the synthesis of a datum from locally acquired data can be written as

$$\widehat{d}_j(\mathbf{k}) = \mathbf{w}^T_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})\mathbf{d}_{\mathcal{L}(\mathbf{k})} \tag{4.15}$$

and the associated estimated encoding vector can be expressed by replacing the acquisition set, $\mathcal{A}$, with the local set, $\mathcal{L}(\mathbf{k})$, in Eq. 4.12:

$$\widehat{\mathbf{e}}^T_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k}) = \mathbf{w}^T_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})E_{\mathcal{L}(\mathbf{k})}. \tag{4.16}$$

Each row of $E_{\mathcal{L}(\mathbf{k})}$ contains an encoding vector at one of the locations in $\mathcal{L}(\mathbf{k})$. Note that $E_{\mathcal{L}(\mathbf{k})}$ has far fewer rows than $E_{\mathcal{A}}$ and the encoding vectors comprising $E_{\mathcal{L}(\mathbf{k})}$ are not expected to span the space of the magnetization vector $\mathbf{m}$. As such, $\widehat{\mathbf{e}}^T_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$ will not be a perfect estimate of $\mathbf{e}^T_j(\mathbf{k})$. Rather, it is desirable to find the linear combination weights, $\mathbf{w}^T_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$, that minimize the magnitude of the error in the estimated encoding function:

$$\min_{\mathbf{w}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})} \left\|\boldsymbol{\epsilon}^T_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})\right\| = \min_{\mathbf{w}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})} \left\|\mathbf{w}^T_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})E_{\mathcal{L}(\mathbf{k})} - \mathbf{e}^T_j(\mathbf{k})\right\|. \tag{4.17}$$

Both local projection calibration and local projection interpolation are named using 'local projection' because the estimated encoding vector $\widehat{\mathbf{e}}^T_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$ is always in the space spanned by the encoding vectors in the local set and can be visualized as a projection of the true encoding vector onto this 'local subspace'. Local projection calibration is a method for generating the linear combination weights that would make $\widehat{\mathbf{e}}^T_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$, as calculated in Eq. 4.16, a projection of $\mathbf{e}^T_j(\mathbf{k})$ onto the local subspace. While Eq. 4.17 would generate such a set of linear combination weights, for it to be possible to use Eq. 4.17 directly, the encoding vectors would need to be known, implying that the coil sensitivities would also need to be known. Local projection calibration does not use Eq. 4.17 directly. Instead, it provides a way to find near optimal local weights without needing to know the encoding vectors.

Local projection calibration takes advantage of the fact that linear combination weights which are found in one region of k-space can be applied in a completely

different region. The reason such an approach works is because the element-by-element magnitude of the error in the estimated encoding vector, $\left|\epsilon_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})\right|$, which I refer to as the *magnitude image* of the error in the estimated encoding vector, does not change when the encoding vector location and its neighborhood is shifted in k-space from position $\mathbf{k}$ to position $\mathbf{k}'$. This can be shown by noting that shifting an encoding vector in k-space is equivalent to multiplying the vector by a gradient encoding term, allowing us to write $\mathbf{e}_j(\mathbf{k}') = \mathbf{e}_j(\mathbf{k}) \cdot \mathbf{g}(\mathbf{k}' - \mathbf{k})$. Since $\epsilon_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$ is a linear combination of encoding vectors, the shifted error vector can be written as $\epsilon_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k}) \cdot \mathbf{g}(\mathbf{k}' - \mathbf{k})$. Finally, since $g^*(\mathbf{k}, \mathbf{r})g(\mathbf{k}, \mathbf{r}) = \exp(i2\pi\mathbf{k}^T\mathbf{r})\exp(-i2\pi\mathbf{k}^T\mathbf{r}) = 1$

$$\left|\epsilon_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k}) \cdot \mathbf{g}(\mathbf{k}' - \mathbf{k})\right| = \left|\epsilon_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})\right|. \tag{4.18}$$

Alternatively, one can recognize that a shift in k-space is equivalent to applying a linear phase shift to the image-space vector $\epsilon_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$, which does not affect the magnitude of any element.

Because shifting in k-space does not affect the magnitude of an image-space vector, the same weights that minimize the magnitude of the shifted error vector will minimize the magnitude of the original error vector. Thus, what matters in determining the weights is not where the encoding vectors are in k-space, but the pattern they form relative to the k-space location of the encoding vector being synthesized. To take advantage of the ability to freely shift in k-space, the pattern set is defined as

$$\mathcal{P}(\mathbf{k}, \mathbf{k}') = \{(j, \mathbf{k}' + \Delta\mathbf{k}) | (j, \mathbf{k} + \Delta\mathbf{k}) \in \mathcal{L}(\mathbf{k})\}, \tag{4.19}$$

which takes the pattern of acquired locations around location $\mathbf{k}$ and centers them around location $\mathbf{k}'$. Illustrations of $\mathcal{A}$, $\mathcal{L}(\mathbf{k})$ and $\mathcal{P}(\mathbf{k}, \mathbf{k}')$ are given in Fig. 4.1. Using the pattern set, a modification of Eq. 4.16 can be written, in which all of the encoding vectors have been shifted in k-space from position $\mathbf{k}$ to position $\mathbf{k}'$:

$$\widehat{\mathbf{e}}^T_{j,\mathcal{P}(\mathbf{k},\mathbf{k}')}(\mathbf{k}') = \mathbf{w}^T_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k}) E_{\mathcal{P}(\mathbf{k},\mathbf{k}')}. \tag{4.20}$$

Equation 4.20 expresses the estimate of the encoding function at location $(j, \mathbf{k}')$ from

encoding functions in the local neighborhood of $\mathbf{k}'$ that are of the same pattern as the acquired encoding functions are around $\mathbf{k}$. The error in the estimated encoding vector in Eq. 4.20 can be written as

$$\begin{aligned} \boldsymbol{\epsilon}_{j,\mathcal{P}(\mathbf{k},\mathbf{k}')}(\mathbf{k}') &= \widehat{\mathbf{e}}_{j,\mathcal{P}(\mathbf{k},\mathbf{k}')}(\mathbf{k}') - \mathbf{e}_j(\mathbf{k}') && (4.21)\\ &= \left[\widehat{\mathbf{e}}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k}) - \mathbf{e}_j(\mathbf{k})\right] \cdot \mathbf{g}(\mathbf{k}' - \mathbf{k}) && (4.22)\\ &= \boldsymbol{\epsilon}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k}) \cdot \mathbf{g}(\mathbf{k}' - \mathbf{k}). && (4.23) \end{aligned}$$

Thus, using the pattern set, Eq. 4.18 can be written as

$$\left|\boldsymbol{\epsilon}_{j,\mathcal{P}(\mathbf{k},\mathbf{k}')}(\mathbf{k}')\right| = \left|\boldsymbol{\epsilon}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})\right|. \tag{4.24}$$

Note that both Eq. 4.16 and Eq. 4.20 use the same set of weights, $\mathbf{w}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$, to calculate the estimated encoding vector. When the same weights are used, Eq. 4.24 shows that the magnitude images of the error vectors for both encoding vectors are the same.

Without knowing the coil sensitivities, no encoding vectors are known, so one cannot find the linear combination weights using Eq. 4.17. However, there is sufficiently sampled data within the low spatial frequency central region of k-space and this data can be used to generate the linear combination weights.

All of the data used to find the linear combination weights comes from the so-called *central region* of k-space. That is, local projection calibration assumes that there is a region, called the central region, where $d_j(\mathbf{k})$ is known, or can be found, for all encoding locations $(j, \mathbf{k})$ as long as $\mathbf{k}$ is within the central region. In addition to the central region, we define the *fit region* as a region within the central region as shown in Fig 4.2. The fit region is important because for any location $\mathbf{k}'$ within the fit region, the pattern set $\mathcal{P}(\mathbf{k}, \mathbf{k}')$ will always be in the central region.

Local projection calibration determines the local weights by minimizing

$$\left\| \mathbf{w}^T_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k}) \left[\mathbf{d}_{\mathcal{P}(\mathbf{k},\mathbf{k}'_1)} \ldots \mathbf{d}_{\mathcal{P}(\mathbf{k},\mathbf{k}'_{N_{\mathbf{k}'}})}\right] - \left[d_j(\mathbf{k}'_1) \ldots d_j(\mathbf{k}'_{N_{\mathbf{k}'}})\right] \right\|, \tag{4.25}$$

where $\mathbf{k}'_1 \ldots \mathbf{k}'_{N_{\mathbf{k}'}}$ denote $N_{\mathbf{k}'}$ k-space locations that fall on a full-FOV Cartesian grid

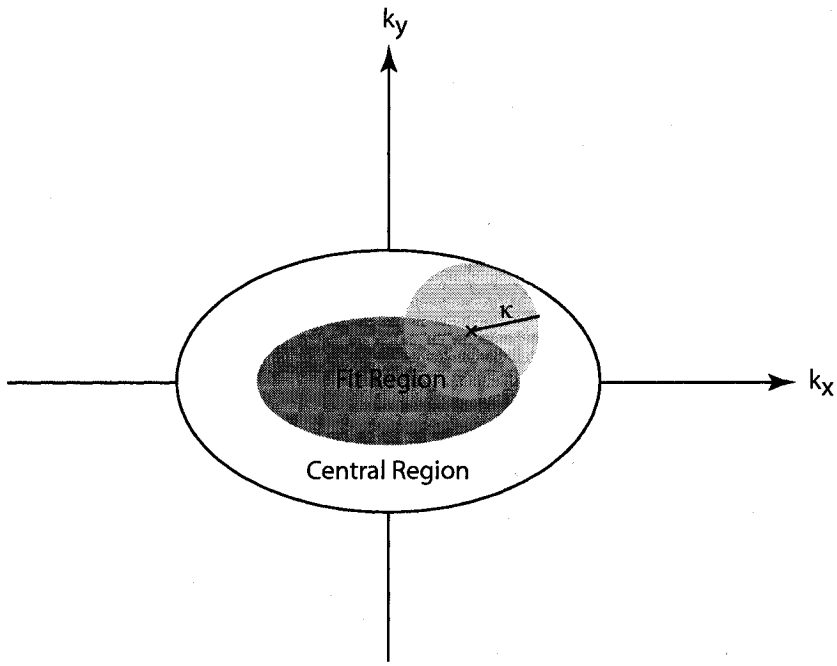


Figure 4.2: The central region is the region within which all possible $d_j(\mathbf{k})$ are known. The fit region is the region inside of the central region, with a margin of κ, the radius of the pattern set.

in the fit region and $\mathbf{d}_{\mathcal{P}(\mathbf{k},\mathbf{k}')}$ contains data at the locations in $\mathcal{P}(\mathbf{k},\mathbf{k}')$. I will develop more insight into the choice of $\mathbf{k}'_1 \ldots \mathbf{k}'_{N_{\mathbf{k}'}}$ shortly, as we analyze Eq. 4.25. In addition, it is important to remember that in this section it is assumed that within the central region of k-space, the data values are known or can be found for every encoding location $(j,\mathbf{k})$. When central region interpolation is discussed, it will become clear how the data values in $\mathbf{d}_{\mathcal{P}(\mathbf{k},\mathbf{k}'_i)}$ and $d_j(\mathbf{k}'_i)$ are determined when they are not explicitly acquired. The weights can be computed directly as

$$\mathbf{w}^T_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k}) = \left[d_j(\mathbf{k}'_1) \ldots d_j(\mathbf{k}'_{N_{\mathbf{k}'}})\right] \left[\mathbf{d}_{\mathcal{P}(\mathbf{k},\mathbf{k}'_1)} \ldots \mathbf{d}_{\mathcal{P}(\mathbf{k},\mathbf{k}'_{N_{\mathbf{k}'}})}\right]^\dagger, \tag{4.26}$$

where $\dagger$ denotes a pseudo-inverse.

While Eq. 4.26 shows how the local projection calibration technique finds linear combination weights using data from the central region, it does not give much information on the properties of these weights. I now take advantage of the notation I have developed to demonstrate that the weights found by minimizing Eq. 4.25 will minimize the error in the estimated encoding vector weighted by the magnetization. That the minimization is performed on the magnetization-weighted error in the estimated encoding vector and not on the unweighted error in the estimated encoding vector is to some degree unavoidable. However, this weighting is not undesirable, as the estimated encoding vector will tend to be more accurate where there are large amounts of magnetization. Errors in the estimated encoding vectors at locations where there are large amounts of magnetization can result in large errors in the synthesized data values, so it is especially important that the estimated encoding vectors be accurate at these locations.

To demonstrate that such a minimization is, in fact, accomplished by local projection calibration, I start by looking at the data estimation error for a k-space location $\mathbf{k}'$, within the fit region and expanding each data value in terms of its encoding vector

and the magnetization, $d_j(\mathbf{k}) = \mathbf{e}_j^T(\mathbf{k})\mathbf{m}$ and $\mathbf{d}_{\mathcal{P}(\mathbf{k},\mathbf{k}')} = E_{\mathcal{P}(\mathbf{k},\mathbf{k}')}\mathbf{m}$.

$$\begin{aligned}
\mathbf{w}_{j,\mathcal{L}(\mathbf{k})}^T(\mathbf{k})\mathbf{d}_{\mathcal{P}(\mathbf{k},\mathbf{k}')} - d_j(\mathbf{k}') &= \mathbf{w}_{j,\mathcal{L}(\mathbf{k})}^T(\mathbf{k})E_{\mathcal{P}(\mathbf{k},\mathbf{k}')}\mathbf{m} - \mathbf{e}_j^T(\mathbf{k}')\mathbf{m} && (4.27)\\
&= \widehat{\mathbf{e}}_{j,\mathcal{P}(\mathbf{k},\mathbf{k}')}^T(\mathbf{k}')\mathbf{m} - \mathbf{e}_j^T(\mathbf{k}')\mathbf{m} && (4.28)\\
&= \boldsymbol{\epsilon}_{j,\mathcal{P}(\mathbf{k},\mathbf{k}')}^T(\mathbf{k}')\mathbf{m} && (4.29)\\
&= \mathbf{g}^T(\mathbf{k}')\left[\boldsymbol{\epsilon}_{j,\mathcal{P}(\mathbf{k},\mathbf{0})}(\mathbf{0})\cdot\mathbf{m}\right]. && (4.30)
\end{aligned}$$

The expression $\mathbf{g}^T(\mathbf{k}')\left[\boldsymbol{\epsilon}_{j,\mathcal{P}(\mathbf{k},\mathbf{0})}(\mathbf{0})\cdot\mathbf{m}\right]$ can be interpreted in a very natural way. First of all, note that $\boldsymbol{\epsilon}_{j,\mathcal{P}(\mathbf{k},\mathbf{0})}(\mathbf{0})$ is a special case; since $\mathbf{e}_j(\mathbf{0}) = \mathbf{s}_j$, the sensitivity of coil j, $\widehat{\mathbf{e}}_{j,\mathcal{P}(\mathbf{k},\mathbf{0})}(\mathbf{0})$ is an estimate of the sensitivity of coil j and $\boldsymbol{\epsilon}_{j,\mathcal{P}(\mathbf{k},\mathbf{0})}(\mathbf{0})$ is the error in that estimate. $\boldsymbol{\epsilon}_{j,\mathcal{P}(\mathbf{k},\mathbf{0})}(\mathbf{0})\cdot\mathbf{m}$ is then the sensitivity estimation error weighted by the magnetization. Finally, multiplication by $\mathbf{g}^T(\mathbf{k}')$ computes the discrete space Fourier transform (DSFT) of the weighted sensitivity estimation error, evaluated at $\mathbf{k}'$.

In Eq. 4.25, the magnitude is taken of a vector, each element of which contains the data estimation error, $\mathbf{w}_{j,\mathcal{L}(\mathbf{k})}^T(\mathbf{k})\ \mathbf{d}_{\mathcal{P}(\mathbf{k},\mathbf{k}')} - d_j(\mathbf{k}')$, for one of the $\mathbf{k}'$ locations in $\mathbf{k}'_1 \ldots \mathbf{k}'_{N_{\mathbf{k}'}}$. Expressing the data estimation error for each k-space location, $\mathbf{k}'_1 \ldots \mathbf{k}'_{N_{\mathbf{k}'}}$, as a sample of the discrete space Fourier transform of the sensitivity estimation error weighted by the magnetization, as in Eq. 4.30, Eq. 4.25 can be written as the magnitude of the discrete space Fourier transform of the magnetization-weighted sensitivity estimation error, sampled at the low spatial frequency locations $\mathbf{k}'_1 \ldots \mathbf{k}'_{N_{\mathbf{k}'}}$:

$$\left\| \mathrm{DSFT}\left\{\boldsymbol{\epsilon}_{j,\mathcal{P}(\mathbf{k},\mathbf{0})}(\mathbf{0})\cdot\mathbf{m}\right\}(\mathbf{k}) \sum_{i=1}^{N_{\mathbf{k}'}} \delta(\mathbf{k}'_i - \mathbf{k}) \right\|. \qquad (4.31)$$

Since Eq. 4.25 is mathematically equivalent to Eq. 4.31, the weights that minimize Eq. 4.25 will also be the weights that minimize Eq. 4.31. Thus, the weights obtained using local projection calibration minimize the low frequency components of the magnetization-weighted sensitivity estimation error. Since the majority of the energy of the magnetization and sensitivity functions is contained in their low frequency components, local projection calibration will tend to find weights that successfully

minimize the overall magnetization-weighted sensitivity estimation error.

By setting $\mathbf{k}' = \mathbf{0}$ in Eq. 4.24, it becomes apparent that the magnitude image of the sensitivity estimation error is equivalent to the magnitude image of the error in the estimated encoding vector used to synthesize the datum at location $\mathbf{k}$:

$$\left|\epsilon_{j,\mathcal{P}(\mathbf{k},\mathbf{0})}(\mathbf{0})\right| = \left|\epsilon_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})\right| \tag{4.32}$$

Thus, the magnetization-weighted sensitivity estimation error is equal to the magnetization-weighted error in the estimated encoding vector,

$$\left\|\epsilon_{j,\mathcal{P}(\mathbf{k},\mathbf{0})}(\mathbf{0}) \cdot \mathbf{m}\right\| = \left\|\epsilon_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k}) \cdot \mathbf{m}\right\| \tag{4.33}$$

and minimizing the magnetization-weighted sensitivity estimation error is equivalent to minimizing the magnetization-weighted error in the estimated encoding vector.

Equation 4.31 also provides some insight into the choice of $\mathbf{k}'_1 \ldots \mathbf{k}'_{N_{\mathbf{k}'}}$. By choosing $\mathbf{k}'_1 \ldots \mathbf{k}'_{N_{\mathbf{k}'}}$ to cover a full-FOV grid in the fit region, one ensures that all of the low frequency components of the magnetization-weighted sensitivity estimation error are included in the minimization. Choosing the extent of the fit region in k-space is still an inexact art, depending to some degree on the energy distribution in k-space of the magnetization and coil sensitivities. However, by choosing $N_{\mathbf{k}'}$ to be much larger than the total number of elements in the local set, $\mathcal{L}(\mathbf{k})$, the error in the estimated encoding vector will be determined more by the ability of the local encoding vectors, $E_{\mathcal{L}(\mathbf{k})}$, to synthesize the estimated encoding vector than by the effect the extent of the fit region has on the calculation of the weight values.

In the above analysis of local projection calibration, it was shown that high quality local weights can be found without knowing the coil sensitivities, assuming all data values required by Eq. 4.26 are known, or can be found. While interpolation might be necessary when required data values have not been acquired directly, when all of the acquired data falls on a full-FOV uniform Cartesian grid, acquiring data at all grid locations in the central region is sufficient for local projection calibration. As shown in Fig. 4.3(a), this is because all of the data values required by Eq. 4.26 are contained in the acquisition set $\mathcal{A}$. The autocalibration technique used by Cartesian GRAPPA is

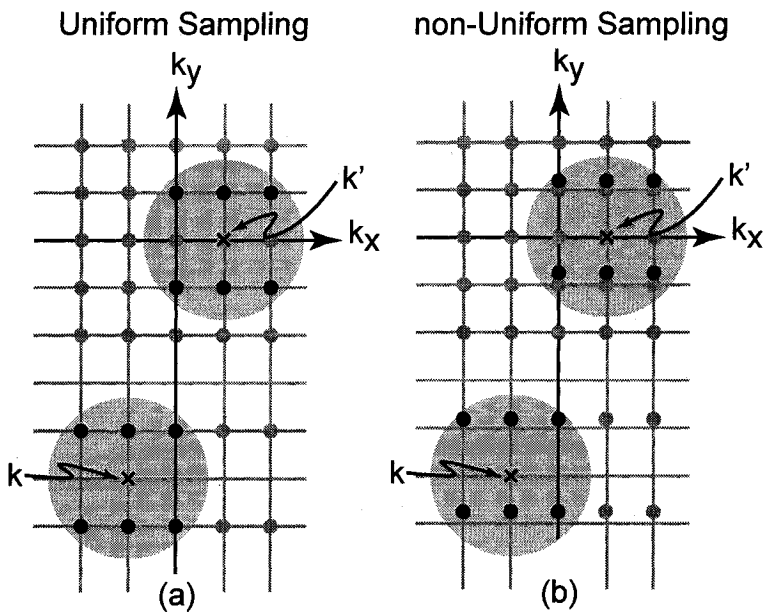


Figure 4.3: (a) When data is acquired on a uniform Cartesian grid, it is possible to choose the $\mathbf{k}'$ such that the pattern set $\mathcal{P}(\mathbf{k}, \mathbf{k}')$ is contained in the acquisition set $\mathcal{A}$. In this case, no interpolation within the central region is required to perform local projection calibration. (b) When a non-uniform sampling trajectory is used, $\mathcal{P}(\mathbf{k}, \mathbf{k}')$ is not contained in the acquisition set $\mathcal{A}$ and data at the locations in $\mathcal{P}(\mathbf{k}, \mathbf{k}')$ must be synthesized.

a practical implementation of local projection calibration that takes advantage of this case. However, to use local projection calibration for arbitrary k-space trajectories, one must find a practical way for dealing with the reality that data is not acquired for every location in the central region. In the next section, I show how APPEAR uses central region interpolation to deal with this problem.

Central Region Interpolation

When data is acquired using a k-space sampling pattern that does not fall on a uniform Cartesian grid, local projection calibration cannot be used directly since it requires data values from locations in the central region that have not been acquired, as shown in Fig. 4.3(b). To overcome this problem, the APPEAR method accurately synthesizes data values in the central region from the acquired data. By being able to synthesize any value in the central region, APPEAR is able to take full advantage of the local projection calibration technique.

Note that APPEAR uses two types of data synthesis. Local projection interpolation is used to synthesize unacquired data from data acquired on *multiple coils*, and the linear combination weights for this are found using local projection calibration. Central region interpolation uses central region data from a *single coil* to synthesize central region data on that same coil. When data from multiple coils is used for synthesis, the challenge is to perform accurate calibration and generate good linear combination weights without knowing the coil sensitivities (which prevents one from knowing the encoding functions). In the above section on local projection calibration, I describe how APPEAR deals with this challenge. When an unacquired datum is synthesized only from data acquired on the same coil, as is the case for central region interpolation, only the gradient encoding is relevant and the gradient encoding vectors are known. Thus, calibration is not a problem in this case. Rather, the challenge is to ensure that the synthesis is accurate. Here, I show how central region interpolation achieves this goal.

Before engaging the details of central region interpolation, I summarize the overall APPEAR procedure. As illustrated in Fig. 4.4, for each point on the k-space grid to be synthesized (from acquired data on all coils), APPEAR determines the local pattern of acquired data and then synthesizes data (from acquired data on a single coil) in that pattern at locations across a full-FOV grid in the fit region. In the central region, a synthesized datum is denoted by $\widetilde{d}_j(\mathbf{k})$, differentiating it from the synthesized datum value $\widehat{d}_j(\mathbf{k})$. Whereas $\widehat{d}_j(\mathbf{k})$ is synthesized from data from all coils in regions where the gradient encoding does not sufficiently encode for the FOV, $\widetilde{d}_j(\mathbf{k})$ is synthesized only from coil j data in the central region, where the gradient encoding

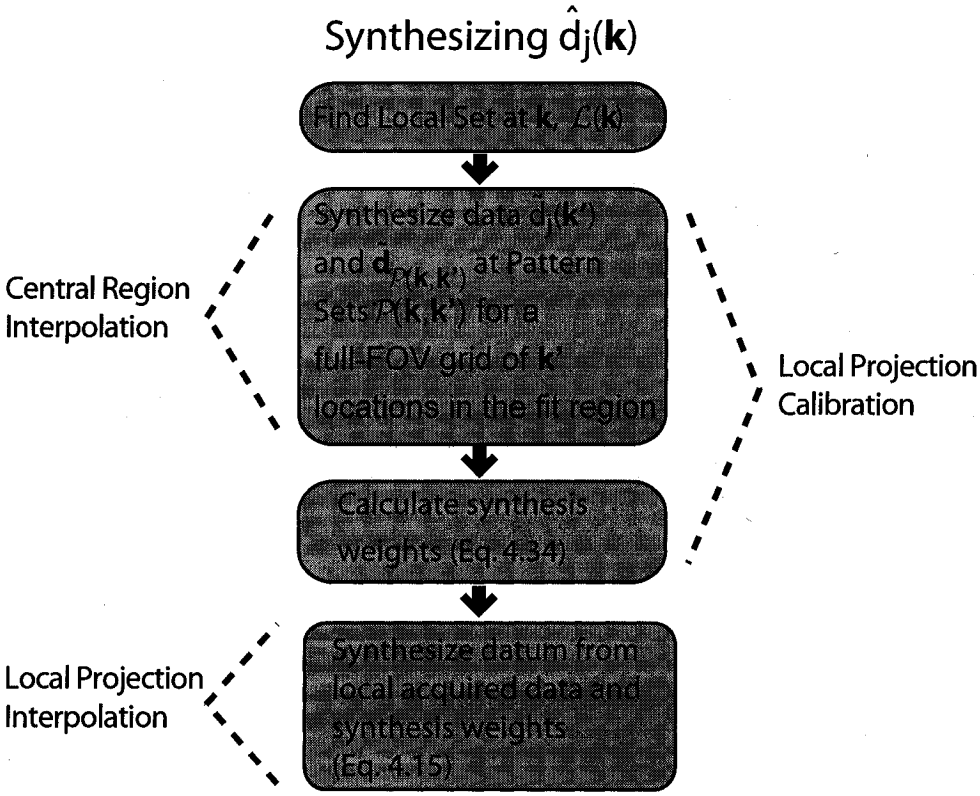


Figure 4.4: The APPEAR procedure for synthesizing an unacquired datum at location $\mathbf{k}$ using local acquired data from multiple coils.

does sufficiently encode for the FOV. The data values synthesized at the locations in $\mathcal{P}(\mathbf{k}, \mathbf{k}')$ can be assembled into a vector, denoted by $\widetilde{\mathbf{d}}_{\mathcal{P}(\mathbf{k},\mathbf{k}')}$. The weights can then be calculated as

$$\mathbf{w}_{j,\mathcal{L}(\mathbf{k})}^{T}(\mathbf{k}) = \left[\widetilde{d}_j(\mathbf{k}'_1) \ldots \widetilde{d}_j(\mathbf{k}'_N)\right] \left[\widetilde{\mathbf{d}}_{\mathcal{P}(\mathbf{k},\mathbf{k}'_1)} \ldots \widetilde{\mathbf{d}}_{\mathcal{P}(\mathbf{k},\mathbf{k}'_N)}\right]^{\dagger}, \tag{4.34}$$

which is identical to Eq. 4.26, except that the data values are synthesized. Using Eq. 4.34, APPEAR determines a separate set of weights for each grid point to be synthesized, repeating the procedure illustrated in Fig. 4.4 to synthesize data at all of the grid points outside of the central region.

The success of the APPEAR method rests heavily on the ability to synthesize data values within the central region of a coil's k-space using only the data acquired on that receiver coil. Techniques, such as gridding [1, 2] and BURS [8], for synthesizing data in new k-space locations from a set of full-FOV data are commonly used for reconstructing non-Cartesian k-space sampling patterns acquired using only one coil. However, these methods are designed to minimize errors in a particular FOV in image space, whereas the synthesis method used by APPEAR needs to minimize errors in the low spatial frequency region of k-space. This is a subtle but important difference. For example, when gridding is performed on an oversampled grid, errors can be pushed outside of the FOV of the object, separating the errors from the object. However, these errors cannot be separated from the data in such a clean fashion in k-space. I now develop the procedure that APPEAR uses to get high quality synthesized data suitable for use with the local projection calibration technique.

The single-coil data synthesis problem can be formulated in a similar way to the multi-coil formulation. However, in the single-coil case, the unknown coil sensitivity is grouped with the magnetization by letting $\mathbf{m}_s$ denote the sensitivity-weighted magnetization, described in Table 4.4. Since only one coil is being considered at a time, the coil index j has been dropped. The analysis of the sensitivity-weighted magnetization is done by gradient encoding, such that $d(\mathbf{k}) = \mathbf{g}^T(\mathbf{k})\mathbf{m}_s$. Similar to the multi-coil case, one can synthesize an encoding function from acquired encoding functions in the local k-space neighborhood, but in this case the encoding is solely

Table 4.4: Central Region Interpolation Variables

$\widetilde{d}_j(\mathbf{k})$	Synthesized data value at location $(j, \mathbf{k})$ from coil j acquired data.
$\mathbf{m}_s$	Sensitivity weighted image. $N_v \times 1$
$\mathcal{C}(\mathbf{k})$	Kernel set. Contains all the acquired k-space locations within radius κ of $\mathbf{k}$.
$N_{\mathcal{C}(\mathbf{k})}$	Number of elements in $\mathcal{C}(\mathbf{k})$.
$G_{\mathcal{C}(\mathbf{k})}$	Gradient encoding matrix for data at locations in $\mathcal{C}(\mathbf{k})$. $N_{\mathcal{C}(\mathbf{k})} \times N_v$
$\mathbf{w}_{\mathcal{C}(\mathbf{k})}(\mathbf{k})$	Vector containing all weights to synthesize $\mathbf{g}(\mathbf{k})$ from gradient encoding vectors at locations in $\mathcal{C}(\mathbf{k})$. $N_{\mathcal{C}(\mathbf{k})} \times 1$
$\widehat{\mathbf{g}}_{\mathcal{C}(\mathbf{k})}(\mathbf{k})$	Estimated gradient encoding vector for location $\mathbf{k}$. $N_v \times 1$

accomplished by the gradients. The single coil version of Eq. 4.16 can be written as

$$\widehat{\mathbf{g}}^T_{\mathcal{C}(\mathbf{k})}(\mathbf{k}) = \mathbf{w}^T_{j,\mathcal{C}(\mathbf{k})}(\mathbf{k}) G_{\mathcal{C}(\mathbf{k})}, \tag{4.35}$$

where $\mathcal{C}(\mathbf{k})$ is the k-space kernel set

$$\mathcal{C}(\mathbf{k}) = \{\mathbf{k} + \mathbf{\Delta k} | d_j(\mathbf{k} + \mathbf{\Delta k}) \text{ was acquired}, \|\mathbf{\Delta k}\| < \kappa\} \tag{4.36}$$

and $G_{\mathcal{C}(\mathbf{k})}$ is the gradient encoding matrix, each row of which contains a gradient encoding vector whose location is in the kernel set. The estimated gradient encoding vector is denoted by $\widehat{\mathbf{g}}_{\mathcal{C}(\mathbf{k})}(\mathbf{k})$ and the linear combination weights, $\mathbf{w}_{j,\mathcal{C}(\mathbf{k})}(\mathbf{k})$, can be found as the weights that minimize the magnitude of the error in the estimated gradient encoding vector:

$$\min_{\mathbf{w}_{j,\mathcal{C}(\mathbf{k})}(\mathbf{k})} \left\| \widehat{\mathbf{g}}^T_{\mathcal{C}(\mathbf{k})}(\mathbf{k}) - \mathbf{g}^T(\mathbf{k}) \right\| = \min_{\mathbf{w}_{j,\mathcal{C}(\mathbf{k})}(\mathbf{k})} \left\| \mathbf{w}^T_{j,\mathcal{C}(\mathbf{k})}(\mathbf{k}) G_{\mathcal{C}(\mathbf{k})} - \mathbf{g}^T(\mathbf{k}) \right\|. \tag{4.37}$$

Unlike the multi-coil case where the encoding vectors are not known, in the single coil case the gradient encoding vectors are known and the weights can be found directly by minimizing Eq. 4.37.

While such a procedure finds the optimal weights for estimating $\mathbf{g}(\mathbf{k})$ from its surrounding neighborhood, this estimate is still not of high enough accuracy when using a small number of points. To illustrate this, I use a one dimensional example, showing the accuracy that can be expected from this approach. Five different attempts

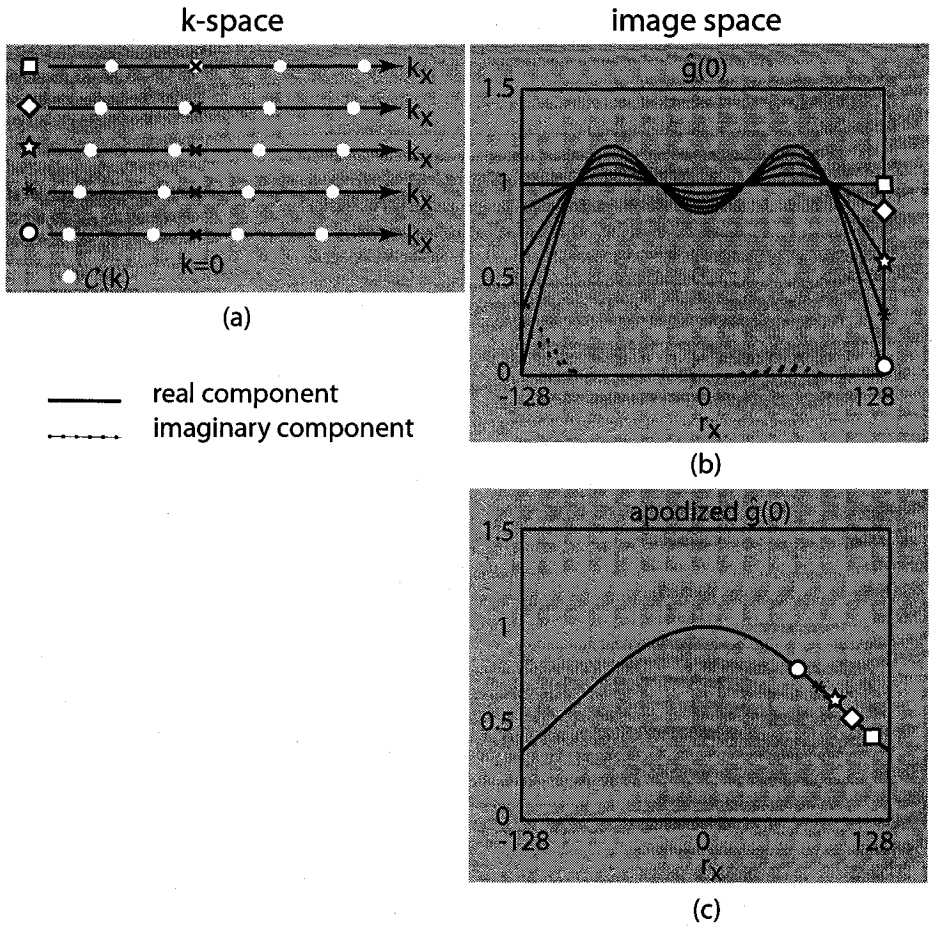


Figure 4.5: Oversampling and apodization can greatly increase the accuracy of synthesized data values in the central region. In this 1D illustration, gradient encoding vectors at four acquired locations are used to synthesize the gradient encoding vector at $k_x = 0$. (a) The synthesis is performed for different shifted versions of the acquired vector. (b) Without apodization or oversampling, the accuracy of the synthesis is poor. (c) The acquired locations oversample the FOV with an oversampling factor of 1.5 and an apodized version of the gradient encoding vector at $k_x = 0$ is synthesized. In this case, all of the synthesized gradient encoding vectors overlap with the apodization vector target, showing that this synthesis can be done with high accuracy.

are made to synthesize $\mathbf{g}(\mathbf{0})$, which equals unity, from four local gradient encoding vectors. In each of the five attempts, the four local gradient encoding vectors are shifted slightly in relation to $\mathbf{g}(\mathbf{0})$, as shown in Fig. 4.5(a). For each attempt, I find the optimal weights by minimizing Eq. 4.37 and use these weights in Eq. 4.35 to find the estimated gradient encoding vector. These five estimated encoding vectors are plotted in Fig. 4.5(b). In the first attempt, where one of the local gradient encoding vectors is at location $\mathbf{k} = \mathbf{0}$, $\widehat{\mathbf{g}}(\mathbf{k})$ is unity throughout. However, as the local gradient encoding vectors get shifted, the synthesis degrades. In the last attempt, where $\mathbf{g}(\mathbf{0})$ is midway between two local gradient encoding vectors, the edges of the estimated encoding vector are close to zero.

The accuracy of the data synthesis can be greatly increased using oversampling and apodization. While both of these concepts are used in gridding, they must be applied in a slightly different way in this case. Instead of depositing the data on an oversampling grid, in this case oversampling is accomplished by acquiring data in the central region more closely together (spaced for a larger FOV than the FOV of the object). Since higher oversampling factors reduce the overall SNR efficiency of the scan [34], it is important to use the smallest oversampling factor that still allows for highly accurate data synthesis. Currently an oversampling ratio of 1.5 is used, which has been shown to give high accuracy in gridding [35]. APPEAR takes advantage of apodization by finding the weights that synthesize an apodized version of the gradient encoding vector instead of the gradient encoding vector itself. By synthesizing all of the data in the central region using similarly apodized versions of the gradient encoding vectors, the resultant data, $\widetilde{d}_j(\mathbf{k})$, is equivalent to data obtained by encoding an apodized version of the sensitivity-weighted magnetization with the non-apodized gradient encoding vectors.

Figure 4.5(c) shows that using oversampling and apodization lead to a dramatic improvement in the fidelity of the synthesized data. In Fig. 4.5(c), all five apodized estimated gradient encoding vectors and the target apodization function overlap, appearing as one curve.

In a case such as the example in Fig. 4.5, where the samples in the central region are uniformly spaced, finding the weights that fit to an apodized version of the

gradient encoding vector can be accomplished by convolving the acquired data with an appropriate gridding kernel and sampling at the unacquired location. Thus, the result shown in Fig. 4.5(c) can be realized by using a Kaiser-Bessel window with shape parameter $\beta = 7.9$, chosen according to Eq. 5 in [35] for an oversampling ratio of 1.5 and a kernel width of 4. In this case, the apodization will be equivalent to the Fourier transform of the gridding kernel. Figure 4.6 shows an image space picture illustrating how oversampling and apodization allow for highly accurate synthesis from uniformly spaced samples. As I discuss in the next chapter, when the samples in the central region are not uniformly spaced, the gridding approach cannot be expected to find the optimal weights. However it is possible that for some k-space trajectories such as radial and spiral, gridding with density compensation might give sufficient accuracy. While a gridding approach is computationally attractive, the weights can alternatively be found directly by solving for the weights that minimize the error in the estimated apodized gradient encoding function:

$$\min_{\mathbf{w}_{j,\mathcal{C}(\mathbf{k})}(\mathbf{k})} \left\| \mathbf{w}_{j,\mathcal{C}(\mathbf{k})}^T(\mathbf{k}) G_{\mathcal{C}(\mathbf{k})} - \mathbf{g}_{\mathbf{a}}^T(\mathbf{k}) \right\| \tag{4.38}$$

where $\mathbf{g}_{\mathbf{a}}(\mathbf{k}) = \mathbf{a} \cdot \mathbf{g}(\mathbf{k})$ for apodization vector $\mathbf{a}$. This approach is attractive because it will find the optimal weights for any given kernel set, $\mathcal{C}(\mathbf{k})$, as long as it is sufficiently oversampled for the FOV.

When fitting to an apodized gradient encoding function, the synthesized data in the central region is equivalent to data that would be obtained with the same coil sensitivity but with a magnetization that is an apodized version of the actual magnetization. Thus, instead of the local projection calibration technique minimizing the encoding estimate error weighted by the magnetization, in the APPEAR case it minimizes the encoding estimate error weighted by an apodized version of the magnetization. Because of this, one must be careful not to allow too severe of an apodization, which can reduce the quality of the estimated encoding vector toward the edges of the FOV. In all experiments to date, the apodization function used (Fourier transform of a Kaiser-Bessel window) performs very well.

By slightly oversampling in the central region and synthesizing data that would

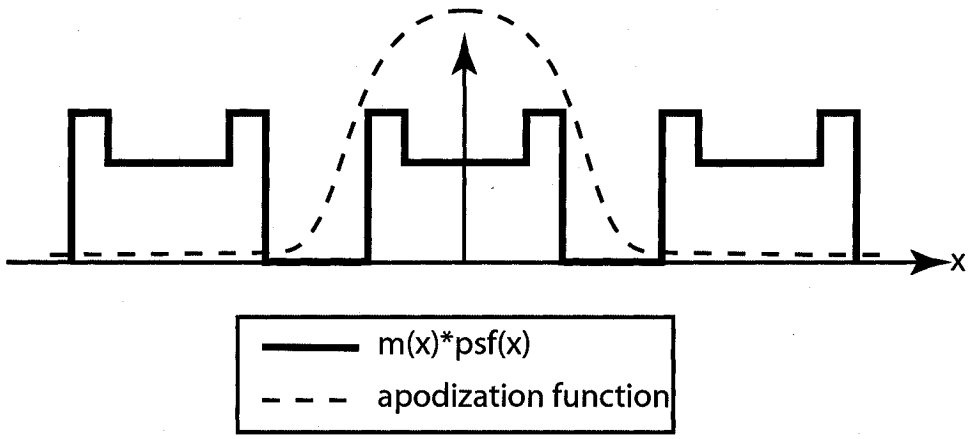


Figure 4.6: An image space illustration of how oversampling and apodization allow high quality synthesis of the apodized magnetization when the samples in the central region are uniformly spaced. Oversampling the central region pushes the image repetitions further away, allowing for a transition band. The apodization function (the Fourier transform of the gridding kernel) then suppresses the image repetitions. Unlike gridding, in k-space interpolation, $m(\mathbf{r}) * \mathrm{psf}(\mathbf{r})$ must be zero (or sufficiently close to zero) in the transition band.

be generated from an apodized version of the magnetization, instead of insisting on synthesis of data that would be acquired by the scanner, one is able to provide the local projection calibration technique with data values at any location in the central region, as required by the local projection calibration technique. In doing so, APPEAR gives a practical implementation of local projection calibration for arbitrary k-space trajectories.

4.2.3 Noise

Noise in the acquired data samples affects APPEAR in two ways. Firstly, noise in the central region can degrade the quality of the synthesis weights thereby increasing the errors in the estimated encoding vectors. Since errors in the estimated encoding vectors can manifest themselves as structured image artifacts, it is important that the data acquired in the central region have a sufficiently high signal-to-noise ratio to ensure that the estimated encoding vectors are accurate. Secondly, noise in the data samples leads to noise in the reconstructed image. Similar to other multi-coil reconstruction methods, the noise level varies from voxel to voxel. Since APPEAR is non-iterative, the noise level at each voxel can be calculated in a straightforward manner from the receiver noise matrix and the synthesis weights, as shown in the appendix D.

4.3 Methods

Numerical simulations were performed in Matlab (MathWorks, Natick, MA) using a Shepp-Logan phantom and the same coil architecture used for the numerical simulations in [36]. The k-space trajectory used, shown in Fig. 4.7(a), consisted of 128 phase-encode lines, the central 20 lines spaced with an oversampling ratio of 1.5. The remaining 108 lines were uniformly spaced, giving a constant acceleration of 2.25 in the high spatial frequency region. As a full-FOV dataset for equivalent k-space coverage would require 256 phase-encode lines, the net acceleration was 2. Reconstructions were performed using PARS and APPEAR with a local set radius, $\kappa = 4/\,\mathrm{FOV}$. For

APPEAR, interpolation in the central region was performed using a kernel set radius of 4/(3FOV). The apodization function used was the Fourier transform of a Kaiser-Bessel window with shape parameter, $\beta = 7.9$, illustrated in Fig. 4.5(c).

Scanner experiments were performed on a 1.5T GE Excite scanner using two k-space trajectories: the trajectory used in the numerical simulations and a non-uniformly spaced trajectory. The non-uniformly spaced trajectory was identical to the uniformly spaced trajectory except that the 108 high spatial frequency phase-encodes were non-uniformly spaced, giving an acceleration ranging from 1 to 3.5, as seen in Fig. 4.7(c). A standard GRASS sequence was used for all scans.

The first experiment consisted of scanning an axial slice of a ball phantom using an 8-channel high-resolution knee coil (MRI Devices Corp., Waukesha, Wisconsin). In this experiment, the frequency encoding was done in the A/P direction. The second experiment consisted of scanning an axial slice of the brain of a healthy volunteer using an 8-channel high-resolution head coil (MRI Devices Corp., Waukesha Wisconsin). In this experiment, the frequency encoding was done in the R/L direction. A 5 mm slice was excited for both experiments. The ball was scanned with a TE/TR of 10/1000 and an 11 cm FOV. The brain was scanned with a TE/TR of 10/50 and a 22 cm FOV. The data was reconstructed using PARS, APPEAR and by separately gridding the data from each coil followed by a sum-of-squares combination. The gridding was performed as described in [35], with a Kaiser-Bessel kernel, an oversampling factor of 1.25 and a kernel width of 4.

4.4 Results

The results for the numerical simulations are shown in Fig. 4.8. While the APPEAR method is able to remove the aliasing artifact, the PARS reconstruction still has visible aliasing artifact. Since the coil sensitivities are known in the case of the numerical phantom experiment, it is possible to calculate the error in the estimated encoding vector, $\boldsymbol{\epsilon}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$, directly for a set of linear combination weights. The error vector, $\boldsymbol{\epsilon}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$, can be viewed as an image, similar to the magnetization vector $\mathbf{m}$. While the error vector images do not directly represent errors in the reconstructed

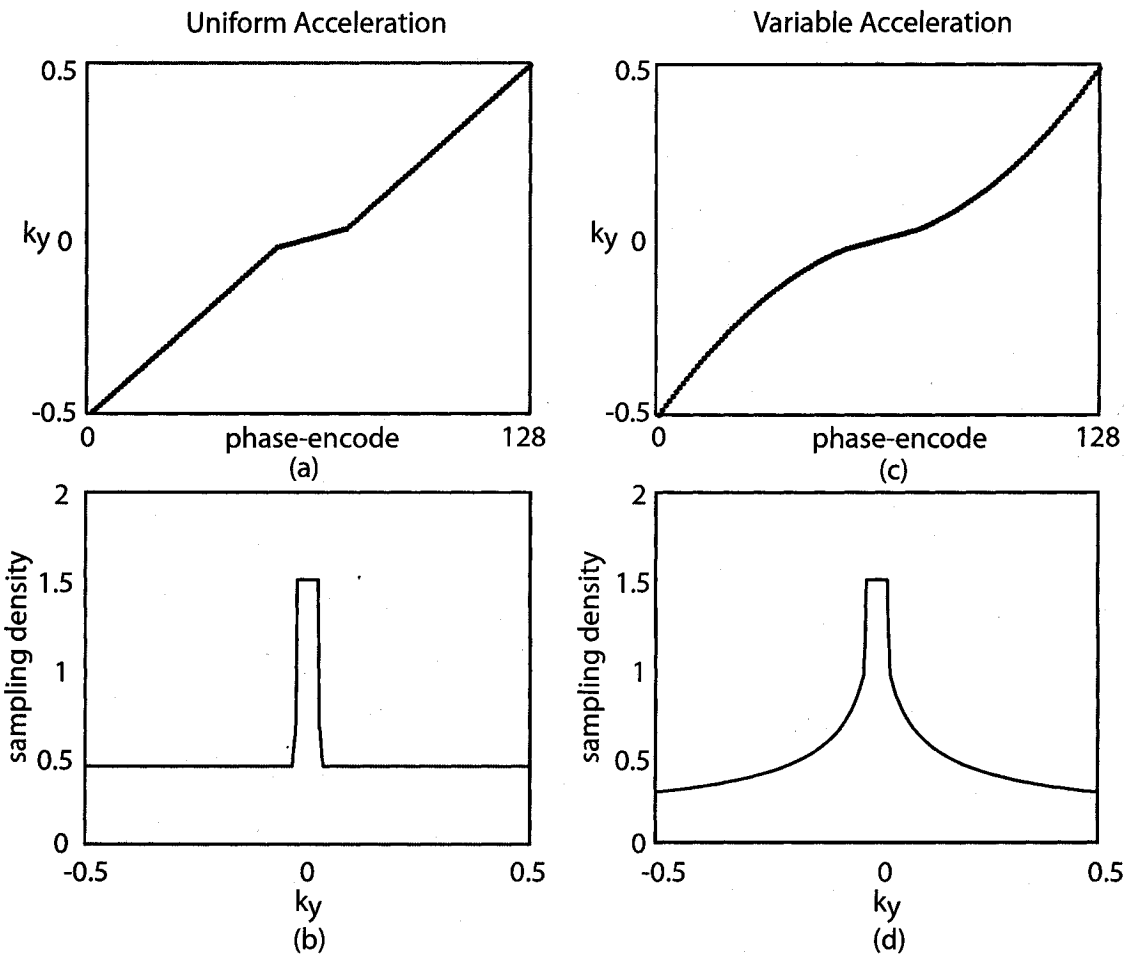


Figure 4.7: (a) Phase-encode locations for trajectory with uniform acceleration outside of the central region. (b) Sampling density of the trajectory in (a). (c) Phase-encode locations for trajectory with variable acceleration outside of the central region. (d) Sampling density of the trajectory in (c).

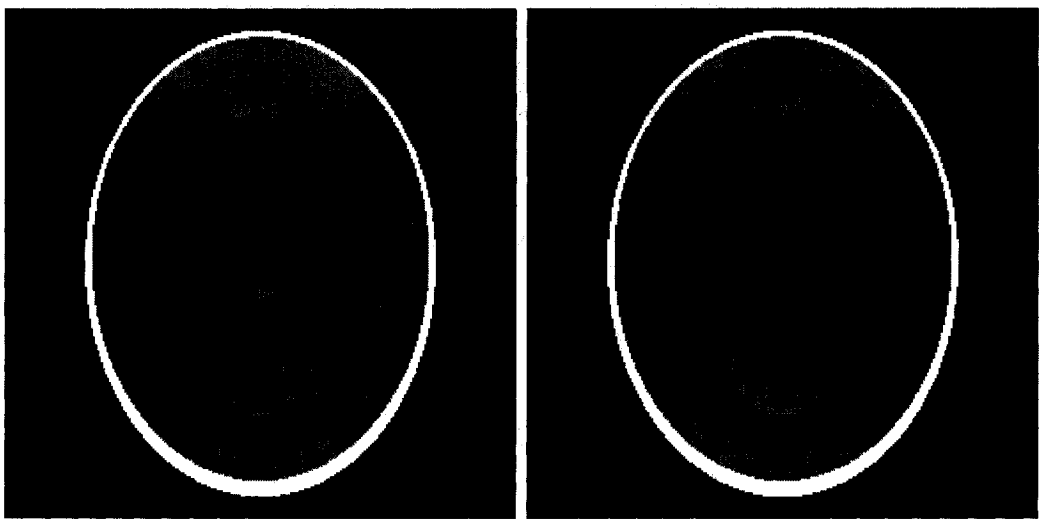


Figure 4.8: (a) Reconstruction of Shepp-Logan numerical phantom using the PARS method. (b) Reconstruction of the same data-set as in (a) using the APPEAR method.

images, they do indicate the accuracy with which unknown data values are synthesized. The error in a synthesized datum value can be found by multiplying the error vector image, corresponding to the encoding location of the synthesized datum, by the magnetization and summing the result. Thus, nonzero values in $\epsilon_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$ where there is no magnetization are unimportant.

Figure 4.9 compares the error in the estimated encoding vector at one encoding location, obtained using the APPEAR weights, to the error obtained using the PARS weights and to the error obtained using the weights found directly from Eq. 4.17. In the region where there is magnetization, the error in the estimated encoding vector obtained using the PARS weights, where the coil sensitivities are estimated from low spatial frequency data, is significantly larger than the error obtained using the APPEAR weights. As well, since APPEAR only minimizes the error in locations where there is magnetization, it is able to do a better job of estimating the encoding function at locations where there is magnetization than a strict application of Eq. 4.17.

The results of the phantom experiment are shown in Fig. 4.10. When the data from each coil is reconstructed separately by gridding, the aliasing artifacts due to sampling at a reduced FOV are evident. The variable acceleration trajectory reduces the structure of the aliasing artifact and spreads it throughout the image. Thus, while the PARS result for the uniform trajectory shows some residual artifact, the artifact is not noticeable for the PARS result with the variable acceleration trajectory. The cost of using a variable density trajectory is a reduction in the SNR efficiency [34]. The local projection calibration technique used by APPEAR is able to remove the aliasing artifact for both trajectories.

Figure 4.11 shows the *in vivo* results. Aliasing artifact is clearly visible in the gridding reconstructions and there is some aliasing of the skull into the middle of the image in the PARS reconstruction when the uniform acceleration trajectory is used. Once again, APPEAR is able to get rid of aliasing artifact for both trajectories.

4.5 Discussion

As a method for reconstructing images from partially parallel encoded acquisitions, the APPEAR method performs well. In the experiments presented, APPEAR had significantly less artifact than the PARS method. The main difference between PARS and APPEAR is that APPEAR uses local projection calibration whereas PARS uses the self-calibration technique [23], in which data in the central region is used to estimate the *in vivo* coil sensitivities. My results indicate that using the local projection calibration technique leads to less errors than using the self-calibration technique.

To understand these results, it is instructive to look more closely at the self-calibration technique used by PARS. PARS finds the local weights that satisfy

$$\min_{\mathbf{w}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})} \left\| \mathbf{w}_{j,\mathcal{L}(\mathbf{k})}^T(\mathbf{k}) E_{\mathrm{PARS},\mathcal{L}(\mathbf{k})} - \mathbf{e}_{\mathrm{PARS},j}^T(\mathbf{k}) \right\|, \tag{4.39}$$

which is similar to Eq. 4.17, except that $E_{\mathcal{L}(\mathbf{k})}$ and $\mathbf{e}_j^T(\mathbf{k})$ are replaced with $E_{\mathrm{PARS},\mathcal{L}(\mathbf{k})}$ and $\mathbf{e}_{\mathrm{PARS},j}^T(\mathbf{k})$. For the PARS encoding functions, the coil sensitivities are replaced with low resolution images reconstructed by inverse Fourier transform from the central

Figure 4.9: For the numerical simulation, the coil sensitivities are known. Here magnitude image of the error in the estimated encoding function, $\left|\epsilon_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})\right|$, for a coil on the right-hand side of the image and the unacquired k-space location $\mathbf{k} = [0, 21.3/\,\mathrm{FOV}]^T$, is shown for three different methods of finding the linear combination weights. (a) Knowing the coil sensitivity functions, weights are found by minimizing Eq. 4.17. (b) Weights are found using PARS, where the coil sensitivities are estimated from the central region data. (c) Weights are calculated using the APPEAR method. By minimizing the magnetization-weighted error in the estimated encoding function, APPEAR is very accurate in the area where signal is being acquired. All images have been windowed up 100X with respect to the magnitude of the encoding function being synthesized.

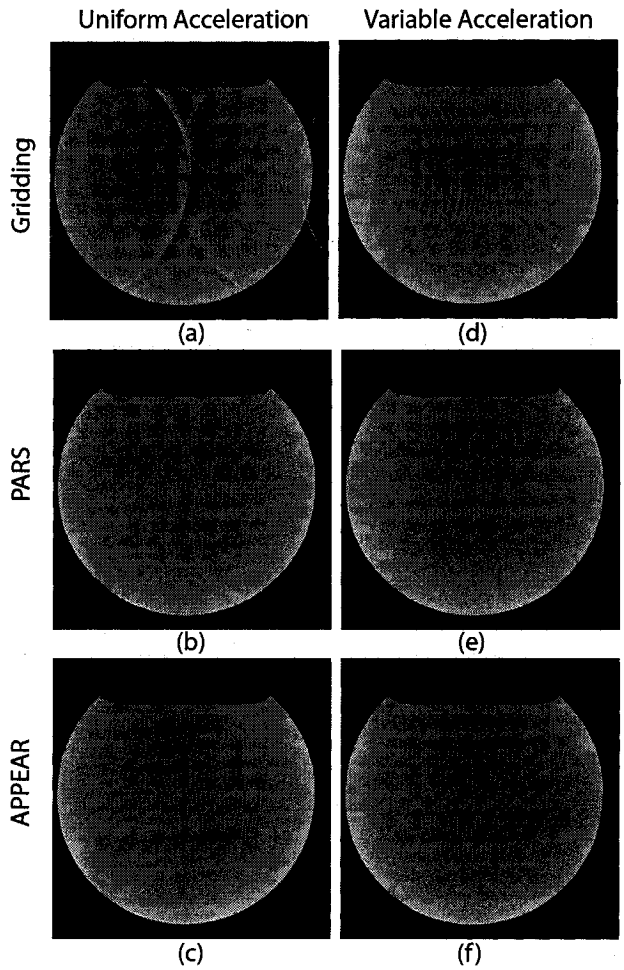


Figure 4.10: An axial slice of a ball phantom was scanned using an 8-channel high-resolution knee coil. Data was obtained using both trajectories shown in Fig. 4.7. In this experiment, the frequency encoding was done in the A/P direction. (a),(b),(c) were all reconstructed from the same data, as were (d),(e),(f).

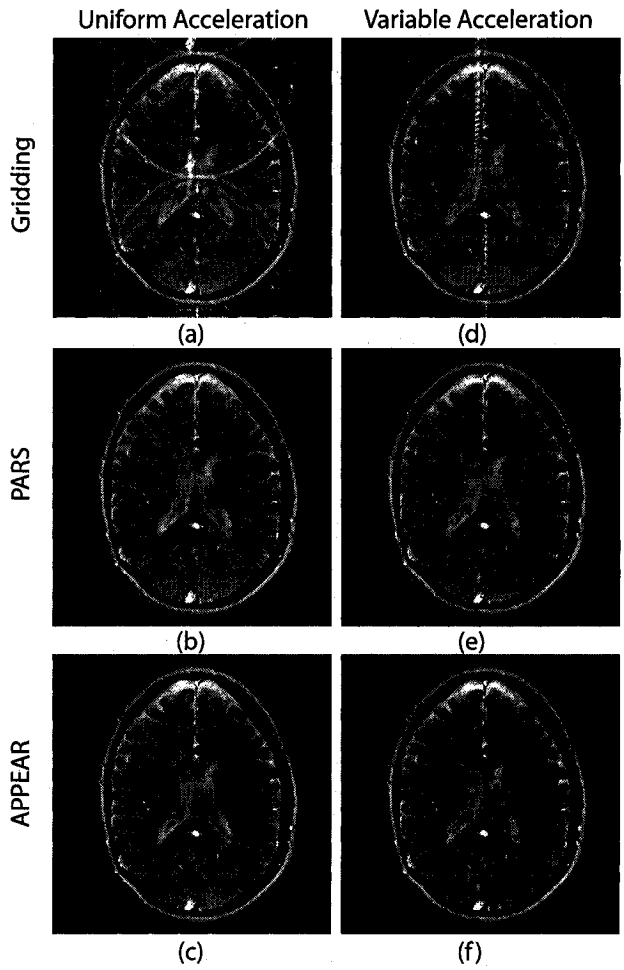


Figure 4.11: An axial brain slice was scanned using an 8-channel high-resolution head coil. Data was obtained using both trajectories shown in Fig. 4.7. In this experiment, the frequency encoding was done in the R/L direction. (a),(b),(c) were all reconstructed from the same data, as were (d),(e),(f). The artifact in (b) can be found by looking at the same location as the artifact in (a).

region data. These images can be written as $[m(\mathbf{r})s_j(\mathbf{r})] * \text{psf}(\mathbf{r})$, where $\text{psf}(\mathbf{r})$ is the point-spread-function associated with the sampling pattern in the central region. The point-spread-function can be modified by applying different weightings to the k-space sampling locations. As described in [23], a Kaiser window weighting can be used to reduce the Gibbs ringing in the resultant image. Regardless of the weights chosen for the k-space sampling locations, convolution with $\text{psf}(\mathbf{r})$ makes it difficult to extract $s_j(\mathbf{r})$. Self-calibration is built on the approximation that $s_j(\mathbf{r})$ is very low frequency and

$$[m(\mathbf{r})s_j(\mathbf{r})] * \text{psf}(\mathbf{r}) \approx [m(\mathbf{r}) * \text{psf}(\mathbf{r})]\, s_j(\mathbf{r}), \tag{4.40}$$

which separates $s_j(\mathbf{r})$ from the other terms. Note that Eq. 4.40 is simply Eq. 1 in [23], written in notation consistent with this paper. Once $s_j(\mathbf{r})$ is separated in this way, coil-by-coil images can be reconstructed [37]. The error in the approximation given in Eq. 4.40 tends to be largest near locations where $m(\mathbf{r})s_j(\mathbf{r})$ changes abruptly, such as at the edge of the object. These errors in turn lead to errors in the reconstruction matrix and estimated encoding functions. While local projection calibration limits itself to synthesizing estimated encoding functions from local neighborhoods, it is able to minimize Eq. 4.31 without approximation. Because of this, local projection calibration is not susceptible to errors due to the approximation in Eq. 4.40.

The autocalibration procedure used in Cartesian GRAPPA is a case of local projection calibration, where the pattern sets needed to find the synthesis weights align perfectly with the acquired data in the central region. However, the extension of the autocalibration technique to radial, spiral and VIPR trajectories [28, 29, 30, 31, 32, 33], is not a case of local projection calibration. These extensions of autocalibration focus on finding weights that optimize the fit between raw unprocessed scanner-acquired data, or data which has been interpolated in the time dimension, but do not insist on the integrity of the pattern set. This is very different from local projection calibration, which insists on the integrity of the pattern set, but accepts the use of synthesized (and apodized) data generated in the central calibration region from single-coil data in order to generate the linear combination weights used to synthesize unacquired data from acquired data on multiple coils.

When autocalibration is used for non-Cartesian trajectories, one is presented with the complication of finding acquired data in the necessary patterns for the calibration procedure. For radial [28] and VIPR [33] trajectories, the autocalibration procedure requires an initial full-FOV, full-resolution calibration scan. This is in contrast to both PARS and APPEAR, which are able to calibrate on the inherently oversampled central region of these trajectories. In addition, as noted by Samsonov *et al.* [38], the calibration data required by autocalibration for these trajectories often requires extensive time averaging, since calibration is performed in outer portions of k-space, where the signal-to-noise ratio is worse than the signal-to-noise ratio in the central region.

For the dual-density spiral trajectory [31, 32], autocalibration is able to calibrate on the central region. However, unlike PARS and APPEAR, which use the entire central region in finding the linear combination weights for a particular pattern set, the autocalibration procedure is only able to use a fraction of the central region, as shown in Fig. 4 in [32].

In addition, in order to find a sufficiently large number of data fits for calibration, these methods disregard small differences between pattern sets. For example, two pattern sets that differ by a small rotation or dilation might be treated as the same pattern set. In Fig. 4.12, it is shown that disregarding small differences between pattern sets degrades the quality of the estimated encoding vectors. To generate this figure, a common set of weights were found to minimize the magnetization-weighted errors in the encoding vectors for a collection of pattern sets which only differ by a slight rotation ($\pm 5°$). Comparing these results to Fig. 9, we see that this results in an error level similar to PARS. By using local projection calibration, APPEAR is able to achieve a smaller error level.

APPEAR provides a great deal of freedom in choosing the k-space trajectory used to acquire the data. While the two trajectories used in this paper have the same number of phase-encodes and hence same net acceleration, they have different properties. The variable acceleration trajectory has worse SNR compared to the uniform acceleration trajectory, but is more robust to errors in the estimated encoding vectors and less likely to produce images with visible aliasing artifacts.

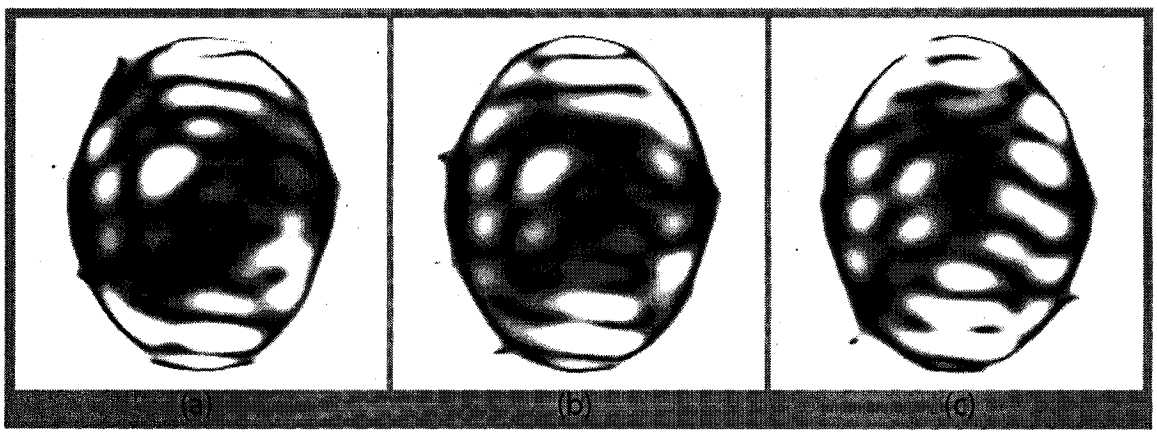


Figure 4.12: Disregarding small rotational differences in pattern sets. The pattern set around location $\mathbf{k} = [0, 21.3/\,\mathrm{FOV}]^T$ (Fig. 4.9) is rotated continuously through 10° (−5° to 5°), creating a collection of pattern sets which only differ by a slight rotation. Data is collected from the central region for each of these pattern sets. Then, treating the collection of pattern sets as a common pattern set, the data is concatenated and weights are found (Eq. 4.26). The magnitude image of the error in the estimated encoding function is calculated and windowed, as in Fig. 4.9. Shown, the error in the estimated encoding function for the pattern set rotated (a) −5°, (b) 0°, (c) 5°. By disregarding small rotational differences in pattern sets, as is done with autocalibration for non-Cartesian trajectories, these weights lead to significantly larger errors in the estimated encoding vectors. By ensuring the integrity of the pattern sets, APPEAR achieves much smaller errors in the estimated encoding vector as shown in Fig. 4.9(c).

The APPEAR method is unique in requiring an *oversampled* region in the center of k-space. For some trajectories, such as radial, VIPR and 3D cones [39], where oversampling is inherent, this is not a disadvantage. However, for many trajectories, this will require attention in the trajectory design. Specifically, the central region should be designed to critically sample a FOV larger than the extent of the object being imaged. In my experience, a FOV of 1.5 times the extent of the object is sufficient, but may not be optimal.

In this chapter, I have focused on describing the APPEAR method and providing some preliminary results. As the method matures, I am exploring the tradeoffs in the choice of parameters, such as the extent of the local neighborhood and central region. In addition, since APPEAR, like PARS, finds a unique set of weights in order to synthesize each unacquired datum, it is a computationally intensive method. The following chapter discusses the computation required by APPEAR in more detail.

It should also be noted that the utility of the APPEAR method is to reduce the encoding time necessary to acquire an image and as such, should only be employed when the encoding time needs to be reduced. Many scans are not limited by encoding time, but by the time required to obtain sufficient SNR [40]. In this case, the APPEAR method is of limited utility. Indeed, the first use of multiple receiving coils in MRI was to increase SNR, not to reduce encoding time [41].

Still, there are many cases when it is advantageous to reduce the encoding time. One case is cardiac and body imaging where motion during the acquisition can lead to artifacts. Another case is contrast enhanced imaging where high signal levels are present, but only for a limited time frame. For such cases, the results presented here indicate that APPEAR will provide high quality reconstructions. APPEAR is simple to use, since no coil sensitivities are needed and APPEAR is non-iterative. This simplicity does not limit the flexibility of APPEAR, which is able to reconstruct arbitrary k-space trajectories.

4.6 Conclusions

The theory behind local projection calibration has been derived. This theory shows that local projection calibration will attain near optimal linear combination weights for data synthesis, even though the coil sensitivities are not known. This theory also shows that there is a fundamental difference between local projection calibration and techniques which estimate coil sensitivities from low spatial frequency data. The results of this chapter indicate that local projection calibration produces reconstructions with less aliasing artifact than reconstructions which use estimated coil sensitivities from low spatial frequency data.

In this chapter, the APPEAR method is introduced as a way to take advantage of local projection calibration with arbitrary k-space trajectories. APPEAR is a straightforward method that does not require iteration. Preliminary experiments confirm that APPEAR is able to produce high quality reconstructions.

Chapter 5

Correlation Values

5.1 Introduction

The previous chapter focused on the image quality benefits of the APPEAR method, giving only passing reference to the computation requirements. In this chapter the computation required by a straightforward implementation of APPEAR is analyzed and the concept of *correlation values* is developed and used to improve the computational efficiency of APPEAR by a factor of approximately 10X. Although the concept of correlation values was developed while looking to improve the computational efficiency of APPEAR, correlation values also lead to a novel way of viewing the reconstruction problem that can lead to useful insights into image reconstruction methods. In this chapter the concept of correlation values is developed and used to provide insight into synthesis errors, reconstruction regularization, limitations in the gridding method and how local projection calibration relates to calibration by estimating coil sensitivity functions. A 2-D version of APPEAR is implemented and phantom and *in-vivo* results are obtained for variable-density spiral trajectories.

Computationally, APPEAR is similar to the PARS method in that both methods generate a set of linear combination weights, used to synthesize an unacquired datum, by solving a linear system. Straightforward implementations of both APPEAR and PARS can have a heavy computational burden for non-Cartesian trajectories because many sets of linear combination weights must be generated. For a 2-D non-Cartesian

trajectory such as a spiral trajectory, where a unique set of linear combination weights is found for each unacquired k-space location to be synthesized, this can involve solving tens of thousands of linear systems and over an hour to reconstruct one 2-D image [22].

One proposal for reducing the computation time of PARS for the radial trajectory is given by Samsonov et al. [38]. In this proposal, linear combination weights are found for a fraction of the unacquired k-space locations and these linear combination weights are then interpolated to generate the linear combination weights for the remaining unacquired k-space locations. Such an approach has similarities to non-Cartesian GRAPPA in that both methods reduce the computational burden of reconstruction by reducing the number of pattern sets for which solving a system of linear equations is used to find the linear combination weights. Since the only difference between PARS and APPEAR is how the linear combination weights are generated for a given pattern set, such an approach can be used with APPEAR. In such a scheme, APPEAR could be used to calculate linear combination weights for a fraction of the pattern sets and the weights for the remaining pattern sets could be found using interpolation.

In this chapter, I focus on a complementary avenue to reducing the computation of APPEAR: reducing the computation necessary to generate the weights for a given pattern set by reducing the size of the linear system that must be solved. Just as the scheme proposed by Samsonov for speeding up PARS can be used to speed up APPEAR, so too can the scheme proposed in this chapter be used to speed up PARS and GRAPPA. In addition, the scheme proposed in this chapter can further reduce the computation required by the Samsonov scheme, by reducing the computation required to calculate the fraction of linear combination weights that are not derived from interpolation.

The following section provides an analysis of the computation required by a straightforward implementation of APPEAR. Next, correlation values are introduced and it is shown that a linear system used to find linear combination weights can be distilled to a system composed entirely of correlation values. For scans that are

entirely gradient encoded, it is shown that simple analytic expressions for the correlation values can be found, leading to fast computation. This finding is significant for APPEAR, which must interpolate data in the central region. Since gridding is commonly viewed as a k-space interpolation technique, a section is devoted to comparing gridding with interpolation based on correlation values (local projection interpolation). While correlation values are both pedagogically useful and useful when only gradient encoding is considered, they are not immediately useful in dealing with sensitivity encoding, where the encoding functions are not known. However, many parallel imaging reconstruction methods can be expressed in terms of *approximate* correlation values. Methods that are amenable to being expressed in terms of approximate correlation values include those that estimate the coil sensitivity functions as well as the APPEAR method, which uses local projection calibration. The difference between local projection calibration and coil sensitivity estimation is studied by examining the difference in how these methods approximate correlation values. Following this, techniques for efficiently computing approximate correlation values are described. Finally, an implementation of APPEAR that takes advantage of correlation values is described and results are shown for variable-density spiral imaging.

5.2 Analysis of Computation

Like PARS and GRAPPA, the APPEAR method can be split into two steps:

1. Calibration: find the linear combination weights.
2. Synthesis: synthesize unacquired data from surrounding k-space data and the relevant linear combination weights.

The fact that direct methods such as APPEAR can be split into calibration and synthesis steps makes them attractive, especially for real-time applications. With such methods, the calibration can be performed once and used to synthesize multiple images. Since the synthesis step is computationally similar to gridding, it can be performed in real-time on modern computers for 2-D images and a moderate number

of receiver coils (for example, eight receiver coils). Iterative methods such as conjugate gradient (CG) SENSE cannot be split up in this way, requiring iteration for each frame. Nevertheless, for non-Cartesian trajectories CG-SENSE is often much faster than APPEAR since the calibration step used in APPEAR is very computationally intensive. As most of the computation is contained in the calibration step, I focus on this step for the remainder of this section.

The calibration step consists of finding linear combination weights for each unique pattern set. Since the problem of finding linear combination weights for one pattern set is independent of finding the linear combination weights for a different pattern set, the solution for each pattern set can be computed separately, making it easy to parallelize the calibration computation. This is an advantage over iterative CG-SENSE where the iterations must proceed serially. While APPEAR has the attractive features of separating the calibration from the synthesis and being easy to parallelize, it has the drawback that the amount of computation required can be too large for some applications even when multiple processors are used to parallelize the computation.

The amount of computation grows with the number of unique pattern sets for which linear combination weights need to be found. For a Cartesian k-space trajectory with an acceleration of two outside the central region, calibration only needs to be performed for a single pattern set. When the acceleration outside the central region is increased to three, calibration must be performed on two pattern sets. Calibration of Cartesian k-space trajectories is not prohibitive since linear combination weights need to be found for only a handful of pattern sets.

For non-Cartesian trajectories, APPEAR takes the same approach as PARS, finding a unique set of linear combination weights for each data location to be synthesized. This significantly increases the computation necessary for the calibration step: for a spiral trajectory with an FOV/resolution ratio of 256, this entails computing linear combination weights for about 50,000 different pattern sets. To understand how much computation this requires, it is necessary to look at the computation needed to calculate the linear combination weights for one pattern set.

By far, the majority of the computation in calculating the linear combination weights for a pattern set using Eq. 4.34 is contained in two steps: synthesizing

pattern set data for a full-FOV grid of $\mathbf{k}'$ locations in the fit region and multiplying

$$\left[\begin{array}{ccc} \widetilde{\mathbf{d}}_{\mathcal{P}(\mathbf{k},\mathbf{k}'_1)} & \cdots & \widetilde{\mathbf{d}}_{\mathcal{P}(\mathbf{k},\mathbf{k}'_{N_{\mathbf{k}'}})} \end{array} \right] \left[\begin{array}{c} \widetilde{\mathbf{d}}^H_{\mathcal{P}(\mathbf{k},\mathbf{k}'_1)} \\ \vdots \\ \widetilde{\mathbf{d}}^H_{\mathcal{P}(\mathbf{k},\mathbf{k}'_{N_{\mathbf{k}'}})} \end{array} \right] \tag{5.1}$$

which is the computationally significant step in calculating the pseudo-inverse of $\left[\widetilde{\mathbf{d}}_{\mathcal{P}(\mathbf{k},\mathbf{k}'_1)} \cdots \widetilde{\mathbf{d}}_{\mathcal{P}(\mathbf{k},\mathbf{k}'_{N_{\mathbf{k}'}})}\right]$ since generally $N_{\mathbf{k}'}$, the number of $\mathbf{k}'$ locations that fall on a full-FOV Cartesian grid in the fit region, is much larger than $N_{\mathcal{L}(\mathbf{k})}$, the length of $\widetilde{\mathbf{d}}_{\mathcal{P}(\mathbf{k},\mathbf{k}')}$. Synthesizing the pattern set data in the central region is proportional to $N_{\mathbf{k}'}N_{\mathcal{L}(\mathbf{k})}$ and the matrix multiplication is proportional to $N_{\mathbf{k}'}N^2_{\mathcal{L}(\mathbf{k})}$. For both of these steps, the computation grows linearly with $N_{\mathbf{k}'}$.

While synthesizing the pattern set data is a step unique to APPEAR, the matrix multiplication step is also performed in GRAPPA. In order to reduce the computation time for the calibration step, it is not uncommon for GRAPPA implementations to use a smaller $N_{\mathbf{k}'}$, the number of fits in the fit region. The downside of such an approach is that the quality of the calibration can suffer from calibrating with a reduced fit region. In practice, this is not a significant downside, since trajectories used with Cartesian GRAPPA typically have a fit region that goes to the edge of sampled k-space in the readout direction. GRAPPA implementations that do not use the portions of the fit region that are near the edges of sampled k-space in the readout direction still calibrate effectively since the outer portions of k-space contain only a small portion of the energy of the image. Nevertheless, one of the contributions of correlation values is to make the computation less dependent on $\mathbf{k}'$.

It is helpful at this point to compare the computation required by APPEAR to that required by PARS. Recall that PARS solves Eq. 4.39 to generate the linear combination weights:

$$\mathbf{w}^T_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k}) = \mathbf{e}^T_{\mathrm{PARS},j}(\mathbf{k}) E^{\dagger}_{\mathrm{PARS},\mathcal{L}(\mathbf{k})}. \tag{5.2}$$

Since $E_{\mathrm{PARS},\mathcal{L}(\mathbf{k})}$ is a very fat matrix, with the number of columns equal to N_v, the number of voxels in the image and the number of rows equal to $N_{\mathcal{L}(\mathbf{k})}$, the number of elements in the local set, the most computationally significant part of this calculation is multiplying $E_{\mathrm{PARS},\mathcal{L}(\mathbf{k})}E^H_{\mathrm{PARS},\mathcal{L}(\mathbf{k})}$ in calculating the pseudo-inverse of $E_{\mathrm{PARS},\mathcal{L}(\mathbf{k})}$. Thus, the cost for PARS to compute a set of linear combination weights is proportional to $N_v N^2_{\mathcal{L}(\mathbf{k})}$. Since N_v is typically much larger than $N_{\mathbf{k}'}$, a straightforward implementation of PARS is typically more computationally intensive than a straightforward implementation of APPEAR.

The computation required by APPEAR and PARS is similar, both being proportional to $N^2_{\mathcal{L}(\mathbf{k})}$; they differ in that APPEAR is also proportional to $N_{\mathbf{k}'}$ whereas PARS is proportional to N_v. Using the correlation value, the dependence of the computation on $N_{\mathbf{k}'}$ and N_v can be largely removed, leaving only the common (and typically much smaller) dependence on $N_{\mathcal{L}(\mathbf{k})}$.

5.3 What are Correlation Values?

A correlation value is a complex number expressing the mutual relation between two encoding vectors. Correlation values are useful because the optimal linear combination weights for a given pattern set can be found quickly once the correlation values that relate the members of the pattern set are known. The correlation value relating encoding location $(j_1, \mathbf{k}_1)$ to encoding location $(j_2, \mathbf{k}_2)$ is given by an inner or dot product between the corresponding weighted encoding functions:

$$c(j_1, \mathbf{k}_1; j_2, \mathbf{k}_2) = \int \widetilde{e}^*_{j_1}(\mathbf{k}_1, \mathbf{r})\, \widetilde{e}_{j_2}(\mathbf{k}_2, \mathbf{r})\; d\mathbf{r}, \tag{5.3}$$

where

$$\widetilde{e}_j(\mathbf{k}, \mathbf{r}) = f(\mathbf{r}) e_j(\mathbf{k}, \mathbf{r}) \tag{5.4}$$

is an encoding function where each spatial location has been weighted by $f(\mathbf{r})$. The weighting function $f(\mathbf{r})$ allows more importance to be placed on the correlation between encoding functions at certain spatial locations (for example within the field-of-view or where more magnetization is expected). A discussion of how $f(\mathbf{r})$ is chosen

for a specific application is dealt with in the following sections. For discrete encoding vectors, the correlation value can be written as

$$c(j_1, \mathbf{k}_1; j_2, \mathbf{k}_2) = \widetilde{\mathbf{e}}_{j_1}^H(\mathbf{k}_1)\, \widetilde{\mathbf{e}}_{j_2}(\mathbf{k}_2)\,. \tag{5.5}$$

The problem formulation for finding optimal linear combination weights for a local set can be written completely in terms of correlation values. Before showing this, it is helpful to develop a simplified notation. A local set $\mathcal{L}(\mathbf{k})$, defined in Eq. 4.14 and illustrated in Fig. 4.1(b) contains $N_{\mathcal{L}(\mathbf{k})}$ encoding vectors, corresponding to the acquired data locations *for all coils* in a radius κ around $\mathbf{k}$. To simplify the notation when dealing with encoding locations within a local set, each encoding location can be specified by an index i in the local set. In this scheme, encoding location i would be synonymous with encoding location $(j_i, \mathbf{k}_i)$, the ith encoding location in $\mathcal{L}(\mathbf{k})$. The index i is in the range $1 \ldots N_{\mathcal{L}(\mathbf{k})}$ and can be used to simplify the notation for encoding vectors and correlation values as:

$$\begin{aligned}
\mathbf{e}_{j_i}(\mathbf{k}_i) &= \mathbf{e}_i, \\
\widetilde{\mathbf{e}}_{j_i}(\mathbf{k}_i) &= \widetilde{\mathbf{e}}_i, \\
c(j_i, \mathbf{k}_i; j, \mathbf{k}) &= c(i; j, \mathbf{k}) \text{ and} \\
c(j_{i_1}, \mathbf{k}_{i_1}; j_{i_2}, \mathbf{k}_{i_2}) &= c(i_1; i_2)\,.
\end{aligned}$$

Using this simplified notation, I now proceed to express the optimal linear combination weights in terms of correlation values. I start with a version of Eq. 4.17, modified such that the error in the estimated encoding function is weighted by $\mathbf{f}$, the vector version of $f(\mathbf{r})$:

$$\min_{\mathbf{w}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})} \left\| \boldsymbol{\epsilon}_{j,\mathcal{L}(\mathbf{k})}^T(\mathbf{k}) \cdot \mathbf{f}^T \right\| = \min_{\mathbf{w}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})} \left\| \mathbf{w}_{j,\mathcal{L}(\mathbf{k})}^T(\mathbf{k}) \widetilde{E}_{\mathcal{L}(\mathbf{k})} - \widetilde{\mathbf{e}}_j^T(\mathbf{k}) \right\|, \tag{5.6}$$

where

$$\widetilde{E}_{\mathcal{L}(\mathbf{k})} = \begin{bmatrix} \widetilde{\mathbf{e}}_1^T \\ \vdots \\ \widetilde{\mathbf{e}}_{N_{\mathcal{L}(\mathbf{k})}}^T \end{bmatrix}. \tag{5.7}$$

Including the $\mathbf{f}$ weighting makes Eq. 5.6 more flexible than Eq. 4.17. For example, if it is known *a priori* that the magnetization is encompassed in a circular FOV, $\mathbf{f}$ can be used to minimize the error only within this FOV, resulting in less error within the FOV, where the error can affect the image quality. Finding the linear combination weights $\mathbf{w}_{j,\mathcal{L}(\mathbf{k})}^T(\mathbf{k})$ that minimize Eq. 5.6 is a least-squares problem that can be solved as:

$$\mathbf{w}_{j,\mathcal{L}(\mathbf{k})}^T(\mathbf{k}) = \widetilde{\mathbf{e}}_j^T(\mathbf{k})\widetilde{E}_{\mathcal{L}(\mathbf{k})}^{\dagger} \tag{5.8}$$

$$= \widetilde{\mathbf{e}}_j^T(\mathbf{k})\widetilde{E}_{\mathcal{L}(\mathbf{k})}^H \left(\widetilde{E}_{\mathcal{L}(\mathbf{k})}\widetilde{E}_{\mathcal{L}(\mathbf{k})}^H\right)^{-1}. \tag{5.9}$$

Taking the transpose of Eq. 5.9 and expanding $\widetilde{E}_{\mathcal{L}(\mathbf{k})}$ as in Eq. 5.7, $\mathbf{w}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$ can be

written as

$$\mathbf{w}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k}) = \left(\tilde{\mathbf{e}}_j{}^T(\mathbf{k}) \begin{bmatrix} \tilde{\mathbf{e}}_1^T \\ \vdots \\ \tilde{\mathbf{e}}_{N_{\mathcal{L}(\mathbf{k})}}^T \end{bmatrix}^H \left(\begin{bmatrix} \tilde{\mathbf{e}}_1^T \\ \vdots \\ \tilde{\mathbf{e}}_{N_{\mathcal{L}(\mathbf{k})}}^T \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{e}}_1^* & \dots & \tilde{\mathbf{e}}_{N_{\mathcal{L}(\mathbf{k})}}^* \end{bmatrix} \right)^{-1} \right)^T \quad (5.10)$$

$$= \left(\begin{bmatrix} \tilde{\mathbf{e}}_1^H \\ \vdots \\ \tilde{\mathbf{e}}_{N_{\mathcal{L}(\mathbf{k})}}^H \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{e}}_1 & \dots & \tilde{\mathbf{e}}_{N_{\mathcal{L}(\mathbf{k})}} \end{bmatrix} \right)^{-1} \begin{bmatrix} \tilde{\mathbf{e}}_1^H \\ \vdots \\ \tilde{\mathbf{e}}_{N_{\mathcal{L}(\mathbf{k})}}^H \end{bmatrix} \tilde{\mathbf{e}}_j(\mathbf{k}) \quad (5.11)$$

$$= \begin{bmatrix} \tilde{\mathbf{e}}_1^H\tilde{\mathbf{e}}_1 & \dots & \tilde{\mathbf{e}}_1^H\tilde{\mathbf{e}}_{N_{\mathcal{L}(\mathbf{k})}} \\ \vdots & \ddots & \vdots \\ \tilde{\mathbf{e}}_{N_{\mathcal{L}(\mathbf{k})}}^H\tilde{\mathbf{e}}_1 & \dots & \tilde{\mathbf{e}}_{N_{\mathcal{L}(\mathbf{k})}}^H\tilde{\mathbf{e}}_{N_{\mathcal{L}(\mathbf{k})}} \end{bmatrix}^{-1} \begin{bmatrix} \tilde{\mathbf{e}}_1^H\tilde{\mathbf{e}}_j(\mathbf{k}) \\ \vdots \\ \tilde{\mathbf{e}}_{N_{\mathcal{L}(\mathbf{k})}}^H\tilde{\mathbf{e}}_j(\mathbf{k}) \end{bmatrix} \quad (5.12)$$

$$= \begin{bmatrix} c(1;1) & \dots & c(1;N_{\mathcal{L}(\mathbf{k})}) \\ \vdots & \ddots & \vdots \\ c(N_{\mathcal{L}(\mathbf{k})};1) & \dots & c(N_{\mathcal{L}(\mathbf{k})};N_{\mathcal{L}(\mathbf{k})}) \end{bmatrix}^{-1} \begin{bmatrix} c(1;j,\mathbf{k}) \\ \vdots \\ c(N_{\mathcal{L}(\mathbf{k})};j,\mathbf{k}) \end{bmatrix} \quad (5.13)$$

$$= \mathbf{C}_{\mathrm{aa}}^{-1}\mathbf{c}_{\mathrm{as}}, \quad (5.14)$$

where

$$\mathbf{C}_{\mathrm{aa}} = \begin{bmatrix} c(1;1) & \dots & c(1;N_{\mathcal{L}(\mathbf{k})}) \\ \vdots & \ddots & \vdots \\ c(N_{\mathcal{L}(\mathbf{k})};1) & \dots & c(N_{\mathcal{L}(\mathbf{k})};N_{\mathcal{L}(\mathbf{k})}) \end{bmatrix}, \quad (5.15)$$

$$\mathbf{c}_{\mathrm{as}} = \begin{bmatrix} c(1;j,\mathbf{k}) \\ \vdots \\ c(N_{\mathcal{L}(\mathbf{k})};j,\mathbf{k}) \end{bmatrix}. \quad (5.16)$$

The correlation matrix $\mathbf{C}_{\mathrm{aa}}$ is composed of correlation values that relate every member of the local set to every other member. The subscript indicates that the matrix relates *analysis* locations to other *analysis* locations. The members of the local set are considered analysis locations, as acquiring data at these locations can be viewed as analyzing the magnetization, each acquired datum measuring a component

of the overall magnetization. The correlation matrix $\mathbf{C}_{\mathrm{aa}}$ is also Hermitian ($\mathbf{C}_{\mathrm{aa}}^{H} = \mathbf{C}_{\mathrm{aa}}$) and positive definite (all of the eigenvalues of $\mathbf{C}_{\mathrm{aa}}$ are positive).

The correlation vector $\mathbf{c}_{\mathrm{as}}$ is composed of correlation values that relate every member of the local set (analysis locations) to the unacquired location $(j, \mathbf{k})$. The subscript indicates that the vector relates analysis locations to a *synthesis* location. The unacquired location is also referred to as the synthesis location since the datum at this location is synthesized from the data at the analysis locations.

Once the correlation values that populate $\mathbf{C}_{\mathrm{aa}}$ and $\mathbf{c}_{\mathrm{as}}$ have been generated, the computation time for computing the linear combination weights $\mathbf{w}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$ is proportional to $N_{\mathcal{L}(\mathbf{k})}^{3}$, exchanging the $N_{\mathbf{k}'}$ dependence in a straightforward implementation of APPEAR for a dependence on the far smaller $N_{\mathcal{L}(\mathbf{k})}$.

5.4 Fast Computation of Correlation Values

In this section, I show how correlation values can be computed quickly when the encoding functions are known and some or all of the encoding is gradient (Fourier) encoding.

When only gradient encoding is employed, the encoding functions have the form:

$$e(\mathbf{k}, \mathbf{r}) = g(\mathbf{k}, \mathbf{r}) \tag{5.17}$$

$$= e^{-i2\pi\mathbf{k}\cdot\mathbf{r}}. \tag{5.18}$$

In this case, the weighting function $f(\mathbf{r})$ should be chosen to take into account the correlation only within the FOV of interest. This can be accomplished with a rect function. I start with a simple 1-D example, where

$$f(x) = \sqrt{\frac{1}{\mathrm{FOV}}}\,\sqcap\left(\frac{x}{\mathrm{FOV}}\right). \tag{5.19}$$

Substituting Eqs. 5.18 and 5.19 into Eq. 5.3 gives

$$\begin{aligned} c(k_{x,1};k_{x,2}) &= \int \sqrt{\frac{1}{\mathrm{FOV}}}\sqcap\left(\frac{x}{\mathrm{FOV}}\right)e^{i2\pi k_{x,1}x} \\ &\quad \sqrt{\frac{1}{\mathrm{FOV}}}\sqcap\left(\frac{x}{\mathrm{FOV}}\right)e^{-i2\pi k_{x,2}x}dx \qquad (5.20) \\ &= \int \frac{1}{\mathrm{FOV}}\sqcap\left(\frac{x}{\mathrm{FOV}}\right)e^{-i2\pi(k_{x,2}-k_{x,1})x}dx \qquad (5.21) \\ &= \mathrm{FT}\left\{\frac{1}{\mathrm{FOV}}\sqcap\left(\frac{x}{\mathrm{FOV}}\right)\right\}(k_{x,2}-k_{x,1}) \qquad (5.22) \\ &= \mathrm{sinc}(\mathrm{FOV}(k_{x,2}-k_{x,1})). \qquad (5.23) \end{aligned}$$

For 2-D imaging, when a circular FOV is chosen,

$$c(\mathbf{k}_1;\mathbf{k}_2) = \mathrm{jinc}(\mathrm{FOV}\,\|\mathbf{k}_2-\mathbf{k}_1\|) \qquad (5.24)$$

and when a rectangular FOV is chosen

$$c(\mathbf{k}_1;\mathbf{k}_2) = \mathrm{sinc}(\mathrm{FOV}_x(k_{x,2}-k_{x,1}))\,\mathrm{sinc}(\mathrm{FOV}_y(k_{y,2}-k_{y,1})). \qquad (5.25)$$

For spiral scans, which have a circular FOV, central region interpolation used in APPEAR takes advantage of Eq. 5.24 to reduce the computation for central region interpolation.

When receiver coil sensitivity encoding is also employed, the encoding functions can be written as

$$\begin{aligned} e_j(\mathbf{k},\mathbf{r}) &= s_j(\mathbf{r})g(\mathbf{k},\mathbf{r}) \qquad (5.26) \\ &= s_j(\mathbf{r})e^{-i2\pi\mathbf{k}\cdot\mathbf{r}}. \qquad (5.27) \end{aligned}$$

Substitution into Eq. 5.3 yields

$$c(j_1, \mathbf{k}_1; j_2, \mathbf{k}_2) = \int f^*(\mathbf{r}) s^*_{j_1}(\mathbf{r}) e^{i2\pi \mathbf{k}_1 \cdot \mathbf{r}} f(\mathbf{r}) s_{j_2}(\mathbf{r}) e^{-i2\pi \mathbf{k}_2 \cdot \mathbf{r}} d\mathbf{r} \quad (5.28)$$

$$= \int |f(\mathbf{r})|^2 s^*_{j_1}(\mathbf{r}) s_{j_2}(\mathbf{r}) e^{-i2\pi (\mathbf{k}_2 - \mathbf{k}_1) \cdot \mathbf{r}} dr \quad (5.29)$$

$$= \mathrm{FT}\left\{ |f(\mathbf{r})|^2 s^*_{j_1}(\mathbf{r}) s_{j_2}(\mathbf{r}) \right\} (\mathbf{k}_2 - \mathbf{k}_1). \quad (5.30)$$

Thus, when the coil sensitivity of each coil is known, correlation values can be quickly computed using the Fourier transform.

Choosing $f(\mathbf{r}) = m(\mathbf{r})$ corresponds to the weighting used by *in vivo* coil sensitivities, where images acquired by the coils are used in place of coil sensitivities. Correlation values for the PARS method, which uses low resolution *in vivo* coils sensitivities, can be efficiently calculated using Eq. 5.30.

Thus far, correlation values have been introduced and it was shown that the linear combination weights used in local projection interpolation could be generated by solving a problem composed of correlation values. This section demonstrated techniques for quickly computing correlation values in useful cases when the encoding functions are known. The next section explores correlation values from a different angle, giving them a graphical interpretation that can make it more intuitive to work with correlation values.

5.5 Graphical Interpretation and Regularization

Expressing image space as discrete voxels allows us to express the encoding functions and magnetization in terms of vectors, making it possible to express the reconstruction computation in terms of vector and matrix operations. A further advantage of vectors is that they also have a simple geometric interpretation as a line segment in a multi-dimensional space. In this section, I use the graphical interpretation of vectors to give a visual picture for correlation values as well as local projection interpolation and calibration. The use of regularization with APPEAR is introduced and justified using a graphical approach.

Both the magnetization vector $\mathbf{m}$ and all encoding vectors $\mathbf{e}_j(\mathbf{k})$ are vectors in the same N_v-dimensional "voxel" space. Voxel space is a multi-dimensional space, where each axis corresponds to a single voxel value. In fact, since voxel values are complex-valued in MRI, each voxel is defined by two axis and the dimensionality of $\mathbf{m}$ and $\mathbf{e}_j(\mathbf{k})$ is really $2 \times N_v$. The doubling of the number of dimensions due to complex values is ignored in this section, simplifying the explanations without any loss in generality. Still, the dimensionality of voxel space N_v is usually very large: for a 2-D image, N_v might be $256 \times 256 = 65,536$ and for a 3-D image, N_v might be $256 \times 256 \times 256 = 16,777,216$. Because of the difficulty in viewing such a large vector space, simple examples in two and three dimensions are chosen to develop a graphical interpretation of the mathematical operations being performed. Though these examples are trivial, they can help develop a useful intuition into reconstruction methods.

In a vector space view, a datum value can be seen as measuring the projection of the magnetization vector along the direction of the encoding vector, when the encoding vector is of unit magnitude. When the encoding vector is not of unit magnitude, the data value is scaled by the magnitude of the encoding vector. As shown in Fig. 5.1, a geometric view of encoding the magnetization is of projecting the magnetization along numerous encoding vectors, each pointing in a different direction. When an unacquired datum $\widehat{d}_j(\mathbf{k})$ is synthesized by computing a linear combination of acquired data, the encoding vector $\widehat{\mathbf{e}}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$ for this unacquired datum is a linear combination of the encoding vectors corresponding to the acquired data (the encoding vectors constituting $E_{\mathcal{L}(\mathbf{k})}$). Because of this, $\widehat{\mathbf{e}}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$ is always in the subspace spanned by $E_{\mathcal{L}(\mathbf{k})}$. However, the actual encoding vector of the target synthesis location $\mathbf{e}_j(\mathbf{k})$ is not necessarily in the subspace spanned by $E_{\mathcal{L}(\mathbf{k})}$ and when $\mathbf{e}_j(\mathbf{k})$ is not in the subspace spanned by the encoding vectors of the local set, there is necessarily some error in the estimated encoding vector, $\boldsymbol{\epsilon}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$. Local projection calibration draws its name from the fact that $\widehat{\mathbf{e}}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$ is a projection of $\mathbf{e}_j(\mathbf{k})$ onto the subspace spanned by the local set, as shown in Fig. 5.2. The error in $\widehat{\mathbf{e}}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$ can be larger than necessary if a non-ideal set of linear combination weights is used, as shown in Fig. 5.3. The objective of the calibration step is to generate a set of linear combination

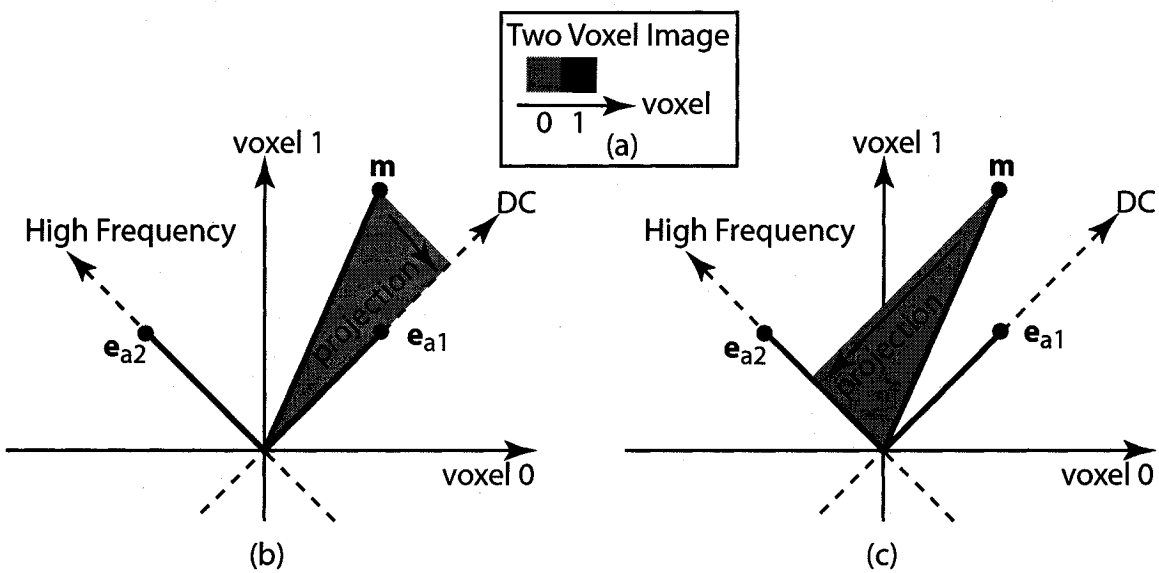


Figure 5.1: Graphical interpretation of gradient encoding a simple two voxel image. (a) The image consists of only two voxels, voxel 0 and voxel 1. This image can be fully encoded using two Fourier components: a low frequency sum of the voxels and a high frequency difference of the voxels. Encoding can be viewed graphically as a projection in voxel space of the magnetization vector $\mathbf{m}$ along the direction of the low frequency encoding vector $\mathbf{e}_{a1}$ (b) and high frequency encoding vector $\mathbf{e}_{a2}$ (c).

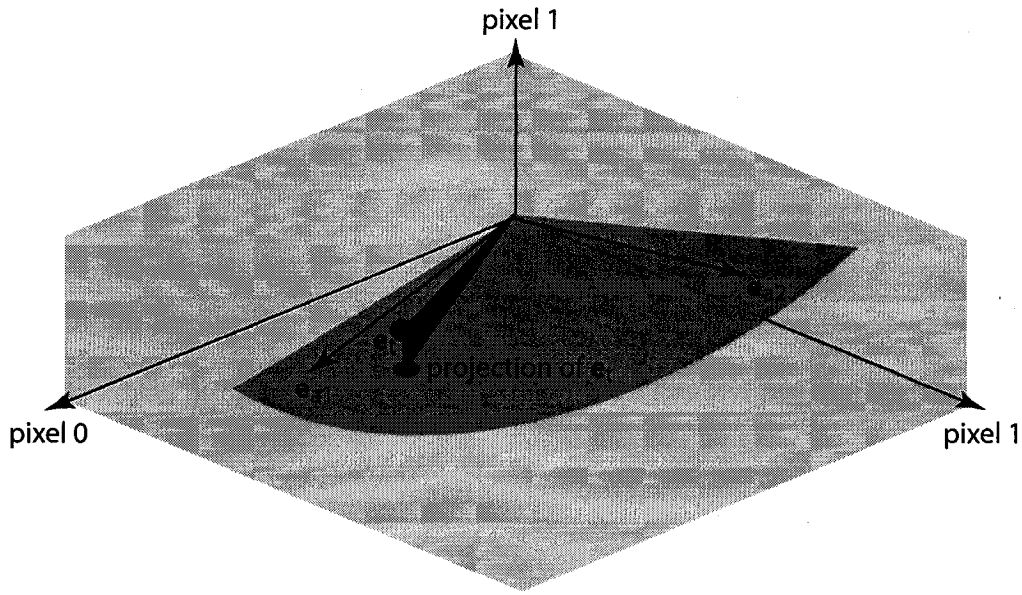


Figure 5.2: Graphical representation when $N_v = 3$, and there are only two local set encoding vectors, $\mathbf{e}_{a1}$ and $\mathbf{e}_{a2}$. The target encoding vector, $\mathbf{e}_t$ is not contained in the subspace spanned by $\mathbf{e}_{a1}$ and $\mathbf{e}_{a2}$, although the difference between $\mathbf{e}_t$ and the projection of $\mathbf{e}_t$ onto the subspace spanned by the local set encoding vectors is small. The optimal linear combination of $\mathbf{e}_{a1}$ and $\mathbf{e}_{a2}$ estimates $\mathbf{e}_t$ by synthesizing the projection of $\mathbf{e}_t$.

weights that minimize $\epsilon_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$.

While the fact that $\widehat{\mathbf{e}}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k})$ is restricted to the subspace spanned by the local set encoding vectors has the disadvantage that some estimation error is almost certainly unavoidable, it greatly simplifies the dimensionality of the problem. Since the best one can achieve is to fit to the projection of $\mathbf{e}_j(\mathbf{k})$ onto the subspace spanned by the local set encoding vectors, the only information one needs to know about $\mathbf{e}_j(\mathbf{k})$ is its projection onto that subspace. In fact the entire problem becomes a problem in the subspace spanned by the local set encoding vectors. PARS maintains the problem in the large N_v-dimensional vector space; APPEAR expresses the problem in the

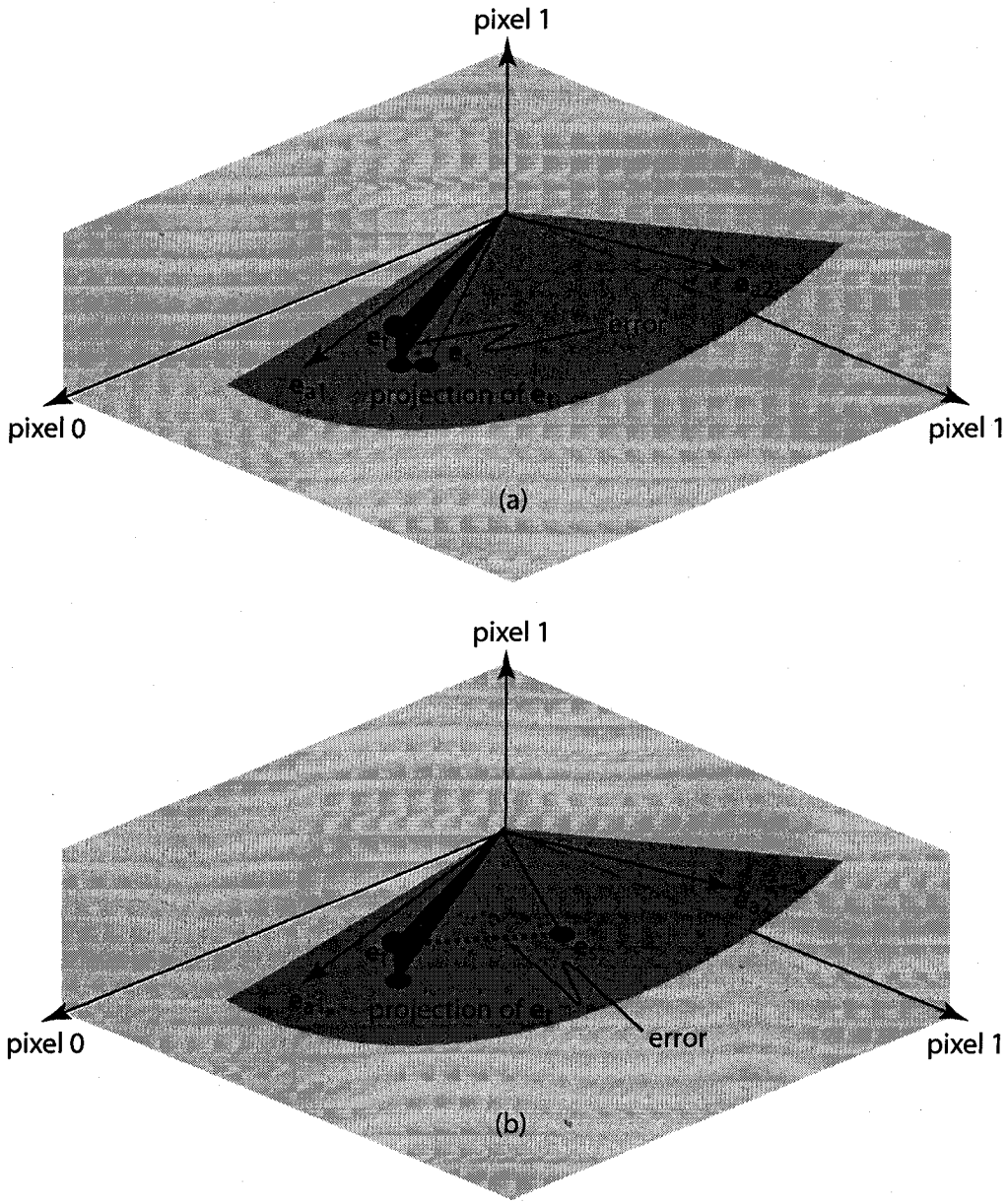


Figure 5.3: The choice of linear combination weights can have a large effect on the error in the estimated encoding vector. The synthesized encoding vector $\mathbf{e}_s$ is a linear combination of $\mathbf{e}_{a1}$ and $\mathbf{e}_{a2}$. In (a), the linear combination weights are chosen such that $\mathbf{e}_s$ is close to the projection of $\mathbf{e}_t$, leading to a small error (dotted line). In (b), the linear combination weights are chosen such that $\mathbf{e}_s$ is not close to the projection of $\mathbf{e}_t$, leading to a much larger error vector (dotted line).

smaller $N_{\mathbf{k}'}$ subspace. However, expressing the problem in terms of correlation values corresponds to working directly in the local set subspace.

Vectors in voxel space are defined by their values along the N_v orthogonal axes. In the local set subspace, no coordinate system is defined. Instead, each vector, including the local set encoding vectors, is defined by its correlation to the local set encoding vectors. Because all of the vectors are defined in a relative way, the absolute position of the local set subspace within the entirety of voxel space is lost. Geometrically, the subspace corresponding to the pattern set $\mathcal{P}(\mathbf{k}, \mathbf{k}')$ is a rotated version of the local set $\mathcal{L}(k)$ subspace. When these two spaces are written in terms of correlation values, the absolute position information is lost and the two subspaces become indistinguishable.

The correlation matrix $\mathbf{C}_{\mathrm{aa}}$ describes the local set subspace. The eigenvectors of $\mathbf{C}_{\mathrm{aa}}$ provide an orthonormal basis for the subspace and the eigenvalues indicate the quality of measurement in each dimension. Analyzing $\mathbf{C}_{\mathrm{aa}}$ by eigenvalues provides insight into the appropriateness of regularization. For small eigenvalues, large linear combination weights are required to synthesize a component along the corresponding eigenvector direction. As noted in the previous chapter, large linear combination weight values amplify noise in the data and can lead to images with a poor signal-to-noise ratio (SNR). To maintain high SNR, one approach is to disregard the components along the eigenvector directions with small eigenvalues. The problem with this approach is that it requires an explicit calculation of the eigenvectors and values of $\mathbf{C}_{\mathrm{aa}}$, which is time consuming. Tychonov regularization achieves much the same effect in a computationally efficient manner. Other regularization approaches are described by Qu *et al.* [42].

A simple 2-D example helps to give a graphical picture of the effect of regularization. Suppose two data samples are acquired at the same k-space location and one datum value is required at this same location for reconstruction. In such a situation, the two data samples could be averaged to reduce the noise in the sample and increase the signal-to-noise ratio by a factor of $\sqrt{2}$. Now suppose that the two data samples are acquired very close to each other in k-space and for reconstruction a data value is required very close to the two acquired data samples. Figure 5.4(a) shows such a situation, where the three points are each separated by 0.01/ FOV and

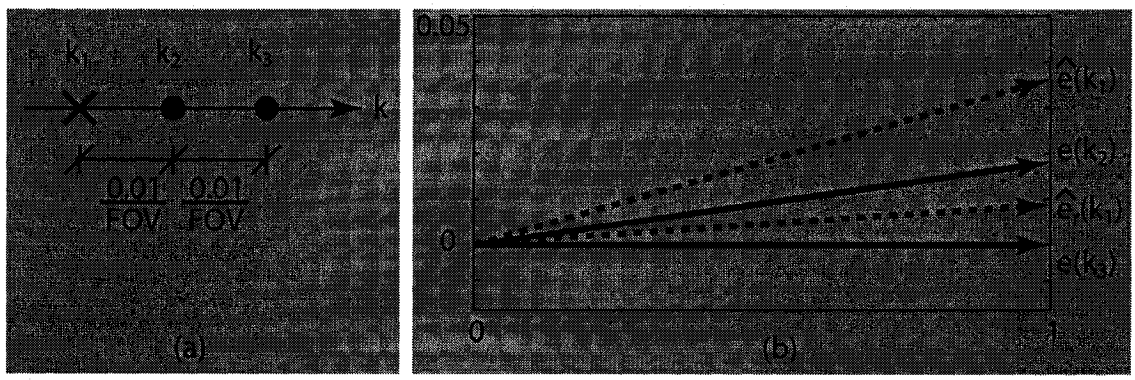


Figure 5.4: (a) Data is acquired at $\mathbf{k}_2$ and $\mathbf{k}_3$ (black circles). A datum needs to be synthesized at $\mathbf{k}_1$ (black x). All three points are very close in k-space. (b) Local set subspace view. The local subspace is the space spanned by $\mathbf{e}(\mathbf{k}_2)$ and $\mathbf{e}(\mathbf{k}_3)$. The projection of $\mathbf{e}(\mathbf{k}_1)$ onto the local set subspace is $\widehat{\mathbf{e}}(\mathbf{k}_1)$. Synthesizing $\widehat{\mathbf{e}}(\mathbf{k}_1)$ from $\mathbf{e}(\mathbf{k}_2)$ and $\mathbf{e}(\mathbf{k}_3)$ requires large linear combination weights. The regularized solution synthesizes $\widehat{\mathbf{e}}_r(\mathbf{k}_1)$ using much smaller linear combination weights. Note that the plot is scaled in the ordinate direction.

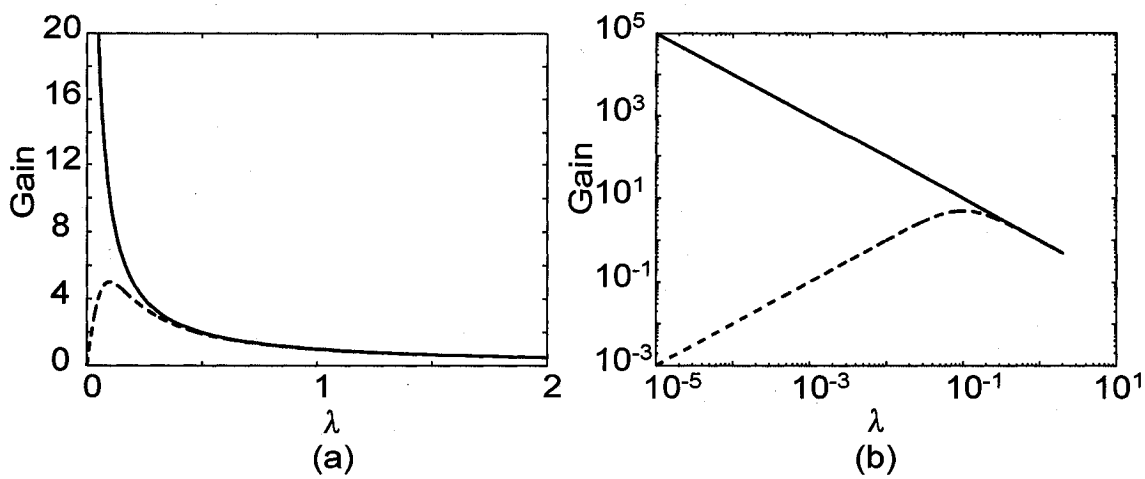


Figure 5.5: Both (a) and (b) plot the gain applied to $\mathbf{c}_{as}$ in an eigenvector direction with eigenvalue λ, when computing the linear combination weights. (a) Linear axes scaling. (b) Logarithmic axes scaling. The solid line is the gain without regularization $(1/\lambda)$. The dashed line is the gain with Tychonov regularization, where $\lambda_{ave} = 1$ and δ is found according to Eq. 5.41.

a datum value at k_1 is to be synthesized from acquired data values at k_2 and k_3. Since all three points are so close to being at the same location, intuition leads to a reasonable solution: average the two data values acquired at k_2 and k_3. Applying gridding to re-sample onto k_1 would also approximate an averaging solution, taking into account density compensation. However, without regularization, local projection interpolation produces a different solution which is much more sensitive to noise in the acquired data values. To show this, we use the expression for correlation values with 1-D gradient encoding in Eq. 5.23 and form $\mathbf{C}_{aa}$ and $\mathbf{c}_{as}$:

$$\mathbf{C}_{aa} = \begin{bmatrix} \text{sinc}(0) & \text{sinc}(0.01) \\ \text{sinc}(0.01) & \text{sinc}(0) \end{bmatrix} \tag{5.31}$$

$$\mathbf{c}_{as} = \begin{bmatrix} \text{sinc}(0.01) \\ \text{sinc}(0.02) \end{bmatrix}. \tag{5.32}$$

The weights can then be calculated as

$$\mathbf{w} = \mathbf{c}_{\text{as}}^{-1}\mathbf{c}_{\text{as}} \tag{5.33}$$

$$= \begin{bmatrix} 1.9995 \\ -0.9999 \end{bmatrix}. \tag{5.34}$$

Rather than increasing the SNR, the SNR of the synthesized datum is reduced by a factor of $\|\mathbf{w}\| = 2.24$. With two members in the local set, the local set subspace spans two dimensions, shown in Fig. 5.4(b). The projection of $\mathbf{e}(\mathbf{k}_1)$ onto the local set subspace is shown in Fig. 5.4(b) as $\widehat{\mathbf{e}}(\mathbf{k}_1)$. Although the local set subspace spans two dimensions, one of those dimensions has a very small eigenvalue. Regularization suppresses the projection along the direction of the small eigenvalue, producing an estimated encoding vector $\widehat{\mathbf{e}}_r(\mathbf{k}_1)$ as shown in Fig. 5.4(b). In doing so, the local set subspace is effectively reduced to one dimension, in the direction of the large eigenvalue and the SNR benefit of two measurements is realized. When $\mathbf{C}_{\text{aa}}$ is better conditioned, regularization has a minimal effect on the solution.

Tychonov regularization computes the weights as

$$\mathbf{w} = \left(\mathbf{C}_{\text{aa}}^H\mathbf{C}_{\text{aa}} + \delta I\right)^{-1}\mathbf{C}_{\text{aa}}^H\mathbf{c}_{\text{as}} \tag{5.35}$$

where I is the $N_{\mathcal{L}(\mathbf{k})} \times N_{\mathcal{L}(\mathbf{k})}$ identity matrix and δ is a parameter of the regularization. Continuing with our 2-D example, we start by analyzing $\mathbf{C}_{\text{aa}}$ in Eq. 5.32 in terms of eigenvectors and eigenvalues:

$$\mathbf{C}_{\text{aa}} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{5.36}$$

where the two eigenvalues are $\lambda_1 = 1+\text{sinc}(0.01) = 1.99984$ and $\lambda_2 = 1-\text{sinc}(0.01) = 0.00016$. The inverse of $\mathbf{C}_{\text{aa}}$ is found by taking the inverse of the eigenvalues:

$$\mathbf{C}_{\text{aa}}^{-1} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} \frac{1}{\lambda_1} & 0 \\ 0 & \frac{1}{\lambda_2} \end{bmatrix}\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}. \tag{5.37}$$

Since $\lambda_2 = 0.00016$ is very small, $1/\lambda_2 = 6080$ is very large and can lead to large weight values. Using Tychonov regularization, $\mathbf{C}_{aa}^{-}1$ is replaced with

$$\left(\mathbf{C}_{aa}^{H}\mathbf{C}_{aa} + \delta I\right)^{-1}\mathbf{C}_{aa}^{H} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} \frac{\lambda_1}{\lambda_1^2+\delta} & 0 \\ 0 & \frac{\lambda_2}{\lambda_2^2+\delta} \end{bmatrix}\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}. \tag{5.38}$$

When an eigenvalue $\lambda \gg \sqrt{\delta}$,

$$\frac{\lambda}{\lambda^2+\delta} \approx \frac{1}{\lambda}. \tag{5.39}$$

When an eigenvalue $\lambda \ll \sqrt{\delta}$

$$\frac{\lambda}{\lambda^2+\delta} \approx \frac{\lambda}{\delta} \approx 0. \tag{5.40}$$

Thus, δ should be chosen to be significantly larger than the square of small eigenvalues and significantly smaller than the square of large eigenvalues. While the aforementioned principle for choosing δ is true, it is very vague and does not give a useful expression for δ. In practice, I have found that

$$\delta = 0.01\lambda_{ave}^2 \tag{5.41}$$

works well, where λ_{ave} is the average eigenvalue of $\mathbf{C}_{aa}$ and can be quickly calculated as

$$\lambda_{ave} = \frac{\text{trace}(\mathbf{C}_{aa})}{N_{\mathcal{L}(\mathbf{k})}}. \tag{5.42}$$

Using Eq. 5.41 in the 2-D example, $\delta = 0.01$ and

$$\mathbf{w} = \begin{bmatrix} 0.499 \\ 0.499 \end{bmatrix}, \tag{5.43}$$

which is very close to the intuitive averaging solution, achieving a $\sqrt{2}$ SNR improvement. Figure 5.4(b) shows the estimated encoding vector found by regularization, $\widehat{\mathbf{e}}_r(\mathbf{k}_1)$. For our 2-D example, Fig. 5.5 compares the gain of the non-regularized and regularized solutions as a function of the eigenvalue, showing that the regularized

solution suppresses the gain from small eigenvalues.

While the correlation matrix $\mathbf{C}_{aa}$ describes the local set subspace, the correlation vector $\mathbf{c}_{as}$ tells one what the projection of the synthesis encoding vector $\mathbf{e}_j(\mathbf{k})$ is onto the local set subspace. When projecting $\mathbf{e}_j(\mathbf{k})$ onto the local set subspace, information about $\mathbf{e}_j(\mathbf{k})$ is lost. For example, assuming $N_v = 256 \times 256 = 65,536$ and $N_{\mathcal{L}(\mathbf{k})} = 100$, there are $65,536 - 100 = 65,436$ dimensions in voxel space that are orthogonal to the local set subspace. The value of all components of $\mathbf{e}_j(\mathbf{k})$ along these 65,436 dimensions are lost when $\mathbf{e}_j(\mathbf{k})$ is projected onto the local set subspace. Calibration methods that estimate the receiver-coil sensitivity functions attempt to estimate all N_v dimensions of $\mathbf{e}_j(\mathbf{k})$. When a local kernel interpolation approach is taken, most of the information contained in the receiver-coil sensitivity function is never used since it is orthogonal to the local set subspace. In contrast to the sensitivity function estimation approach, local projection calibration used by APPEAR focuses on accurately estimating the projection of $\mathbf{e}_j(\mathbf{k})$ onto the local set subspace (or identically the projection of the receiver-coil sensitivity function onto the pattern set $\mathcal{P}(\mathbf{k}, 0)$ subspace). Local projection calibration is not concerned with estimating any component of $\mathbf{e}_j(\mathbf{k})$ orthogonal to the local set subspace. While APPEAR is able to accurately estimate the projection of the receiver-coil sensitivity function onto the pattern set $\mathcal{P}(\mathbf{k}, 0)$ subspace, it is difficult to obtain an estimate of the receiver-coil sensitivity functions from this estimate, both because of the loss of information by projection and because the projection result is only known in the pattern set subspace. While local projection calibration can estimate correlation values to put the problem into the local set subspace, recall that the absolute relation of this subspace to the entire voxel space is not known.

5.6 Gridding and Local Projection Interpolation

Both gridding and local projection interpolation can be used to synthesize k-space data falling on a Cartesian grid from non-Cartesian acquired k-space data. Moreover, both methods synthesize data from acquired data within a local k-space neighborhood. However, the two methods have some important differences, which I outline

in this section. The purpose of this section is not to show that one method is superior to the other, but rather to show the strengths and weaknesses of both methods, allowing the reader to make a more informed choice between methods for a specific application.

One property of gridding is that it is *data driven*, convolving the acquired data with a gridding kernel and sampling the result onto a Cartesian grid. In contrast, local projection interpolation is *grid driven*, finding the acquired data local to a grid point and using it to synthesize a datum at the grid point. The difference between being data or grid driven is more a matter of implementation since, once all of the linear combination weights for local projection interpolation are generated, they can be reorganized in the computer memory to allow for a data driven implementation. However, the "gridding kernel" that is generated from local projection interpolation linear combination weights is not necessarily spatially invariant like the gridding kernels discussed in chapter 3. While a data driven approach is attractive for implementation, allowing the data to be processed as it is acquired from the scanner, it is more illuminating to compare gridding and local projection interpolation using a matrix formulation.

When the image is encoded solely with gradient encoding, the acquired data can be expressed with a modified version of Eq. 4.6:

$$\mathbf{d}_{\mathcal{A}} = \mathbf{G}_{\mathcal{A}}\mathbf{m}$$

where $\mathbf{G}_{\mathcal{A}}$ is the gradient encoding matrix for the acquisition set $\mathcal{A}$ and $\mathbf{d}_{\mathcal{A}}$ is the acquired data for the single coil. Unlike the multiple coil case, where the entries of the encoding matrix $\mathbf{E}_{\mathcal{A}}$ are not known, $\mathbf{G}_{\mathcal{A}}$ is completely known. From a matrix perspective then, the reconstructed image $\widehat{\mathbf{m}}$ can be computed as

$$\widehat{\mathbf{m}} = \mathbf{G}_{\mathcal{A}}^{\dagger}\mathbf{d}_{\mathcal{A}}.$$

The reason such a computation is not often attempted is that the matrix $\mathbf{G}_{\mathcal{A}}^{\dagger}$ is very large, making such an approach very computationally intensive. Gridding computes

$\widehat{\mathbf{m}}$ as

$$\widehat{\mathbf{m}} = \mathbf{G}_{\mathcal{A}}^{H}\mathbf{W}\mathbf{d}_{\mathcal{A}} \tag{5.44}$$

where $\mathbf{W}$ is a diagonal matrix that contains the density compensation weights. Note that $\mathbf{G}_{\mathcal{A}}^{H}$, the Hermitian conjugate of $\mathbf{G}_{\mathcal{A}}$, is the inverse Fourier transform that gridding computes quickly using convolution interpolation and the Fast Fourier Transform. Gridding works well when

$$\mathbf{G}_{\mathcal{A}}^{H}\mathbf{W} \approx \mathbf{G}_{\mathcal{A}}^{\dagger} \tag{5.45}$$

and Sedarat and Nishimura [43] show that Eq. 5.45 can be used to find optimal density compensation weights. How well $\mathbf{G}_{\mathcal{A}}^{H}\mathbf{W}$ approximates $\mathbf{G}_{\mathcal{A}}^{\dagger}$ depends on the sampling trajectory, $S(k)$. For a DFT trajectory, which samples a Cartesian grid, $\mathbf{G}_{\mathcal{A}}^{H} = \mathbf{G}_{\mathcal{A}}^{\dagger}$. For radial and spiral trajectories, suitable density compensation can make $\mathbf{G}_{\mathcal{A}}^{H}\mathbf{W}$ a good approximation for $\mathbf{G}_{\mathcal{A}}^{\dagger}$.

The gridding approximation for $\mathbf{G}_{\mathcal{A}}^{\dagger}$ can break down when uneven sampling is used. Consider the sampling function

$$S(\mathbf{k}) = \text{Ш}\left(N_x k_x\right)\left[\frac{1}{2}\text{Ш}\left(\frac{2}{3}N_y k_y - \frac{1}{3}\right) + \frac{1}{2}\text{Ш}\left(\frac{2}{3}N_y k_y + \frac{1}{3}\right)\right]. \tag{5.46}$$

The magnetization is assumed to be in the $N_x \times N_y$ FOV. This sampling function samples evenly at the Nyquist rate in the k_x-direction and unevenly in the k_y-direction. In the k_y-direction, the sampling alternates between the Nyquist rate and twice the Nyquist rate, as shown in Fig. 5.6(a). Since the Fourier transform is separable in k_x and k_y, we can consider only the k_y-dimension without loss of generality:

$$S(k_y) = \frac{1}{2}\text{Ш}\left(\frac{2}{3}N_y k_y - \frac{1}{3}\right) + \frac{1}{2}\text{Ш}\left(\frac{2}{3}N_y k_y + \frac{1}{3}\right). \tag{5.47}$$

For this sampling function, it is optimal for all of the density compensation weights to be equal, as can be seen by the symmetry of the sampling function. With this density compensation, the point-spread function psf(y) is the inverse Fourier transform of

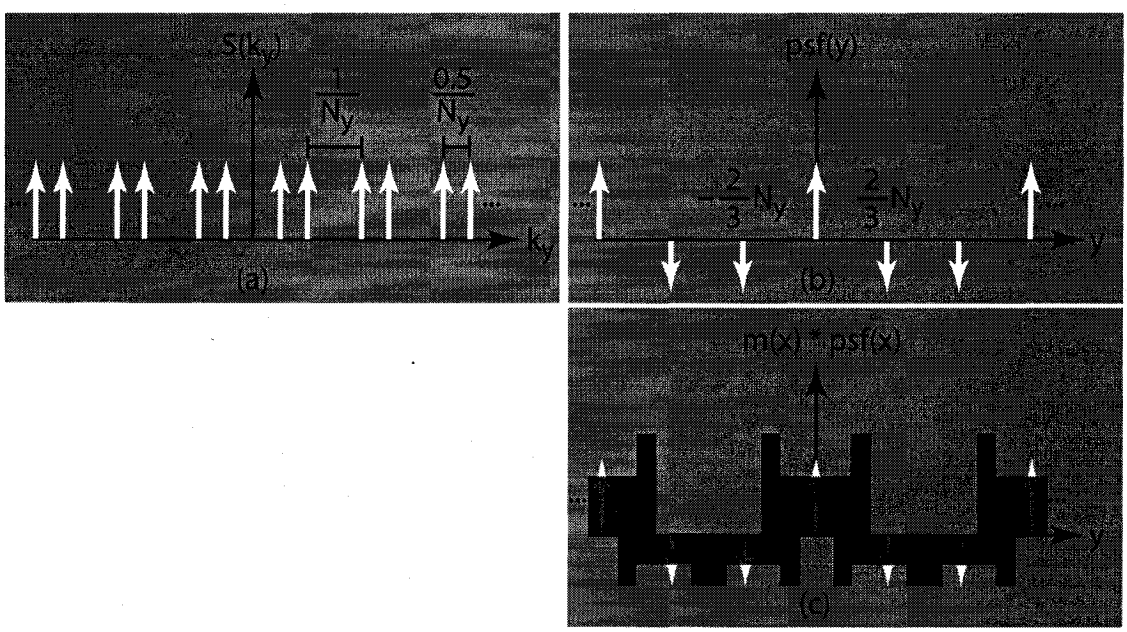


Figure 5.6: (a) The sampling function $S(k_y)$ (Eq. 5.47) samples k-space unevenly. Note that the samples in $S(k_y)$ are never separated by more than $1/\mathrm{N_y}$, never sampling less than the Nyquist rate. (b) The point-spread function $\mathrm{psf}(y)$ which is the Fourier transform of the sampling function. (c) The point-spread function convolved with magnetization, showing the aliasing artifact produced in the image when gridding is used for reconstruction.

$S(k_y)$:

$$\begin{aligned}
\text{psf}(y) &= \frac{3}{4N_y}\text{Ш}\left(\frac{3}{2N_y}y\right)e^{-i\pi y/N_y} + \frac{3}{4N_y}\text{Ш}\left(\frac{3}{2N_y}y\right)e^{i\pi y/N_y} && (5.48)\\
&= \frac{3}{2N_y}\text{Ш}\left(\frac{3}{2N_y}y\right)\cos\left(\frac{\pi}{N_y}y\right) && (5.49)\\
&= \sum_{n=-\infty}^{\infty}\cos\left(2\pi\frac{n}{3}\right)\delta\left(y - 2N_y\frac{n}{3}\right), && (5.50)
\end{aligned}$$

which is plotted in Fig. 5.6(b). Recall that the final image for the gridding method is

$$\widehat{m}(y) = m(y) * \text{psf}(y). \tag{5.51}$$

There are three delta functions in $\text{psf}(y)$ that contribute to image content within the FOV. Since we are only concerned with the image reconstruction within the FOV, we can write

$$\begin{aligned}
\widehat{m}(y) &= m(y) * \left[\delta(y) + \cos\left(-\frac{2\pi}{3}\right)\delta\left(y + \frac{2N_y}{3}\right) + \cos\left(\frac{2\pi}{3}\right)\delta\left(y - \frac{2N_y}{3}\right)\right]\\
&= m(y) - \frac{1}{2}m\left(y + \frac{2N_y}{3}\right) - \frac{1}{2}m\left(y - \frac{2N_y}{3}\right). && (5.52)
\end{aligned}$$

Thus gridding corrupts the reconstruction with two scaled copies of the magnetization aliasing into the reconstructed image, shown in Fig. 5.6(c). This poor quality reconstruction is a limitation of gridding and is a particular case of the approximation in Eq. 5.45 being a poor approximation.

I turn now to the local projection interpolation approach to solving this problem, clearly showing how this approach can attain qualitatively different results from gridding. For this example, I take the approach that we wish to synthesize unacquired data at the locations:

$$\text{Ш}\left(\frac{2}{3}N_y k_y\right) \tag{5.53}$$

to create an evenly spaced data set. Figure 5.7 shows one of these locations being synthesized from a local neighborhood of eight points. The linear combination weights have been calculated using Eq. 5.35 with the correlation values found using Eq. 5.23.

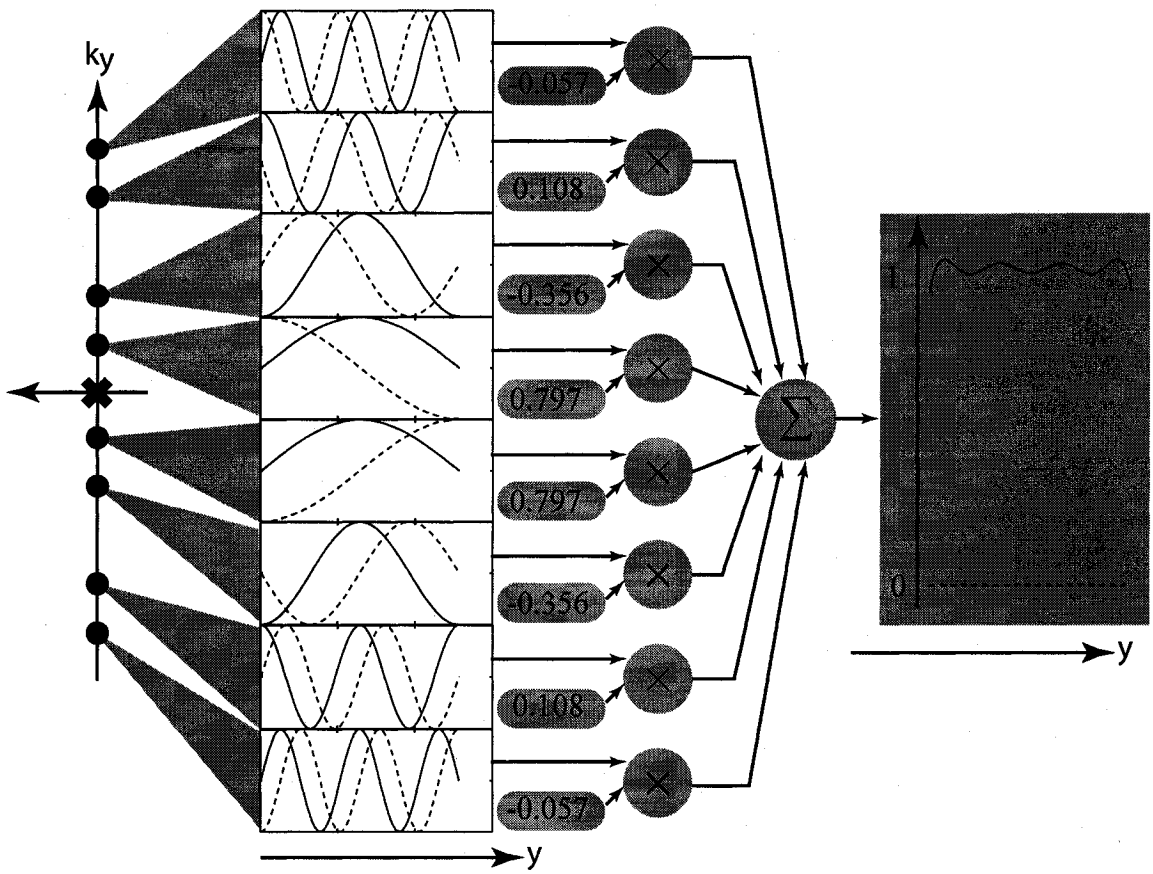


Figure 5.7: Local projection interpolation of $g(\mathbf{k}, \mathbf{r})(0, \mathbf{r}) = e^{-i2\pi 0 \cdot \mathbf{r}} = 1$ from eight local neighbors in $S(k_y)$ (Eq. 5.47).

Figure 5.7 illustrates how each of the gradient encoding functions for the local neighborhood are multiplied by a linear combination weight and the results summed to approximate the gradient encoding function for the unacquired location, in this case unity over the field of view. Figure 5.8 shows that the results with this approach are markedly better than gridding this case. Whereas gridding has the expected aliasing artifacts, local projection calibration can be used to reconstruct the image without the aliasing artifacts seen in the gridding reconstruction.

For trajectories where Eq. 5.44 is a good approximation, gridding is a very efficient reconstruction method that does not require any matrix inverse calculations.

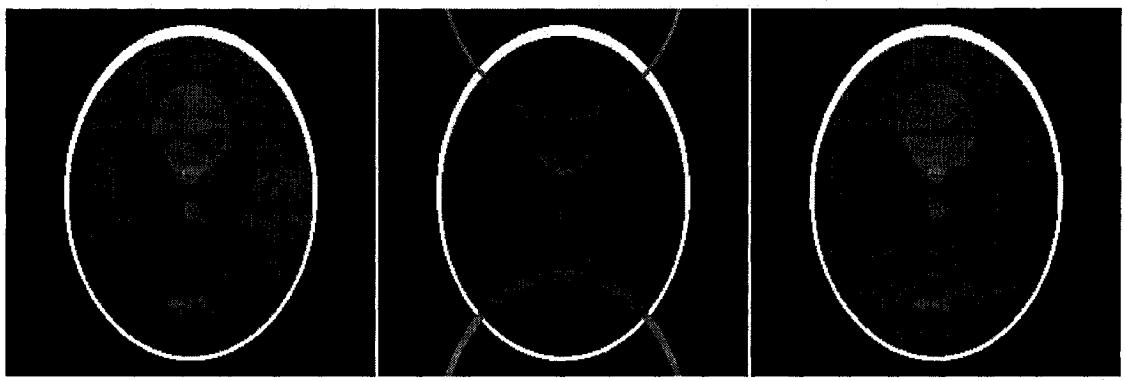


Figure 5.8: (a) Shepp-Logan numerical phantom. (b) Gridding reconstruction for sampling trajectory $S(k_y)$ in Eq. 5.47. (c) Local projection interpolation reconstruction for sampling trajectory in Eq. 5.47. Local projection interpolation reconstruction is able to suppress the aliasing artifacts visible in the gridding reconstruction.

However, when the approximation in Eq. 5.44 is poor, local projection interpolation can be used to reconstruct the image with less aliasing artifact, albeit at the expense of a more computationally intensive method.

Up until this point, I have dealt with correlation values in situations where the encoding functions are known, allowing the correlation values to be explicitly calculated. As highlighted in the previous chapter, the challenge with reconstructing images encoded with receiver-coil sensitivity functions arises from the fact that the encoding functions are not fully known. In the next section, I look at how the correlation values can be approximated in this situation.

5.7 Approximating Correlation Values

In this section, local projection calibration is put in the form of approximating correlation values. By putting local projection calibration in the form of approximating correlation values, a graphical picture can be developed which is useful in understanding local projection calibration and critical to the following section which derives a

fast method for computing the local projection calibration correlation value approximations.

Equation 4.34 shows how APPEAR generates linear combination weights from synthesized data in the central calibration region using local projection calibration. Just as Eq. 5.13 expresses Eq. 5.8 in terms of correlation values, Eq. 4.34 can be expressed as

$$\mathbf{w}_{j,\mathcal{L}(\mathbf{k})}(\mathbf{k}) = \begin{bmatrix} \widehat{c}(1;1) & \dots & \widehat{c}(1;N_{\mathcal{L}(\mathbf{k})}) \\ \vdots & \ddots & \vdots \\ \widehat{c}(N_{\mathcal{L}(\mathbf{k})};1) & \dots & \widehat{c}(N_{\mathcal{L}(\mathbf{k})};N_{\mathcal{L}(\mathbf{k})}) \end{bmatrix}^{-1} \begin{bmatrix} \widehat{c}(1;j,\mathbf{0}) \\ \vdots \\ \widehat{c}(N_{\mathcal{L}(\mathbf{k})};j,\mathbf{0}) \end{bmatrix}, \quad (5.54)$$

where $\widehat{c}(l_1;l_2)$ is the correlation value approximation made by local projection calibration, relating encoding location l_1 to encoding location l_2, where both l_1 and l_2 are in the pattern set $\mathcal{P}(\mathbf{k},\mathbf{0})$. I have chosen to base the encoding locations on $\mathcal{P}(\mathbf{k},\mathbf{0})$ instead of $\mathcal{L}(\mathbf{k})$ as this simplifies the analysis while being mathematically identical.

Each of the correlation value approximations in Eq. 5.54 can be expressed as

$$\begin{aligned}
\widehat{c}(j_1,\mathbf{k}_1;j_2,\mathbf{k}_2) &= \sum_{\mathbf{k}'=\mathbf{k}'_1}^{\mathbf{k}'_{N_{\mathbf{k}'}}} d^*_{j_1}(\mathbf{k}'+\mathbf{k}_1)d_{j_2}(\mathbf{k}'+\mathbf{k}_2) && (5.55)\\
&= \int S_{\text{fit}}(\mathbf{k}') \left[\int m_{j_1}(\mathbf{r}_1)e^{-i2\pi(\mathbf{k}'+\mathbf{k}_1)\cdot\mathbf{r}_1}d\mathbf{r}_1\right]^* \\
&\quad \left[\int m_{j_2}(\mathbf{r}_2)e^{-i2\pi(\mathbf{k}'+\mathbf{k}_2)\cdot\mathbf{r}_2}d\mathbf{r}_2\right] d\mathbf{k}' && (5.56)\\
&= \int\int m^*_{j_1}(\mathbf{r}_1)m_{j_2}(\mathbf{r}_2) \\
&\quad \int S_{\text{fit}}(\mathbf{k}')e^{i2\pi(\mathbf{k}'+\mathbf{k}_1)\cdot\mathbf{r}_1}e^{-i2\pi(\mathbf{k}'+\mathbf{k}_2)\cdot\mathbf{r}_2}d\mathbf{k}'d\mathbf{r}_1d\mathbf{r}_2 && (5.57)\\
&= \int\int m^*_{j_1}(\mathbf{r}_1)m_{j_2}(\mathbf{r}_2)e^{i2\pi\mathbf{k}_1\cdot\mathbf{r}_1}e^{-i2\pi\mathbf{k}_2\cdot\mathbf{r}_2} \\
&\quad \int S_{\text{fit}}(\mathbf{k}')e^{-i2\pi\mathbf{k}'\cdot(\mathbf{r}_2-\mathbf{r}_1)}d\mathbf{k}'d\mathbf{r}_1d\mathbf{r}_2 && (5.58)\\
&= \int\int m^*_{j_1}(\mathbf{r}_1)m_{j_2}(\mathbf{r}_2)\delta_{\text{fit}}(\mathbf{r}_2-\mathbf{r}_1) \\
&\quad e^{i2\pi\mathbf{k}_1\cdot\mathbf{r}_1}e^{-i2\pi\mathbf{k}_2\cdot\mathbf{r}_2}d\mathbf{r}_1d\mathbf{r}_2, && (5.59)
\end{aligned}$$

where $S_{\text{fit}}(\mathbf{k})$ is the sampling function for the fit grid locations $(\mathbf{k}'_1 \ldots \mathbf{k}'_{N_{\mathbf{k}'}})$ and $\delta_{\text{fit}}(\mathbf{r})$ is its Fourier transform:

$$\delta_{\text{fit}}(\mathbf{r}) = \text{FT}\left\{S_{\text{fit}}(\mathbf{k})\right\}(\mathbf{r}). \tag{5.60}$$

Note that I have not included the effect of apodization due to data synthesis in the central calibration region. I have left this out for simplicity. When included, the apodization function $a(\mathbf{r})$ apodizes the correlation weighting function $f(\mathbf{r})$, changing it from $m(\mathbf{r})$ to $m(\mathbf{r})a(\mathbf{r})$. Equation 5.55 expresses a straightforward calculation of the correlation value approximation. Equation 5.59 puts the correlation value approximation in a form that allows for comparison with the definition of the correlation value. To aid in this comparison, Eq. 5.3 can be expanded as

$$\begin{aligned} c(j_1, \mathbf{k}_1; j_2, \mathbf{k}_2) &= \int \widetilde{e}^{\,*}_{j_1}(\mathbf{k}_1, \mathbf{r})\, \widetilde{e}_{j_2}(\mathbf{k}_2, \mathbf{r})\; d\mathbf{r} && (5.61) \\ &= \int\int m^*_{j_1}(\mathbf{r}_1) m_{j_2}(\mathbf{r}_2) \delta(\mathbf{r}_2 - \mathbf{r}_1) \\ &\quad e^{i2\pi\mathbf{k}_1\cdot\mathbf{r}_1} e^{-i2\pi\mathbf{k}_2\cdot\mathbf{r}_2}\; d\mathbf{r}_1 d\mathbf{r}_2. && (5.62) \end{aligned}$$

Local projection calibration approximates the $\delta(\mathbf{r}_2 - \mathbf{r}_1)$ function in Eq. 5.62 with $\delta_{\text{fit}}(\mathbf{r}_2 - \mathbf{r}_1)$, which is the Fourier transform of the sampling function for the fit examples in the fit region. Note that when the fit region is expanded to cover all of k-space, $\delta_{\text{fit}}(\mathbf{r}_2 - \mathbf{r}_1) \rightarrow \delta(\mathbf{r}_2 - \mathbf{r}_1)$.

In the previous chapter, I showed that the approximation made by local projection calibration is different from the approximation made by PARS, which uses low-resolution *in vivo* coil sensitivities. The correlation value approximation used by PARS can be written as

$$\begin{aligned} \widehat{c}(j_1, \mathbf{k}_1; j_2, \mathbf{k}_2) &= \int \left[m_{j_1}(\mathbf{r}) * \text{psf}_{\text{cal}}(\mathbf{r})\right]^* e^{i2\pi\mathbf{k}_1\cdot\mathbf{r}} \\ &\quad \left[m_{j_2}(\mathbf{r}) * \text{psf}_{\text{cal}}(\mathbf{r})\right] e^{-i2\pi\mathbf{k}_2\cdot\mathbf{r}} d\mathbf{r} && (5.63) \\ &= \int \left[m_{j_1}(\mathbf{r}_1) * \text{psf}_{\text{cal}}(\mathbf{r}_1)\right]^* \left[m_{j_2}(\mathbf{r}_2) * \text{psf}_{\text{cal}}(\mathbf{r}_2)\right] \\ &\quad e^{i2\pi\mathbf{k}_1\cdot\mathbf{r}_1} e^{-i2\pi\mathbf{k}_2\cdot\mathbf{r}_2} \delta(\mathbf{r}_2 - \mathbf{r}_1) d\mathbf{r}_1 d\mathbf{r}_2. && (5.64) \end{aligned}$$

Unlike local projection calibration which approximates the delta function, when calibration is performed using low resolution *in vivo* coil sensitivities, the magnetization weighted coil sensitivities are approximated with low resolution versions of the magnetization weighted coil sensitivities.

For both the correlation value definition and the PARS approximation, the correlation value relating encoding locations $(j_1, \mathbf{k}_1)$ and $(j_2, \mathbf{k}_2)$ is a function of j_1 and j_2 and the separation in k-space $\mathbf{k}_2 - \mathbf{k}_1$. That is,

$$c\left(j_1, \mathbf{k}_1 + \Delta\mathbf{k}; j_2, \mathbf{k}_2 + \Delta\mathbf{k}\right) = c\left(j_1, \mathbf{k}_1; j_2, \mathbf{k}_2\right). \tag{5.65}$$

For the local projection calibration correlation value approximation, this shift invariance is not generally true, and points to a difference in approach. Heberlein and Hu show that there is some similarity between Cartesian GRAPPA, a case of local projection interpolation, and the "kriging" method used in geostatistics [44]. However, since the cross-covariance terms used in kriging are shift invariant whereas the correlation value approximations used with Cartesian GRAPPA are not shift invariant, one can deduce that the two methods are not identical. Moreover, the correlation matrix $\mathbf{C}_{\mathbf{aa}}$ generated by local projection calibration is guaranteed to be Hermitian positive definite, whereas the kriging matrix is guaranteed Hermitian but not positive definite.

Equations 5.59, 5.62 and 5.64 all express a correlation value as a 2-D Fourier transform of similar but slightly varied functions. Figure 5.9 illustrates the similarity and variation of these functions graphically. The shift-invariance expressed in Eq. 5.65 corresponds to having non-zero values only along the diagonal where $\mathbf{r}_1 = \mathbf{r}_2$ in Fig. 5.9. While the local projection calibration approximation has non-zero values off of the diagonal, the values diminish quickly as the distance from the diagonal is increased. This property is used in the next section, which develops a method to efficiently calculate the correlation value approximations.

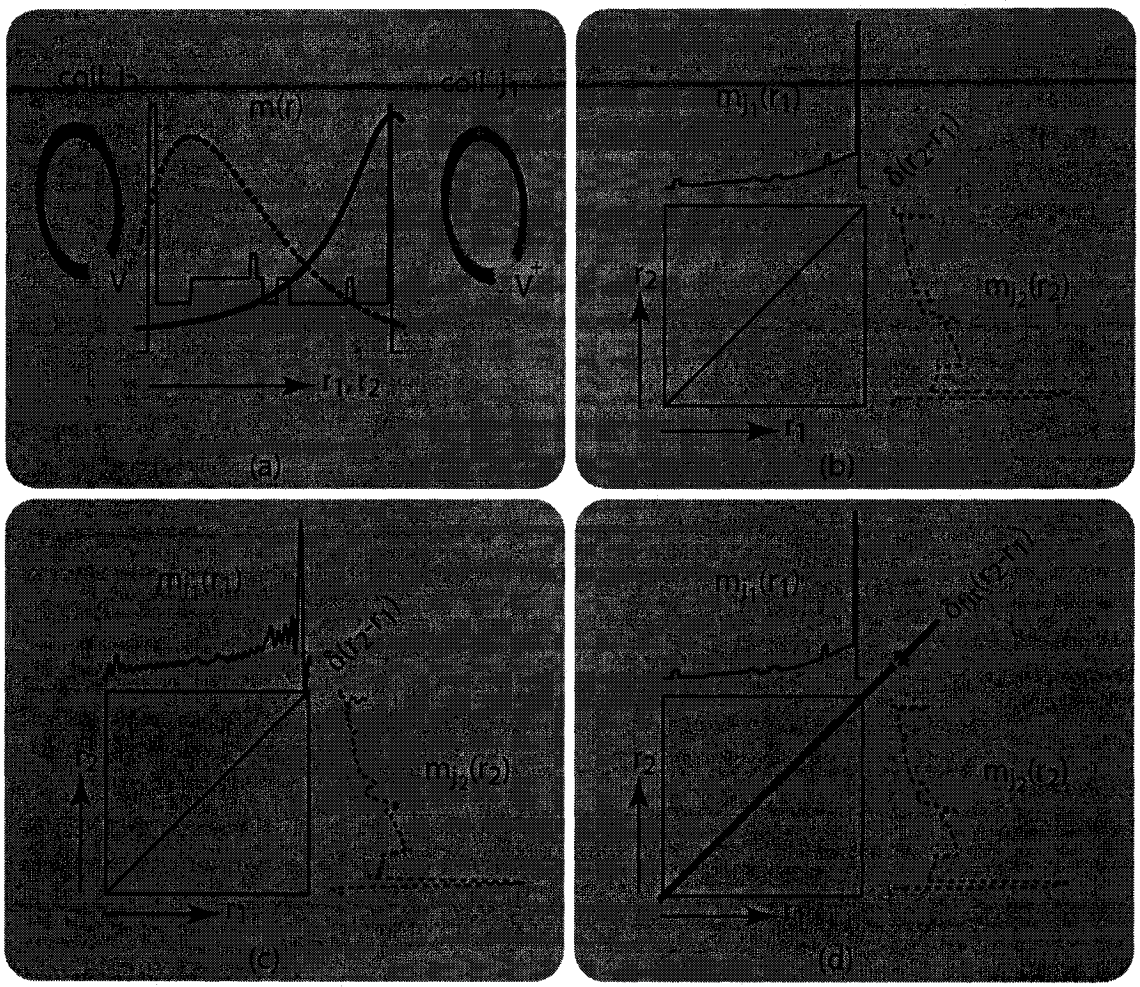


Figure 5.9: This figure contains a graphical comparison between the ideal correlation value (Eq. 5.62) and the correlation value approximation from PARS (Eq. 5.64) and APPEAR (Eq. 5.59). (a) Plot of 1-D magnetization $m(\mathbf{r})$ and coil sensitivity for coil j_1 (solid line) and coil j_2 (dotted line). (b) The 2-D Fourier transform of the correlation values (Eq. 5.62) has three parts: $m_{j_1}(\mathbf{r}_1)$, plotted along $\mathbf{r}_1$, $m_{j_2}(\mathbf{r}_2)$, plotted along $\mathbf{r}_2$ and $\delta(\mathbf{r}_2 - \mathbf{r}_1)$, which sets all values to zero which are not along the diagonal $\mathbf{r}_2 = \mathbf{r}_1$. (c) The PARS approximation replaces $m_{j_1}(\mathbf{r}_1)$ and $m_{j_2}(\mathbf{r}_2)$with low resolution versions of the magnetization-weighted sensitivities. (d) The approximation used by APPEAR replaces the delta function with $\delta_{\text{fit}}(\mathbf{r}_2 - \mathbf{r}_1)$, which is a sinc function when the fit region is a rect. A wider fit region results in a more narrow sinc function, which more effectively suppresses off-diagonal content.

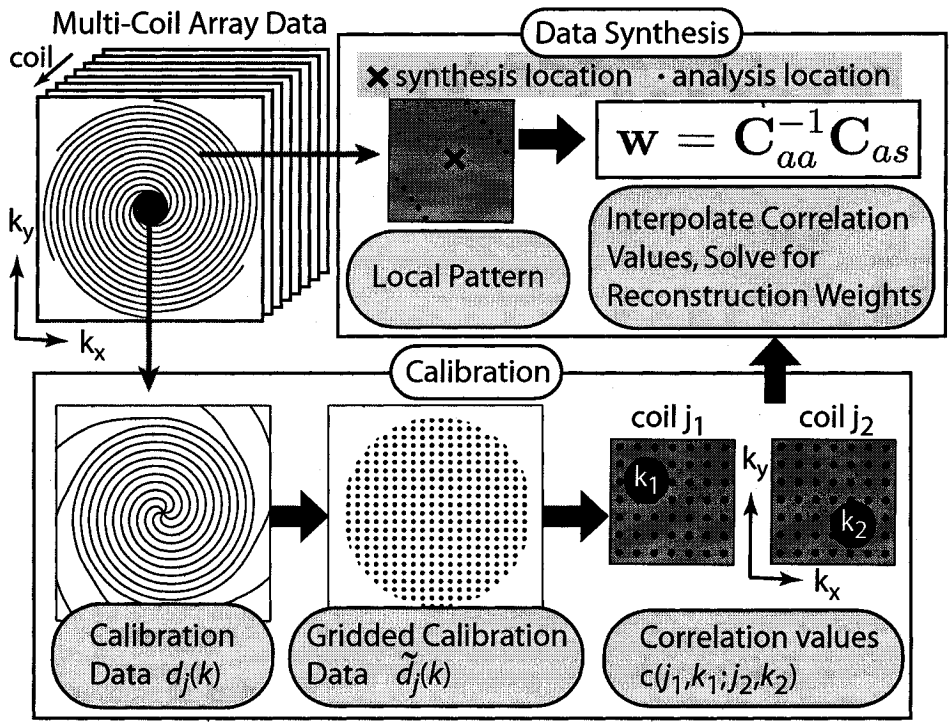


Figure 5.10: The APPEAR procedure modified from Fig. 4.4 to take advantage of more efficient computation using correlation values. For the 2-D spiral trajectories used, this modification reduces the computation time by a factor of ten.

5.8 Computing Approximate Correlation Values

The approach that I take to implement APPEAR with correlation values is to first calculate correlation values relating each coil pair, across a $\mathbf{k}_1$, $\mathbf{k}_2$ grid. Then, the APPEAR method is modified to calculate the linear combination weights from correlation values, where appropriate correlation values are interpolated from the $\mathbf{k}_1$, $\mathbf{k}_2$ grids of correlation values. The modified APPEAR implementation is shown in Fig. 5.10. In this section, I develop a method to quickly calculate the $\mathbf{k}_1$, $\mathbf{k}_2$ correlation value grids and show that any required correlation value can be interpolated from these grids.

To show how I calculate the correlation values for a $\mathbf{k}_1$, $\mathbf{k}_2$ grid, I start by rewriting Eq. 5.56 as

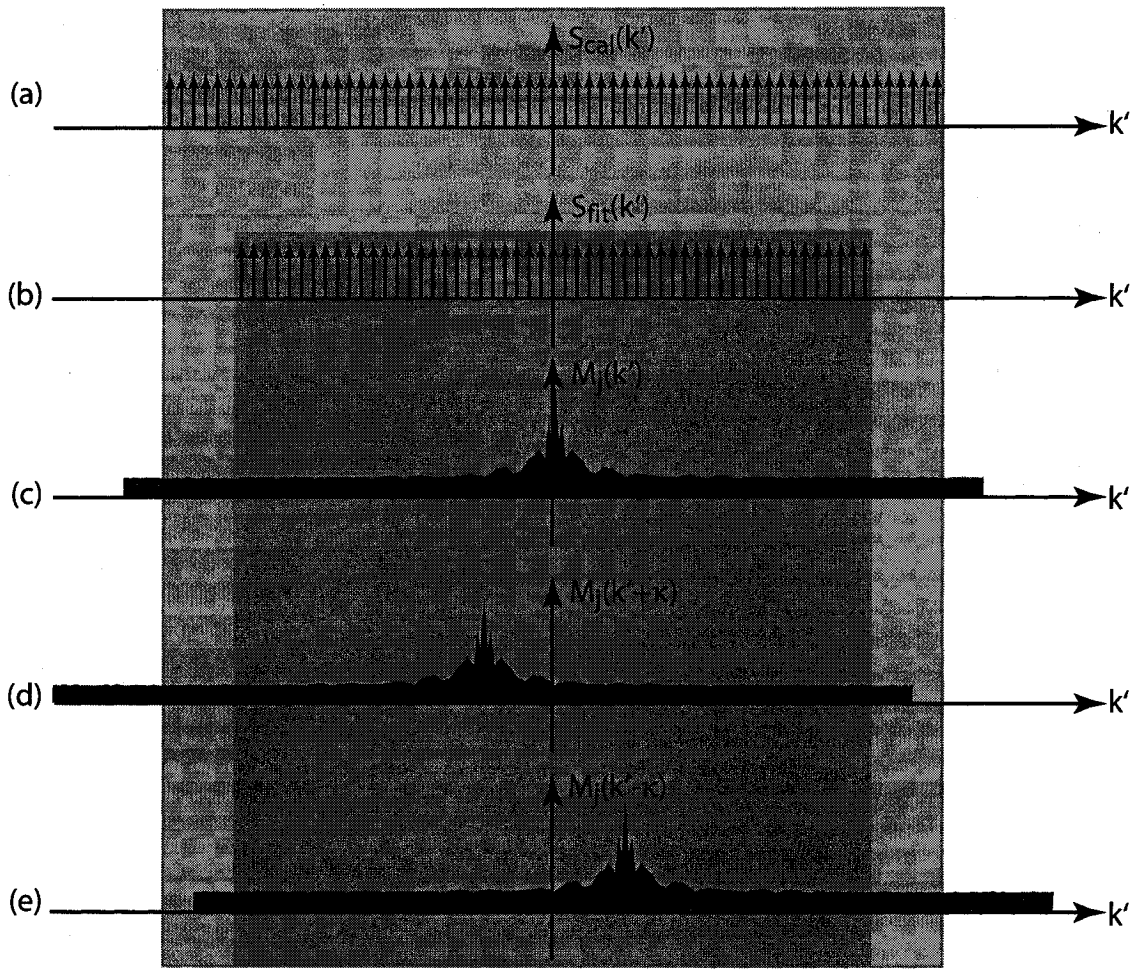


Figure 5.11: 1-D illustration demonstrating that $M^*_{j_1}(\mathbf{k}' + \mathbf{k}_1)$ can be replaced with $M^*_{j_1,\text{cal}}(\mathbf{k}' + \mathbf{k}_1) = \sqcap\left(\frac{\mathbf{k}'+\mathbf{k}_1}{\text{cal}}\right) M^*_{j_1}(\mathbf{k}' + \mathbf{k}_1)$ in the integrand in Eq. 5.66. The value 'cal' specifies the extent of the calibration region in k-space. (a) Sampling function for calibration region, $S_{\text{cal}}(\mathbf{k}')$, spans extent of the calibration region (light gray). (b) The fit region is of smaller extent (dark gray). (c) k-space data $M_j(\mathbf{k}')$. Note that $M_j(\mathbf{k}')$ extends past the calibration region. The portion in the calibration region is shaded in black. (d),(e) $M_j(\mathbf{k}' + \kappa)$ and $M_j(\mathbf{k}' - \kappa)$ are equivalent to $M_j(\mathbf{k}')$ shifted left and right by κ (the radius of the local set). Note that $\|\mathbf{k}_1\| \leq \kappa$ and that only the domain of $M_j(\mathbf{k}')$ within the calibration region enters into the fit region when $M_j(\mathbf{k}')$ is shifted. Because of these properties, $S_{\text{fit}}(\mathbf{k}')M_j(\mathbf{k}'+\mathbf{k}_1) = S_{\text{fit}}(\mathbf{k}')M_{j,\text{cal}}(\mathbf{k}'+\mathbf{k}_1)$ and $M^*_{j_1}(\mathbf{k}' + \mathbf{k}_1)$ can be replaced by $M^*_{j_1,\text{cal}}(\mathbf{k}' + \mathbf{k}_1)$ in the integral in Eq. 5.66 without changing the value of integrand.

$$\widehat{c}(j_1, \mathbf{k}_1; j_2, \mathbf{k}_2) = \int S_{\text{fit}}(\mathbf{k}') M^*_{j_1}(\mathbf{k}' + \mathbf{k}_1) M_{j_2}(\mathbf{k}' + \mathbf{k}_2) d\mathbf{k}', \tag{5.66}$$

where

$$M_j(\mathbf{k}) = \int m_j(\mathbf{r}) e^{-i2\pi \mathbf{k}\cdot\mathbf{r}} d\mathbf{r}. \tag{5.67}$$

As shown in Fig. 5.11, $M^*_{j_1}(\mathbf{k}' + \mathbf{k}_1)$ can be replaced with $M^*_{j_1,\text{cal}}(\mathbf{k}' + \mathbf{k}_1)$ which is zero outside of the central calibration region. In addition, we change variables, letting $\mathbf{k} = \mathbf{k}' + \mathbf{k}_2$ and define

$$M_{j_2,\text{fit}(\mathbf{k}_2)}(\mathbf{k}) \triangleq S_{\text{fit}}(\mathbf{k} - \mathbf{k}_2) M_{j_2}(\mathbf{k}), \tag{5.68}$$

giving

$$\begin{aligned}
\widehat{c}(j_1, \mathbf{k}_1; j_2, \mathbf{k}_2) &= \int M^*_{j_1,\text{cal}}(\mathbf{k} - \mathbf{k}_2 + \mathbf{k}_1) M_{j_2,\text{fit}(\mathbf{k}_2)}(\mathbf{k}) d\mathbf{k} && (5.69)\\
&= \int \left[\int m_{j_1,\text{cal}}(\mathbf{r}_1) e^{-i2\pi(\mathbf{k}-\mathbf{k}_2+\mathbf{k}_1)\cdot\mathbf{r}_1} d\mathbf{r}_1 \right]^* \\
&\quad \int m_{j_2,\text{fit}(\mathbf{k}_2)}(\mathbf{r}_2) e^{-i2\pi\mathbf{k}\cdot\mathbf{r}_2} d\mathbf{r}_2 \; d\mathbf{k} && (5.70)\\
&= \int \int m^*_{j_1,\text{cal}}(\mathbf{r}_1) m_{j_2,\text{fit}(\mathbf{k}_2)}(\mathbf{r}_2) \\
&\quad \int e^{i2\pi(\mathbf{k}-\mathbf{k}_2+\mathbf{k}_1)\cdot\mathbf{r}_1} e^{-i2\pi\mathbf{k}\cdot\mathbf{r}_2} d\mathbf{k} \; d\mathbf{r}_1 \; d\mathbf{r}_2 && (5.71)\\
&= \int \int m^*_{j_1,\text{cal}}(\mathbf{r}_1) m_{j_2,\text{fit}(\mathbf{k}_2)}(\mathbf{r}_2) e^{-i2\pi(\mathbf{k}_2-\mathbf{k}_1)\cdot\mathbf{r}_1} \\
&\quad \int e^{i2\pi\mathbf{k}\cdot(\mathbf{r}_2-\mathbf{r}_1)} d\mathbf{k} \; d\mathbf{r}_1 \; d\mathbf{r}_2 && (5.72)\\
&= \int \int m^*_{j_1,\text{cal}}(\mathbf{r}_1) m_{j_2,\text{fit}(\mathbf{k}_2)}(\mathbf{r}_2) e^{-i2\pi(\mathbf{k}_2-\mathbf{k}_1)\cdot\mathbf{r}_1} \\
&\quad \delta(\mathbf{r}_2 - \mathbf{r}_1) d\mathbf{r}_1 \; d\mathbf{r}_2 && (5.73)\\
&= \int m^*_{j_1,\text{cal}}(\mathbf{r}) m_{j_2,\text{fit}(\mathbf{k}_2)}(\mathbf{r}) e^{-i2\pi(\mathbf{k}_2-\mathbf{k}_1)\cdot\mathbf{r}} d\mathbf{r} && (5.74)\\
&= \text{FT}\left\{ m^*_{j_1,\text{cal}}(\mathbf{r}) m_{j_2,\text{fit}(\mathbf{k}_2)}(\mathbf{r}) \right\} (\mathbf{k}_2 - \mathbf{k}_1). && (5.75)
\end{aligned}$$

Equation 5.75 expresses the correlation value approximation as the Fourier transform of $m^*_{j_1,\text{cal}}(\mathbf{r})m_{j_2,\text{fit}(\mathbf{k}_2)}(\mathbf{r})$. Both $m_{j_1,\text{cal}}(\mathbf{r})$ and $m_{j_2,\text{fit}(\mathbf{k}_2)}(\mathbf{r})$ can be calculated by a Fourier transformation of central calibration region data. While $\mathbf{k}_2$ is fixed by $m_{j_2,\text{fit}(\mathbf{k}_2)}(\mathbf{r})$, the Fourier transform of $m^*_{j_1,\text{cal}}(\mathbf{r})m_{j_2,\text{fit}(\mathbf{k}_2)}(\mathbf{r})$ contains the correlation values for *all* values of $\mathbf{k}_1$. Since a discrete Fourier transform is used in practice, correlation values for all values of $\mathbf{k}_1$ are not generated; the current implementation uses a grid with an oversampling ratio of 3, allowing fine sampling of $\mathbf{k}_1$, and the ability to interpolate using convolution interpolation without apodization.

By taking the Fourier transform of $m^*_{j_1,\text{cal}}(\mathbf{r})m_{j_2,\text{fit}(\mathbf{k}_2)}(\mathbf{r})$ one is able to compute all of the needed correlation values for a given j_1, j_2 and $\mathbf{k}_2$. By repeating this procedure, varying j_1, j_2 and $\mathbf{k}_2$, all of the correlation value grids can be generated. In the interest of reducing computation time, it is preferable to repeat this procedure as few times as possible, sampling $\mathbf{k}_2$ coarsely while retaining high accuracy. It is possible to take advantage of the diagonal nature of the Fourier transform of the correlation values on a $\mathbf{k}_1$, $\mathbf{k}_2$ grid, as shown in Fig. 5.12. By interpolating diagonally on $\mathbf{k}_1$, $\mathbf{k}_2$ grid, instead of vertically in the $\mathbf{k}_2$ direction, a diagonal stop-band and diagonal pass-bands are created. Even with an oversampling ratio of one or less in the $\mathbf{k}_2$ direction, it is possible to achieve high accuracy interpolation.

5.9 Methods

APPEAR for 2-D non-Cartesian trajectories was implemented in C++. Computation of linear combination weights was implemented using the straightforward calculation described in Fig. 4.4 and using correlation values to reduce the computation (Fig. 5.10).

Numerical simulations were performed using a Shepp-Logan phantom and the same coil architecture used for numerical simulations in [36]. A variable density spiral trajectory was used to encode the Shepp-Logan phantom. This 16-interleave spiral trajectory was designed to have an FOV of 1.5X the FOV of the object, to a resolution of 6.25 pixels, providing the central calibration region. Each interleave then continued in k-space to a resolution of 1 pixel with 0.5X the FOV of the object.

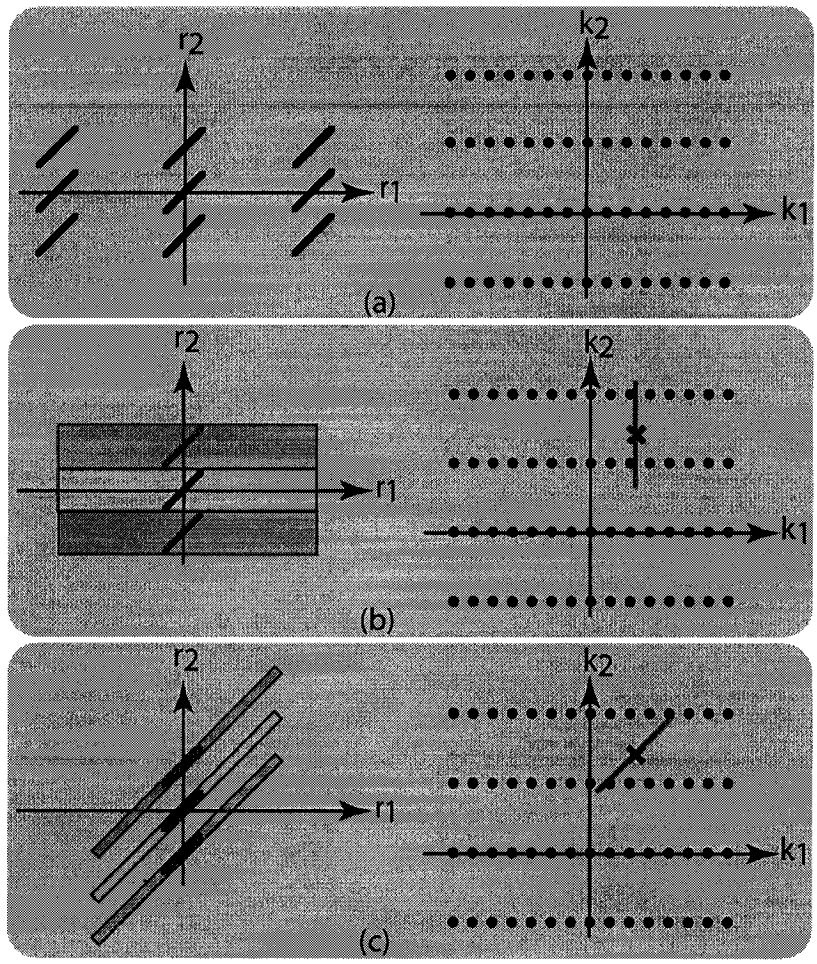


Figure 5.12: (a) (right) Correlation values sampled on a $\mathbf{k}_1$, $\mathbf{k}_2$ grid for a particular j_1, j_2 coil pair. (left) 2-D Fourier transform of these correlation value samples. Even sampling on the $\mathbf{k}_1$, $\mathbf{k}_2$ grid leads to repetition in the Fourier transformed space. Since the $\mathbf{k}_1$, $\mathbf{k}_2$ grid is sampled finely in the $\mathbf{k}_1$ direction, repetitions in the $\mathbf{r}_1$ direction are spaced far apart and the unwanted repetitions can be suppressed using convolution interpolation (gridding). In the $\mathbf{k}_2$ direction, the sampling is more coarse leading to more closely spaced repetitions in the $\mathbf{r}_2$ direction. (b) Convolution interpolation in the $\mathbf{k}_2$ direction requires the stop bands (gray boxes) to border on the pass band, making it difficult to achieve a high accuracy interpolation. (c) By using diagonal convolution interpolation, it is possible to take advantage of the diagonal nature of the Fourier transform of the correlation values. In the diagonal direction, the pass band and stop bands can be much more narrow, allowing for a generous transition band.

The optimal trajectory, subject to slew rate and maximum gradient limitations, that met the design criteria was used, designed using the method outlined in [39]. For this trajectory, the k-space area of the central calibration region is under 2.6% of the total k-space covered.

Scanner experiments were performed on a 1.5T scanner (Signa HDx, GE Healthcare, Waukesha, WI) using a stack-of-spirals trajectory, with a 16-interleave variable-density spiral trajectory having the same constraints as in the numerical experiment. Reconstruction was performed by taking the Fourier transform in the stack direction and then treating each spiral as a separate data set, reconstructing each plane individually using gridding and APPEAR.

Axial scans of a ball phantom and brain were taken using an 8-channel high-resolution head coil. The FOV of these scans was 20 cm $\times$ 20 cm $\times$ 12.8 cm and the resolution was 1 mm $\times$ 1 mm $\times$ 2 mm. A 50 ms TR was used, giving a total scan time of 52 seconds. The readout time for each interleave was 5.4 ms.

Axial scans of a resolution phantom were taken using an 8-channel cardiac coil (MRI Devices Corp., Waukesha Wisconsin). The FOV of these scans was 12 cm $\times$ 12 cm $\times$ 9.6 cm and the resolution was 0.6 mm $\times$ 0.6 mm $\times$ 1.5 mm. A 50 ms TR was used, giving a total scan time of 52 seconds. The readout time for each interleaf was 6.2 ms.

5.10 Results

Figure 5.13 shows the results of the numerical simulation. The aliasing artifacts from reduced gradient encoding, shown in the gridding reconstruction in (a) have been removed in the APPEAR reconstruction shown in (b).

Results for the 8-channel head coil are shown in Fig. 5.14. The APPEAR method (b), (d), is able to remove the aliasing artifacts, seen in the gridding reconstructions (a), (c). Figure 5.15 shows the results with the 8-channel cardiac coil. Once again, APPEAR is able to remove the aliasing artifacts shown in the gridding reconstruction.

There was no difference between image quality between images reconstructed using the straightforward APPEAR procedure (Fig. 4.4) and the modified procedure

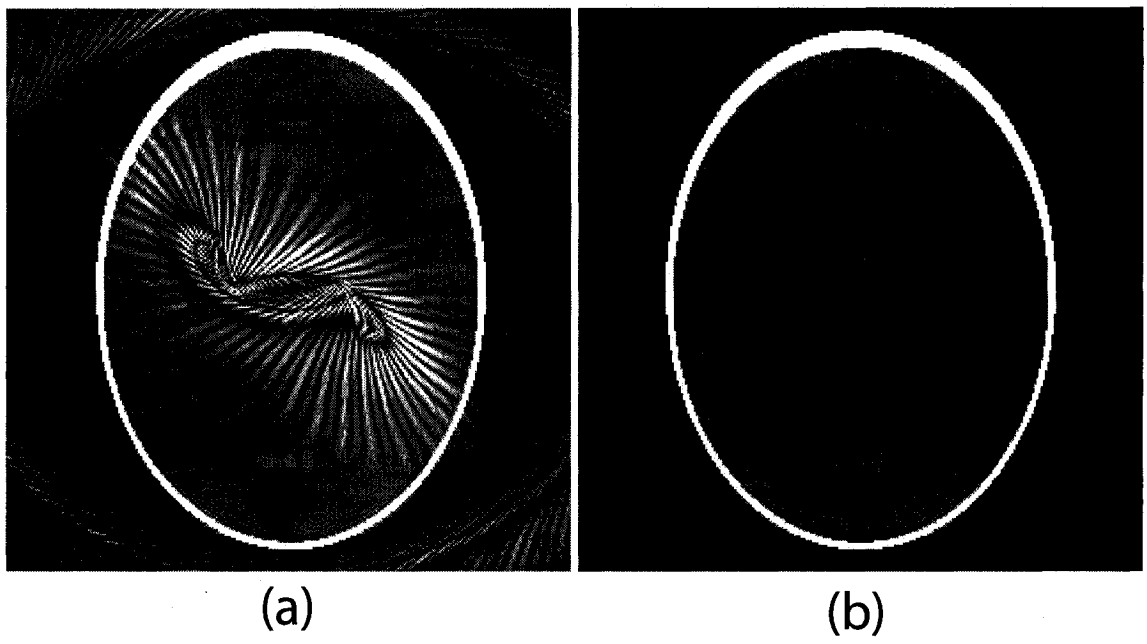


Figure 5.13: (a) Gridding reconstruction of the Shepp-Logan numerical phantom, encoded with variable-density spiral trajectory. (b) APPEAR reconstruction of the same dataset as in (a).

(Fig. 5.10) which takes advantage of correlation values. However, the straightforward procedure took approximately 30 minutes on a 3 GHz Pentium 4 processor to reconstruct one spiral image, whereas the modified procedure was able to reconstruct the same spiral image in approximately 3 minutes.

5.11 Discussion and Conclusions

In this chapter, the concept of correlation values was defined and developed. Correlation values are useful aids for understanding local projection calibration and interpolation. In addition, correlation values provide a common framework with which to view many different parallel imaging methods. For example, APPEAR and PARS cannot be compared in terms of their sensitivity function estimation, as APPEAR

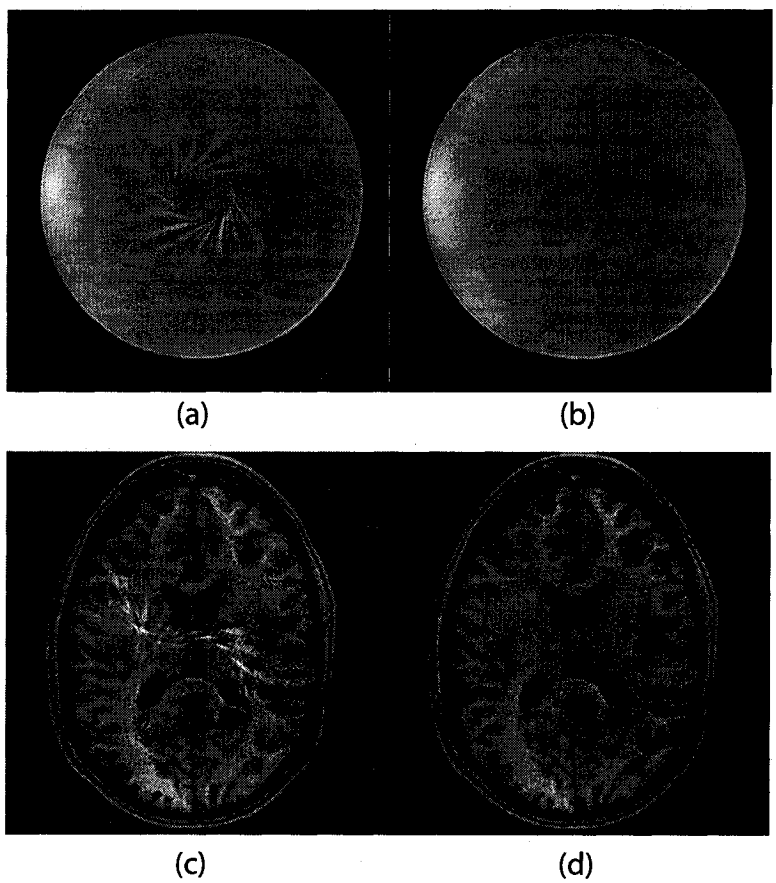


Figure 5.14: Single axial slice from stack-of-spirals trajectory with 8-channel head coil. (a) Ball phantom, gridding reconstruction (b) ball phantom, APPEAR reconstruction. (c) Brain, gridding reconstruction. (d) Brain, APPEAR reconstruction.

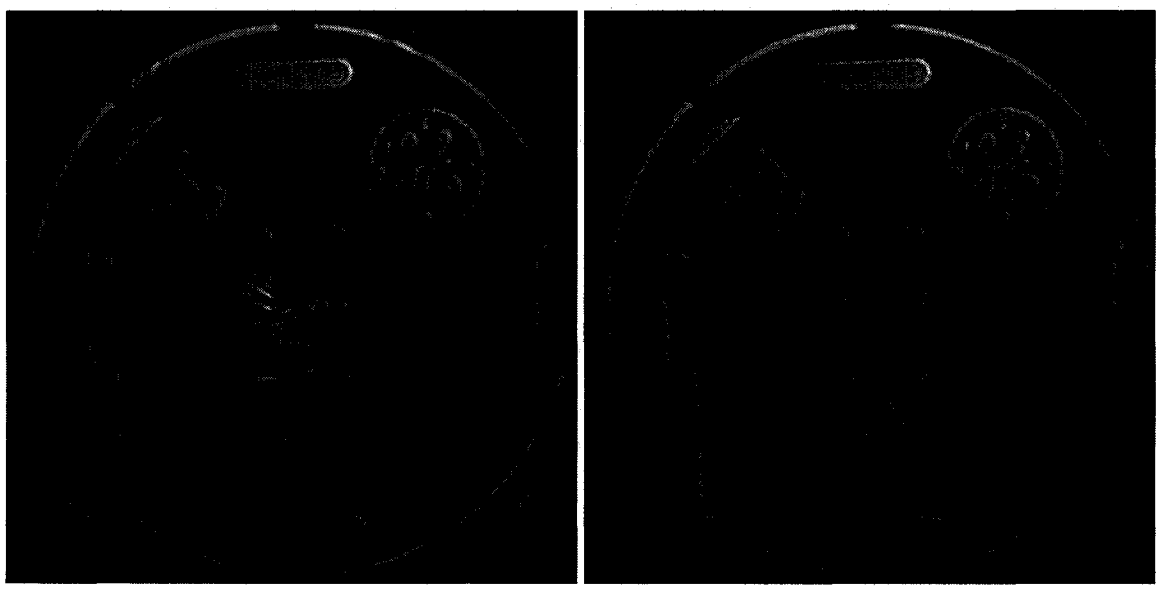


Figure 5.15: (a) Single axial slice of a resolution phantom from stack-of-spirals trajectory with 8-channel cardiac coil. (a) Gridding reconstruction. (b) APPEAR reconstruction.

makes no such estimation. However, both methods can be formulated as approximating correlation values and can be compared in terms of how the correlation value approximations differ.

In addition to providing a useful framework for understanding methods such as GRAPPA, PARS and APPEAR, correlation values lead to a simple way to reduce the computation burden of APPEAR. By using correlation values as outlined in this chapter, the computation time was reduced ten-fold compared to a straightforward implementation as outlined in chapter 4. Certainly there is more work to be done to further reduce the computation time for an APPEAR reconstruction, but the use of correlation values is a significant stride toward reducing computation concerns as an impediment toward adoption of APPEAR.

Chapter 6

Summary and Future Recommendations

Modern MR scanners currently include gradient systems that can encode the magnetization information along arbitrary k-space trajectories and multiple RF receiver-coils that can take advantage of receiver-coil sensitivity encoding. These components of MR scanners enable fast acquisition of the data necessary to form an image.

Development and improvement of methods that can efficiently reconstruct high quality images from MR data is an important area of MRI research. With the more widespread use of non-Cartesian k-space trajectories and parallel imaging, reconstruction methods for MRI have, out of necessity become more sophisticated. In this dissertation, I have presented a number of advances in MRI reconstruction:

- A number of technical refinements to the basic gridding algorithm were developed. These refinements include using a minimal oversampling ratio, presampling the kernel, two methods for designing high performance gridding kernels, the use of the block storage format and the ability to take advantage of flexibly choosing the field-of-view for the reconstructed image. For 3-D non-Cartesian trajectories, these technical advances can be used together to achieve a thirty-fold reduction in computation time and three-fold reduction in computer memory requirements.

- A new method, Anti-aliasing Partially Parallel Encoded Acquisition Reconstruction (APPEAR), was developed for reconstructing images encoded with both non-Cartesian k-space trajectories and multiple receiver coils. The APPEAR method is non-iterative and does not require the receiver coil sensitivities to be known. In the development of APPEAR, the theory behind local projection calibration was derived and interpolation in the calibration region was introduced as a way to take advantage of local projection calibration for arbitrary k-space trajectories.

- The concept of correlation values was developed and shown to provide a useful framework for understanding parallel imaging reconstruction methods. In addition, by using correlation values, the APPEAR method can be implemented more efficiently, leading to a ten-fold reduction in computation time.

6.1 Future Recommendations

While this dissertation includes a number of advances in reconstruction methods for fast MRI, combining non-Cartesian trajectories with parallel imaging remains an active area of research. In particular, balancing reconstruction efficiency with image quality requirements remains a challenge. While in the long term more computing power might alleviate some of the efficiency requirements, in the short term reconstruction efficiency is bound to be a limiting factor as it becomes more commonplace to use a large number of coils in parallel and acquire large 3-D data sets. The following list gives some specific areas of future work related to the APPEAR method:

- A number of parameters used in the APPEAR method have not been adequately or systematically studied. While it is believed that the parameters chosen in this work were reasonable, further study could lead to a more refined implementation of the method. These parameters include: the radius, κ, used to define the local set, the size of the central calibration region, the amount of apodization used to increase the accuracy of interpolation in the central calibration region and the oversampling factor in the central calibration region.

- In the current implementation of APPEAR, data is synthesized directly onto a Cartesian grid. Since APPEAR can synthesize data at any k-space location within the extent of acquired k-space, other possibilities exists. For example, unacquired arms of a spiral trajectory could be synthesized to form a full field-of-view spiral trajectory for each coil; gridding could then be used to reconstruct each coil's image. More work needs to be done to determine where best to place the synthesis locations.

- Related to the above point, it has been shown in [38] that for the radial trajectory, filling in unacquired radial lines can lead to the option for interpolating the linear coefficient weight values which can drastically reduce the computation burden. While such an approach can be used with linear combination weights generated using APPEAR, the effect of interpolating weight values on the error in the estimated encoding functions has not been systematically studied. In addition, it is not clear how well such an approach can be used with variable-density spiral trajectories or the 3-D cones trajectory.

- Partial k-space encoding is another technique to reduce the encoding time necessary to acquire enough data for an image. Future work is needed to determine how partial k-space encoding can be combined with the APPEAR method.

- Thus far, APPEAR has been implemented for 2-D non-Cartesian trajectories, but has not yet been implemented for 3-D non-Cartesian trajectories. 3-D scans stand the most to gain from reduced encoding time and such an implementation is important for future work.

- While much of the theory behind APPEAR has been developed and APPEAR has been implemented for 2-D non-Cartesian trajectories, only limited *in vivo* results have been obtained. The next step in evaluating APPEAR for clinical use is to use it with clinically relevant scans and evaluate its performance.

Appendix A

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. A.1. The periodicity of $c_s(x)$ allows us to write the infinite sum in Eq. 3.3 as a finite sum and thus quickly calculate the aliasing amplitude for a sampled kernel.

Starting from Eq. 3.3, I derive the aliasing amplitude for a sampled interpolated kernel, $c(i) = c_s(i)h(i)$, as follows:

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

I now group the terms of $c(i)$ separated by SG; the periodicity of $c_s(x)$ allows us to

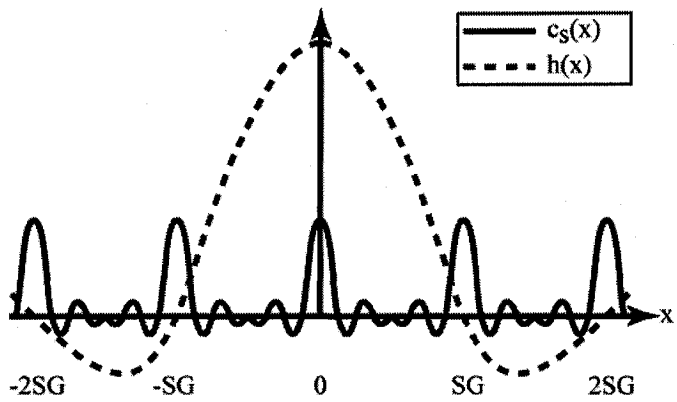


Figure A.1: The sampled and interpolated kernel, $\widehat{c}(x)$, can be viewed as the periodic function $c_s(x) = c(x) * \text{Ш}(SGx)$, with period SG, modulated by the envelope function $h(x)$, which is a sinc function for nearest-neighbor interpolation and a sinc-squared function for linear interpolation.

simplify $c(i + SG) = c_s(i + SG)h(i + SG) = c_s(i)h(i + SG)$.

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

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

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

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

Noting that the $p = 0$ term in Eq. A.1 is independent of $c_s(x)$, one can separate $\varepsilon(i)$ into the quadrature sum of two terms:

$$\varepsilon[i] = \sqrt{(\varepsilon_1[i])^2 + (\varepsilon_2[i])^2}, \tag{A.3}$$

where

$$\varepsilon_1[i] = \sqrt{\left[\frac{\widehat{h}(i)}{h(i)}\right]^2 - 1} \tag{A.4}$$

$$\varepsilon_2[i] = \sqrt{\sum_{p=1}^{S-1}\left[\frac{\widehat{h}(i+Gp)c_s(i+Gp)}{h(i)c_s(i)}\right]^2}. \tag{A.5}$$

A.1 Nearest-Neighbor Interpolation

For nearest-neighbor interpolation, $h(x) = \mathrm{sinc}\,(x/SG)$. Substituting this into Eq. A.2 and A.4, $\widehat{h}(i) = 1$ and

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

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

$$\text{nearest-neighbor } \varepsilon_1[i] \approx \frac{\pi|i|}{\sqrt{3}SG}. \tag{A.6}$$

When $i = -N/2$, $\varepsilon_1 \approx 0.91/\alpha S$ as in Eq. 3.7.

A.2 Linear Interpolation

For linear interpolation, $h(x) = \mathrm{sinc}^2\,(x/SG)$. Once again, I substitute this into Eq. A.2 and A.4 to obtain $\widehat{h}(i) = \sqrt{\frac{2}{3} + \frac{1}{3}\cos\left(2\pi\frac{i}{SG}\right)}$ and

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

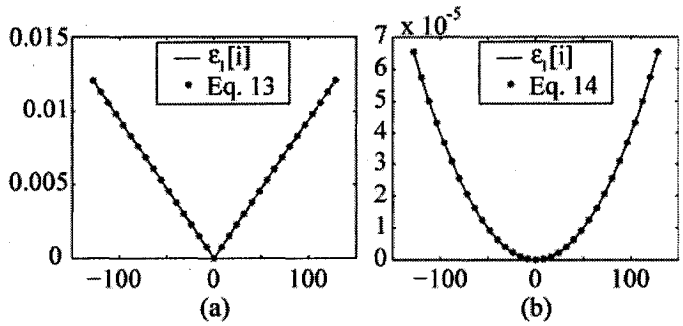


Figure A.2: Although Eqs. A.6 and A.7 are approximations for $\varepsilon_1[i]$, they are very accurate approximations. In this example, $N = 256$, $\alpha = 1.25$ and $S = 60$. (a) shows the agreement for nearest-neighbor interpolation. Eq. A.6 is within 44 parts-per-million of $\varepsilon_1[i]$ in this case. (b) shows the agreement for linear interpolation. Eq. A.7 is within 209 parts-per-million of $\varepsilon_1[i]$ in this case.

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

$$\text{linear } \varepsilon_1[i] \approx \frac{\pi^2}{3\sqrt{5}} \left(\frac{i}{SG}\right)^2. \tag{A.7}$$

When $i = -N/2$, $\varepsilon_1 \approx 0.37/(\alpha S)^2$ as in Eq. 3.8.

Equations A.6 and A.7 provide simple expressions for $\varepsilon_1[i]$, the part of the aliasing amplitude due to nearest-neighbor and linear interpolation respectively. Figure A.2 demonstrates that these approximations are also highly accurate. Equation A.5 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. 3.4.

Appendix B

Apodization Correction Function of a Sampled Kernel

With a continuous kernel, obtaining the exact apodization correction function requires one to have an analytic solution for the inverse Fourier transform of the kernel. With the sampled kernel one 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 C

Calculating the Optimal Sampled Kernel in MATLAB

The SOCP in section 3.5 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, I implement the following SOCP which, though slightly more complicated, gives the same result and is better suited for implementation:

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

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 [45]. In addition, the SeDuMi package is a freely available package that will solve many optimization problems, including SOCP problems [46] [47]. Commercial packages, such as MOSEK

(www.mosek.com), are also available. Here, I provide a MATLAB program which implements the SOCP shown above, using the yalmip MATLAB toolbox and SeDuMi [48]. N, G, W and S are as defined in Tables 4.1 and 3.2.

```
% 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
```

Appendix D

Noise in APPEAR Reconstructions

The noise from each receiver coil is modeled as a random process with Gaussian statistics and zero mean. The variance and cross-correlation of the noise between the receiver coils is given by the $N_c \times N_c$ receiver noise matrix Ψ, as defined by Pruessmann *et al.* [19]. Denoting the noise random process for coil j as $n_j(t)$, the data acquired at location $(j, \mathbf{k})$ can be written as $d_j(\mathbf{k}) = \mathbf{e}_j^T(\mathbf{k})\mathbf{m} + n_j(t_\mathbf{k})$ and the expected value of $n_j^*(t_\mathbf{k})n_{j'}(t_{\mathbf{k}'})$ is given by

$$E\left[n_j^*(t_\mathbf{k})n_{j'}(t_{\mathbf{k}'})\right] = \begin{cases} \Psi_{jj'} & \text{if } t_\mathbf{k} = t_{\mathbf{k}'} \\ 0 & \text{otherwise} \end{cases}. \tag{D.1}$$

The noise variance for the voxel location $\mathbf{r}$ in the reconstructed image for coil j can be expressed as

$$\sigma_j^2(\mathbf{r}) = E\left[\left|\sum_{\mathbf{k}\in\mathcal{G}} \exp(i2\pi\mathbf{k}^T\mathbf{r})\mathbf{w}_{j,\mathcal{L}(\mathbf{k})}^T\mathbf{n}_{\mathcal{L}(\mathbf{k})}\right|^2\right], \tag{D.2}$$

where $\mathcal{G}$ is the set of k-space locations forming a full-FOV grid and $\mathbf{n}_{\mathcal{L}(\mathbf{k})}$ is a vector containing the noise values of the data acquired in the local neighborhood of k-space location $\mathbf{k}$. After some manipulation, $\sigma_j^2(\mathbf{r})$ can be written as

$$\sigma_j^2(\mathbf{r}) = \sum_{\Delta\mathbf{k}\in\mathcal{G}} \exp(i2\pi\Delta\mathbf{k}^T\mathbf{r})\phi_j(\Delta\mathbf{k}), \tag{D.3}$$

where

$$\phi_j(\Delta\mathbf{k}) = \sum_{j',j''} \Psi_{j'j''} \sum_{\mathbf{k}\in\mathcal{G}} \sum_{\mathbf{k}'\in\mathcal{I}(\mathbf{k},\Delta\mathbf{k})} w^*_{j,j'}(\mathbf{k},\mathbf{k}') w_{j,j''}(\mathbf{k}-\Delta\mathbf{k},\mathbf{k}'). \tag{D.4}$$

The k-space set $\mathcal{I}(\mathbf{k},\Delta\mathbf{k}) = \{\mathbf{k}'|(j,\mathbf{k}) \in \mathcal{L}(\mathbf{k}) \cap \mathcal{L}(\mathbf{k}-\Delta\mathbf{k}), \mathbf{k}-\Delta\mathbf{k} \in \mathcal{G}\}$ is the intersection set between two local neighborhoods, one centered at $\mathbf{k}$ and the other centered at $\mathbf{k}-\Delta\mathbf{k}$, given that both centers lie on the grid. Equation D.3 can be viewed as the inverse discrete Fourier transform of $\phi_j(\Delta\mathbf{k})$. Since the set $\mathcal{I}(\mathbf{k},\Delta\mathbf{k})$ is null when $\|\Delta\mathbf{k}\| > 2\kappa$, twice the radius of the local neighborhood used for data synthesis, $\phi_j(\Delta\mathbf{k}) = 0$ for $\|\Delta\mathbf{k}\| > 2\kappa$. Thus, $\sigma_j^2(\mathbf{r})$ is a low frequency function of $\mathbf{r}$. When the noise from the coils is not correlated, the average noise variance in a voxel is given by

$$\phi_j(\mathbf{0}) = \sum_{\mathbf{k}\in\mathcal{G}} \sum_{(j',\mathbf{k}')\in\mathcal{L}(\mathbf{k})} \Psi_{j'j'} \left|w_{j,j'}(\mathbf{k},\mathbf{k}')\right|^2. \tag{D.5}$$

Thus, large weights will increase the noise in the image. While regularization can be used to keep the weights from getting too large, this can result in weights that do not sufficiently suppress the aliasing artifacts.

Bibliography

[1] J. O'Sullivan, "A fast sinc function gridding algorithm for Fourier inversion in computer tomography," *Medical Imaging, IEEE Transactions on*, vol. MI-4, pp. 200–207, 1985.

[2] H. Schomberg and J. Timmer, "The gridding method for image reconstruction by Fourier transformation," *Medical Imaging, IEEE Transactions on*, vol. 14, no. 3, pp. 596–607, 1995.

[3] S. Schaller, T. Flohr, and P. Steffen, "An efficient Fourier method for 3-D radon inversion in exact cone-beam CT reconstruction," *Medical Imaging, IEEE Transactions on*, vol. 17, no. 2, pp. 244–250, 1998.

[4] M. Bronstein, A. Bronstein, M. Zibulevsky, and H. Azhari, "Reconstruction in diffraction ultrasound tomography using nonuniform FFT," *Medical Imaging, IEEE Transactions on*, vol. 21, no. 11, pp. 1395–1401, 2002.

[5] A. Dutt and V. Rokhlin, "Fast Fourier transforms for nonequispaced data," *SIAM J. Sci Comput.*, vol. 14, no. 6, pp. 1369–1393, 1993.

[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. 179–86, 1999.

[7] V. Rasche, R. Proksa, R. Sinkus, P. Bornert, and H. Eggers, "Resampling of data between arbitrary grids using convolution interpolation," *Medical Imaging, IEEE Transactions on*, vol. 18, no. 5, pp. 385–92, 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," *Signal Processing, IEEE Transactions on [see also Acoustics, Speech, and Signal Processing, IEEE Transactions on]*, vol. 51, no. 2, pp. 560–574, 2003.

[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. 343–52, 2004.

[11] J. Jackson, C. Meyer, D. Nishimura, and A. Macovski, "Selection of a convolution function for Fourier inversion using gridding," *Medical Imaging, IEEE Transactions on*, vol. 10, no. 3, pp. 473–478, 1991.

[12] L. Greengard and J.-Y. Lee, "Accelerating the nonuniform fast fourier transform," *SIAM Rev.*, vol. 46, pp. 443–454, 2004.

[13] M. Frigo and S. G. Johnson, *The FFTW web page*, 2004. [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. 656–62, 1995.

[15] F. Wajer, E. Woudenberg, R. de Beer, M. Fuderer, A. Mehlkopf, and D. van Ormondt, "Simple equation for optimal gridding parameters," in *Proceedings of the 7th Annual Meeting of ISMRM*, Philadelphia, 1999, p. 663.

[16] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret, "Applications of second-order cone programming," *Linear Algebra and its Applications*, vol. 284, no. 1-3, pp. 193–228, 1998.

[17] M. A. Griswold, P. M. Jakob, M. Nittka, J. W. Goldfarb, and A. Haase, "Partially parallel imaging with localized sensitivities (PILS)," *Magn. Reson. Med.*, vol. 44, no. 4, pp. 602–9, 2000.

[18] D. K. Sodickson and W. J. Manning, "Simultaneous acquisition of spatial harmonics (SMASH): fast imaging with radiofrequency coil arrays," *Magn. Reson. Med.*, vol. 38, no. 4, pp. 591–603, 1997.

[19] K. P. Pruessmann, M. Weiger, M. B. Scheidegger, and P. Boesiger, "SENSE: sensitivity encoding for fast MRI," *Magnetic Resonance in Medicine*, vol. 42, no. 5, pp. 952–62, 1999.

[20] K. P. Pruessmann, M. Weiger, P. Bornert, and P. Boesiger, "Advances in sensitivity encoding with arbitrary k-space trajectories," *Magn. Reson. Med.*, vol. 46, no. 4, pp. 638–51, 2001.

[21] W. E. Kyriakos, L. P. Panych, D. F. Kacher, C. F. Westin, S. M. Bao, R. V. Mulkern, and F. A. Jolesz, "Sensitivity profiles from an array of coils for encoding and reconstruction in parallel (SPACE RIP)," *Magn. Reson. Med.*, vol. 44, no. 2, pp. 301–8, 2000.

[22] E. N. Yeh, C. A. McKenzie, M. A. Ohliger, and D. K. Sodickson, "Parallel magnetic resonance imaging with adaptive radius in k-space (PARS): constrained image reconstruction using k-space locality in radiofrequency coil encoded data," *Magn. Reson. Med.*, vol. 53, no. 6, pp. 1383–92, 2005.

[23] C. A. McKenzie, E. N. Yeh, M. A. Ohliger, M. D. Price, and D. K. Sodickson, "Self-calibrating parallel imaging with automatic coil sensitivity extraction," *Magn. Reson. Med.*, vol. 47, no. 3, pp. 529–38, 2002.

[24] M. A. Griswold, P. M. Jakob, R. M. Heidemann, M. Nittka, V. Jellus, J. Wang, B. Kiefer, and A. Haase, "Generalized autocalibrating partially parallel acquisitions (GRAPPA)," *Magn Reson Med*, vol. 47, no. 6, pp. 1202–10, 2002.

[25] Z. Wang, J. Wang, and J. A. Detre, "Improved data reconstruction method for GRAPPA," *Magn. Reson. Med.*, vol. 54, no. 3, pp. 738–742, 2005.

[26] P. M. Jakob, M. A. Griswold, R. R. Edelman, and D. K. Sodickson, "AUTO-SMASH: a self-calibrating technique for SMASH imaging. SiMultaneous Acquisition of Spatial Harmonics," *Magma*, vol. 7, no. 1, pp. 42–54, 1998.

[27] R. M. Heidemann, M. A. Griswold, A. Haase, and P. M. Jakob, "VD-AUTO-SMASH imaging," *Magn. Reson. Med.*, vol. 45, no. 6, pp. 1066–74, 2001.

[28] M. A. Griswold, R. M. Heidemann, and P. M. Jakob, "Direct parallel imaging reconstruction of radially sampled data using GRAPPA with relative shifts," in *Proc. Eleventh ISMRM*, 2003, p. 2349.

[29] K. A. Heberlein, Y. Kadah, and X. Hu, "Segmented spiral parallel imaging using GRAPPA," in *Proc. Twelfth ISMRM*, 2004, p. 328.

[30] R. M. Heidemann, M. A. Griswold, G. Krger, S. Kannengiesser, B. Kiefer, and P. M. Jakob, "Fast parallel image reconstructions for spiral trajectories," in *Second International Workshop on Parallel MRI*, 2004.

[31] K. A. Heberlein and X. Hu, "Auto-calibrated parallel imaging using dual-density spirals," in *Second International Workshop on Parallel MRI*, 2004.

[32] K. Heberlein and X. Hu, "Auto-calibrated parallel spiral imaging," *Magn. Reson. Med.*, vol. 55, no. 3, pp. 619–625, 2006.

[33] A. Arunachalam, A. Lu, E. K. Brodsky, and W. F. Block, "GRAPPA for the 3D radial trajectory (VIPR)," in *Proc. Thirteenth ISMRM*, 2005, p. 2674.

[34] C. M. Tsai and D. G. Nishimura, "Reduced aliasing artifacts using variable-density k-space sampling trajectories," *Magn. Reson. Med.*, vol. 43, no. 3, pp. 452–8, 2000.

[35] P. Beatty, D. Nishimura, and J. Pauly, "Rapid gridding reconstruction with a minimal oversampling ratio," *IEEE Trans. Med. Imag.*, vol. 24, no. 6, pp. 799–808, 2005.

[36] M. A. Griswold, S. Kannengiesser, R. M. Heidemann, J. Wang, and P. M. Jakob, "Field-of-view limitations in parallel imaging," *Magn. Reson. Med.*, vol. 52, no. 5, pp. 1118–26, 2004.

[37] C. A. McKenzie, M. A. Ohliger, E. N. Yeh, M. D. Price, and D. K. Sodickson, "Coil-by-coil image reconstruction with SMASH," *Magn. Reson. Med.*, vol. 46, no. 3, pp. 619–23, 2001.

[38] A. A. Samsonov, W. F. Block, A. Arunachalam, and A. S. Field, "Advances in locally constrained k-space-based parallel MRI," *Magn. Reson. Med.*, vol. 55, no. 2, pp. 431–438, 2006.

[39] P. T. Gurney, B. A. Hargreaves, and D. G. Nishimura, "Design and analysis of a practical 3D cones trajectory," *Magn. Reson. Med.*, vol. 55, no. 3, pp. 575–582, 2006.

[40] A. Macovski, "Noise in MRI," *Magn. Reson. Med.*, vol. 36, no. 3, pp. 494–7, 1996.

[41] P. B. Roemer, W. A. Edelstein, C. E. Hayes, S. P. Souza, and O. M. Mueller, "The NMR phased array," *Magn. Reson. Med.*, vol. 16, no. 2, pp. 192–225, 1990.

[42] P. Qu, C. Wang, and G. X. Shen, "Discrepancy-based adaptive regularization for GRAPPA reconstruction," *J Magn Reson Imaging*, vol. 24, no. 1, pp. 248–55, 2006.

[43] H. Sedarat and D. G. Nishimura, "On the optimality of the gridding reconstruction algorithm," *Medical Imaging, IEEE Transactions on*, vol. 19, no. 4, pp. 306–17, 2000.

[44] K. A. Heberlein and X. Hu, "Kriging and GRAPPA: A new perspective on parallel imaging reconstruction," in *Proc. Fourteenth ISMRM*, 2006, p. 2465.

[45] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret, *Second-Order Cone Programming (SOCP)*, 1998. [Online]. Available: http://www.stanford.edu/~boyd/SOCP.html

[46] J. F. Sturm, "Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones," *Optimization Methods and Software*, no. 11-12, pp. 625–653, 1999.

[47] ——, *SeDuMi 1.05*, 2003. [Online]. Available: http://fewcal.kub.nl/sturm/software/sedumi.html

[48] J. Löfberg, *YALMIP 3*, 2004. [Online]. Available: http://control.ee.ethz.ch/~joloef/yalmip.msql