Parallel Imaging and Compressed Sensing

Nicholas Dwork

Abstract

This document provides an introduction to parallel imaging and compressed sensing in MRI.

1 Background

Magnetic Resonance Imaging is conventionally dependent on the Fourier transform. In this section, we will define the Fourier transform (as used in this document, for there are several similar definitions) and list some of its properties. For a more thorough review of the Fourier Transform and its applications, please review the notes at `http://nicholasdwork.com/teaching/1706ee102a/`.

The Fourier Transform is defined as

$$\hat{f}(k) = \mathcal{F}\{f\}(k) = \int_{-\infty}^{\infty} f(x)\, e^{-i2\pi\, kx}\, dx.$$

The Fourier transform is a function $\mathcal{F}$ that accepts a function f as input and outputs another function $\hat{f}$. The Fourier transform is invertible, and its inverse is

$$f(x) = \mathcal{F}^{-1}\{\hat{f}\}(x) = \int_{-\infty}^{\infty} \hat{f}(k)\, e^{i2\pi\, kx}\, dk.$$

There are many powerful theorems associated with the Fourier Transform. Here are a few:

- **Convolution Theorem:** $\mathcal{F}\{f * g\} = \hat{f}\,\hat{g}$. That is, the Fourier transform of f convolved with g equals the Fourier transform of f multiplied by the Fourier transform of g. This is an extremely powerful theorem. Convolution, which is difficult, is converted into multiplication, which is easy.
- **Fourier Shift Theorem:** $\mathcal{F}\{f(x-\Delta)\}(k) = e^{-i2\pi\, k\Delta}\, \hat{f}(k)$.
- Suppose we are provided a set of values $(\ldots, f(-2\Delta), f(-\Delta), f(0), f(\Delta), f(2\Delta)\ldots)$. Suppose further that we would like the value of $f(x)$ for some arbitrary value x. How well can we estimate this value? Wonderfully, if f is sufficiently smooth, then we can determine $f(x)$ perfectly!

  Nyquist's Theorem: Suppose f has a bandwidth of B, meaning that the support of f is a subset of $[-B, B]$. Then $f(x)$ can be determined perfectly as long as $2B < 1/\Delta$ according to

  $$f(x) = \sum_{n=-\infty}^{\infty} f(n\Delta)\,\text{sinc}\left(\frac{x - n\Delta}{\Delta}\right).$$

  The value $1/\Delta$ is often called the *sampling frequency*. With this terminology, Nyquist's theorem is often stated as follows: a function f can be perfectly reconstructed as long as the sampling frequency is greater than twice the bandwidth of the function. (Note that Nyquist's theorem specifies a sufficient condition but not a necessary one.)

The Fourier transform for a function $f : \mathbb{R}^N \to \mathbb{C}$ and its inverse are defined as

$$\hat{f}(k) = \mathcal{F}\{f\}(k) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} f(x)\, e^{-i2\pi\, k\cdot x}\, dx$$
$$f(x) = \mathcal{F}^{-1}\{\hat{f}\}(x) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} \hat{f}(k)\, e^{i2\pi\, k\cdot x}\, dk,$$

where $x \in \mathbb{R}^N$, and $k \in \mathbb{R}^N$. Here, $\cdot$ represents the dot product.

Note that

$$\hat{f}(k) = \mathcal{F}\{f\}(k) = \mathcal{F}_N\{\mathcal{F}_{N-1}\{\cdots \mathcal{F}_2\{\mathcal{F}_1\{f\}\}\cdots\}(k), \tag{1}$$

where $\mathcal{F}_i$ is the one-dimensional Fourier transform with respect to the i^{th} argument. Equivalently, it is the one-dimensional Fourier transform with respect to the i^{th} dimension.

A *hybrid space* can be constructed by computing the Fourier Transform with respect to a strict subset of the independent variables. For example,

$$h(k_x, y) = \mathcal{F}\{f\}(k_x, y) = \int_{-\infty}^{\infty} f(x, y)\, e^{-i2\pi\, k_x x}\, dx.$$

The Discrete Fourier Transform (DFT) for an N element vector f is[1]

$$\hat{f} = \text{DFT}\{f\}_m = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} f_n \, \exp\left(-i2\pi \frac{mn}{N}\right).$$

Note that the DFT is a linear combination and it can be represented by a matrix. We will denote this matrix as F. Therefore, $\hat{f} = Ff = \text{DFT}\{f\}$. F is an invertible matrix; $f = F^{-1}\hat{f}$. Actually, F is unitary; that is, $F^{-1} = F^H$, meaning its inverse is its Hermitian transpose (its conjugate transpose). The explicit expression for the inverse is

$$\text{DFT}^{-1}\{\hat{f}\}_n = \frac{1}{\sqrt{N}} \sum_{m=0}^{N-1} \hat{f}_m \, \exp\left(i\,2\pi\,\frac{mn}{N}\right).$$

There are analogous theorems for the DFT as there are for the Fourier Transform.

- **Convolution Theorem:** $\text{DFT}\{f \circledast g\} = \text{DFT}\{f\} \odot \text{DFT}\{g\}$. That is, the DFT of f circularly convolved with g equals the Hadamaard (or point-wise) product of the DFT of f with the DFT of g.
- **Shift Theorem:** $\text{DFT}\{f(x-\Delta)\}m = \exp\left(i\,2\pi \frac{m\Delta}{N}\right)\, \text{DFT}\{f\}_m$. Note that here, the shift is circular (meaning that if it ends up at a coordinate off the vector, it wraps around).

The DFT serves as a useful numerical approximation of the Fourier transform, which is described in detail here: `http://nicholasdwork.com/tutorials/approxDFT.pdf`.

2 Magnetic Resonance Imaging

At any point in time, the MRI machine acquires a data point which (approximately) adheres to the following expression

$$s(t) = \iiint_{-\infty}^{\infty} M(r)\, \exp(-i\,2\pi\, k \cdot r)\, dV, \tag{2}$$

where $k_x(t) = \gamma/(2\pi) \int_0^t G_x(\tau)\, d\tau$, $k_y(t) = \gamma/(2\pi) \int_0^t G_y(\tau)\, d\tau$, and $k_z(t) = \gamma/(2\pi) \int_0^t G_z(\tau)\, d\tau$. The scalar γ is called the gyromagnetic ratio and is particular to the element imaged; for hydrogen in water, $\gamma \approx 42.5$ MHz/T. You can see an explanation of where this expression comes from here: `https://www.youtube.com/watch?v=wrlQxlo0uT4`. In (2), $r = (x, y, z) \in \mathbb{R}^3$, and $k \in \mathbb{R}^3$. This expression assumes ideal receiver coils (or antennas) which do not distort the measurement at all. In reality, any antenna is (roughly) more sensitive to elements nearer to it and less sensitive to elements farther away. This effect apodizes the object imaged according to

$$s(t) = \iiint_{-\infty}^{\infty} \rho(r)\, M(r)\, \exp(-i\,2\pi\, k(t) \cdot r)\, dV, \tag{3}$$

where $\rho : \mathbb{R}^3 \to \mathbb{C}$ is the coil sensitivity function.

Note that (3) is a Fourier Transform! That is, $s(t) = \mathcal{F}\{\rho M\}(k(t))$. Thus, the mathematics and theorems of the Fourier Transform will be very useful for reconstructing MR imagery. Let's consider an example.

[1] Note that this definition provides the nice property that the DFT is unitary. However, the scaling factor is usually different in numerical packages.

Suppose we wanted to make a $100 \times 512 \times 512$ volume. The scanning protocol will employ phase encodes in y and z with readouts in x. Once the data is acquired, the volume can be reconstructed simply by using an inverse 3D DFT!

But how long does it take? Suppose the scan employed a 15 ms repetition time. Then the total scan time would be $512 \times 512 \times 10 \approx 43$ minutes! And that's for only one scan; a medical protocol may require 10 scans. This consumes the machine for too long and is uncomfortable for the patient. Thus, there is great interest in reducing the time required to achieve scans of comparable quality. We will now discuss two methods for doing so: Parallel Imaging and Compressed Sensing.

3 Parallel Imaging

The idea behind parallel imaging is that additional sensors (additional coils) provide us with additional information that we can use. This information should reduce the number of datapoints required to accurately reconstruct the imagery. For the remainder of this discussion, $\rho^{(c)}$ will denote the sensitivity map of the c^{th} coil. Furthermore, we will limit the discussion to two-dimensional data. (The modification to higher dimensions is methodical.)

3.1 Square Root of Sum-of-Squares

If the data is fully sampled, each coil receives data for a fully sampled image. This data can be used to reconstruct a set of C images from C coils. A simple way to a get a descent reconstruction from these images is to compute the square root of the sum-of-squares of each image as follows:

$$I = \sqrt{\sum_{c=1}^{C} (I^{(c)})^2}, \tag{4}$$

where $I^{(c)}$ is the image of the c^{th} coil.

This is an extremely computationally efficient method for combining the information from different coils. However, it does not take advantage of the new information acquired with multiple coils to reduce the number of data samples required. We will now discuss methods that do.

3.2 SMASH

Let $\Delta k_{pe}/2$ be the sampling frequency in y required to satisfy Nyquist's theorem. With SMASH [1, 2], the sampling frequency is set to Δk_{pe}, twice the sampling distance required. The SMASH reconstruction algorithm will then attempt to synthesize the missing k-space coordinates.

SMASH makes two assumptions about the coil sensitivity maps:

1. $\sum_{c=1}^{C} \rho^{(c)}(x,y)$ is constant in y, and
2. there exists a known $a \in \mathbb{C}^C$ such that $\sum_{c=1}^{C} a_c\, \rho^{(c)}(x,y) \approx \exp(i\, \Delta k_{pe}\, y/2)$.

With assumption 1,

$$\begin{aligned} \sum_{c=1}^{C} s^{(c)}(t) &= \iiint_{-\infty}^{\infty} M(r) \exp(-i2\pi r \cdot k(t))\; dV \\ &= \mathcal{F}(k(t)). \end{aligned}$$

That is, by summing the signals from all coils, we attain the Fourier value of the image at location $k(t)$.

With assumption 2,

$$\begin{aligned} \sum_{c=1}^{C} a_c\, s^{(c)}(t) &= \iiint_{-\infty}^{\infty} M(r) \exp(-i2\pi r \cdot (k_x(t), k_y(t) + \Delta k_{pe}/2))\; dV \\ &= \mathcal{F}(M)\,(k_x(t), k_y(t) + \Delta k_{pe}/2). \end{aligned}$$

That is, by summing the signals with the coefficients from a, we get the value of the missing Fourier value!

In the original SMASH, the coils were specially manufactured to satisfy the assumptions required. The results, though ground breaking, retained a significant amount of aliasing. Moreover, it is laborious and restrictive to make coils in such a particular way and require them to be placed in exact locations. This led to a version of SMASH that includes an auto-calibration element.

3.3 Auto-SMASH

The goal of auto-SMASH [3], rather than determining the Fourier values of M is to estimate the Fourier values of $\rho^{(c)}M$ for all $c \in \{1, 2, \ldots, C\}$. The assumptions of SMASH are eliminated and the following assumption is made:

- For each coil c, there exists $a^{(c)} \in \mathbb{C}^{C\times 2}$ such that

$$\sum_{\nu=1}^{C} a_{\nu,1}^{(c)}\, s_{k_x,k_y+\Delta k_{pe}/2} + a_{\nu,2}^{(c)}\, s_{k_x,k_y-\Delta k_{pe}/2} = \mathcal{F}\{\rho^{(c)}M\}\,(k_x, k_y)\,,$$

where s_{k_x,k_y} is the signal at time corresponding to Fourier domain coordinate (k_x, k_y).

Instead of assuming prior knowledge of the values of a, auto-SMASH determines the values of a from the data itself. Every other line of k-space is acquired except for the center, where three consecutive lines are acquired. This additional row is called the auto-calibration row. One forms a linear system in a with these three lines and solves it.

Once a is determined, the missing Fourier values for each coil are synthesized. The coils are then combined into a single image, perhaps by using (4).

3.4 GRAPPA

In reality, there is nothing magical about the y direction. We could just as easily use the x direction with auto-SMASH, or the diagonal direction. And this is the idea behind GRAPPA [4]. Rather than just using the points above and below the missing k-space coordinate, all points in a neighborhood are used. GRAPPA assumes that for each coil, there exists a set of coefficients $a^{(c)} \in \mathbb{C}^{C\times K}$ such that

$$\sum_{\kappa_x=-\lfloor K/2\rfloor}^{\lfloor K/2\rfloor}\ \sum_{\kappa_y=-\lfloor K/2\rfloor}^{\lfloor K/2\rfloor}\ \sum_{\nu=1}^{C} a_{\nu,\kappa}^{(c)}\, s(k_x+\kappa_x, k_y+\kappa_y) = \mathcal{F}\{\rho^{(c)}\, M\}(k_x, k_y)$$

Again, a set of auto-calibration data is recorded. Instead of measuring a single additional line, though, an entire region surrounding the 0 frequency is measured. This region is called the *auto-calibration region*. Like auto-SMASH, a linear system in a is formulated with the auto-calibration region and is used to determine the coefficients. Once determined, the missing Fourier values are synthesized.

3.5 Model-Based Reconstruction

Model-based reconstruction assumes something different entirely [5]. The assumption is that the coil sensitivity maps $\{\rho^{(1)}, \rho^{(2)}, \ldots, \rho^{(C)}\}$ are known. If known, then we can model the MRI system for the c^{th} coil with the following equation:

$$b^{(c)} = D\, F\, \text{diag}(\rho^{(c)})\, M + n^{(c)}.$$

Here, M is a vector that represents the image we would like to reconstruct. It is the column-extension of the two-dimensional image. That is, it is the concatenation of the first column with the second, and then the third, etc. The vector ρ is the sampled sensitivity map, F is the two-dimensional DFT matrix, and D is a diagonal matrix that isolates which Fourier domain values were collected. The i^{th} row of D corresponds to the i^{th} Fourier sample. Each row has exactly 1 non-zero value in it; the j^{th} column has a value of 1, which means that the i^{th} row corresponds to the j^{th} Fourier domain coordinate. The vector $n^{(c)}$ represents the noise in the MR system, and it is dominated by Gaussian complex thermal noise.

The equations of all coils can be combined into the following:

$$\underbrace{\begin{bmatrix} b^{(1)} \\ b^{(2)} \\ \vdots \\ b^{(C)} \end{bmatrix}}_{\boldsymbol{b}} = \underbrace{\begin{bmatrix} D & & & \\ & D & & \\ & & \vdots & \\ & & & D \end{bmatrix} \begin{bmatrix} F & & & \\ & F & & \\ & & \vdots & \\ & & & F \end{bmatrix} \begin{bmatrix} \text{diag}(\rho^{(1)}) \\ \text{diag}(\rho^{(2)}) \\ \vdots \\ \text{diag}(\rho^{(C)}) \end{bmatrix}}_{\boldsymbol{A}} M + \underbrace{\begin{bmatrix} n^{(1)} \\ n^{(2)} \\ \vdots \\ n^{(C)} \end{bmatrix}}_{\boldsymbol{n}} .$$

This is a linear system plus noise: $\boldsymbol{A}M = \boldsymbol{b} + \boldsymbol{n}$. We want to find M that minimizes the difference between $\boldsymbol{A}M$ and $\boldsymbol{b}$.

If enough data points are collected, then $\boldsymbol{A}$ is tall and skinny. Barring extreme symmetry in the sensitivity maps, an estimate of M can be found using the pseudo-inverse of $\boldsymbol{A}$ according to $\tilde{M} = \boldsymbol{A}^{\dagger}\boldsymbol{b}$ (as described in Appendix A). However, this would require constructing the matrix $\boldsymbol{A}$. This would consume a great deal of memory and be inefficient. (The Fast Fourier Transform is an efficient algorithm for calculating the DFT, and packages like Fastest Fourier Transform in the West take further advantage of hardware specifics [6].) Instead of constructing $\boldsymbol{A}$ directly, then, one implements a function that accepts a vector and returns the multiplication with $\boldsymbol{A}$. The problem can then be solved with the LSQR algorithm [7] or the LSMR algorithm [8]. To do so requires an implementation of both $\boldsymbol{A}$ and its Hermitian transpose. The following is an implementation of $\boldsymbol{A}$ in Matlab along with a call to lsqr.

```
[ Ny Nx nCoils ] = size( kData );  % kData is the k-space data.
                                   % Any point that isn't collected is set to $0$
nb = sum( kData(:) ~= 0 );  % The number of elements in the data array
% It is assumed that a matrix rho is defined of size [ Ny Nx nCoils ] that
% indicates the coil sensitivity map at each location.

function out = applyA( in, type )
  if type == 'transp'
    % Code for Hermitian transpose goes here
  else
    M = reshape( in, [ Ny Nx ] );
    out = zeros( nb, 1 );
    nDataPerCoil = nb / nCoils;
    for coil = 1 : nCoils
      tmp = 1/sqrt(Ny*Nx) * fft2( rho(:,:,coil) .* M );
      out( (coil-1) * nDataPerCoil + 1 : coil * nDataPerCoil ) = ...
        tmp( kData ~= 0 );
    end
    out = out(:);  % reformat out as a column vector
  end
end

M0 = zeros( Ny, Nx );
M_hat = lsqr( applyA, b, [], [], [], [], M0(:) );
M_hat = reshape( M_hat, [ Ny Nx ] )
```

By exploiting the different measurements by the different coils, model-based reconstruction permits accurate imaging with less samples than are required by Nyquist's theorem.

4 Compressed Sensing

Instead of relying on multiple coils, perhaps we can rely on known properties of the final image to reduce the number of data points required. This is the idea behind compressed sensing (sometimes called compressive sampling).

Compressed sensing is the name of the following optimization problem:

$$\begin{aligned} \underset{x}{\text{minimize}} \;\; & \|x\|_0 \\ \text{subject to} \;\; & (1/2)\,\|Ax - b\|_2^2 < \sigma \end{aligned} \tag{5}$$

for some $\sigma > 0$. In this problem, A is a matrix, and x and b are vectors. The function $\|\cdot\|_0$ is called the L0 norm; $\|x\|_0$ is the number of non-zero elements in x.

This is an NP-hard combinatorial optimization problem. We (humans) do not have efficient solvers for these types of problems. As a hack, one might hope that we could replace the L0 norm with the L1 norm (its convex envelope) as follows:

$$\begin{aligned} &\underset{x}{\text{minimize}} && \|x\|_1 \\ &\text{subject to} && (1/2)\|Ax - b\|_2^2 < \sigma. \end{aligned} \tag{6}$$

This is a convex optimization problem and we have efficient algorithms that are guaranteed to solve it. Astoundingly, if A is nice enough, then solving this optimization problem probably yields the answer to that of (5)! [9]

But how do we take advantage of this in MRI? We often know that the signal we are looking for is mostly composed of zeros if we look at it in the right way. For example, Fig. 1 shows an image and its two-dimensional Haar wavelet transform. Note that most of the values of the transform are 0. Moreover, we can increase the number of zeros in the transform by applying the wavelet transform recursively; the result of doing so is shown in Fig. 2.

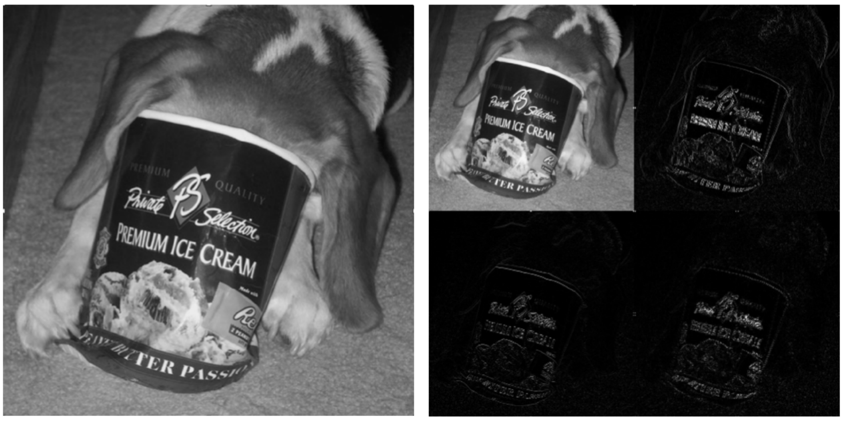

Figure 1: (Left) Original image, (Right) 2D Haar wavelet transform of image

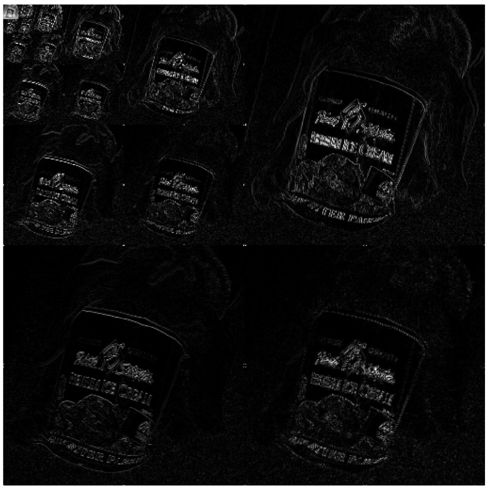

Figure 2: The result of applying the Haar wavelet transform recursively to the upper left bin.

This is the pattern for all natural images. The wavelet transform of almost all natural images has the property that most of its values are very close to 0. Equivalently, we state that the Wavelet transform of an image is sparse. We can use this fact to reconstruct MR imagery [10].

There are several variations of MR image reconstruction with sparsity. If we let $y = WM$, where W represents the matrix of the wavelet transform (a linear transformation), then we can find y by solving

$$\begin{aligned} \text{minimize} \ \ & \|y\|_1 \\ \text{subject to} \ \ & (1/2)\|\boldsymbol{A}W^{-1}y - \boldsymbol{b}\|_2^2 < \sigma \end{aligned} \tag{7}$$

for some value of $\sigma > 0$ based on the amount of noise present. This is the compressed sensing problem and algorithms exist to solve it! For example, one can solve this problem with the Fast Iterative Shrinkage Threshold Algorithm. Once y is found, the image can be reconstructed with $M = Wy$.

Instead of solving (7), we will describe a method to solve the following similar problem:

$$\begin{aligned} \text{minimize} \ \ & (1/2)\|\boldsymbol{A}W^{-1}y - \boldsymbol{b}\|_2^2 \\ \text{subject to} \ \ & \|y\|_0 < \mathcal{C}, \end{aligned}$$

where $\mathcal{C} > 0$ specifies the number of non-zero elements in the Wavelet transform that will be acceptable.

A solution, though probably not the optimal solution, can be found to this problem by using the projected gradient descent algorithm. A Euclidean projection of a vector y onto the set $\{y : \|y\|_0 < \mathcal{C}\}$ retains the top $\mathcal{C}$ values and sets all other values to 0. Let $f(y) = (1/2)\|\boldsymbol{A}W^{-1}y - \boldsymbol{b}\|_2^2$. Then

$$\nabla f(y) = W\boldsymbol{A}^H\boldsymbol{A}W^{-1}y - W\boldsymbol{A}^Hb.$$

Here, we are assuming that W is orthogonal. Armed with these capabilities, the projected gradient algorithm is as shown in Alg 1.

Algorithm 1: Compressed Sensing with Projected Gradient Method

```
Initialize estimate $y^{(0)}$. Choose a step size $\mu > 0$. Set $J$ to some number large enough that the
  algorithm is likely to converge.
For $j = 1, 2, \ldots, J$
    Perform a gradient descent update: $z^{(j)} = y^{(j)} - \mu\nabla f(y^{(j)})$.
    $y^{(j+1)} = \Pi_{\|\cdot\|_0 < \mathcal{C}}\left(z^{(j)}\right)$. That is, form $y^{(j+1)}$ by setting the lower values of $z^{(j)}$ to $0$.
End For
```

Again, once y is found, the image can be reconstructed with $M = Wy$.

The key to compressed sensing is random sampling. Rather than sampling with any regular pattern, in MR image reconstruction, a variable density sampling pattern is required. The density of sampling should be high near 0 frequency and lower farther away.

A Least-squares

Consider the linear system $b = \boldsymbol{Ax} + \boldsymbol{n}$ where $\boldsymbol{n}$ is a vector of Gaussian noise. The maximum likelihood estimate of $\boldsymbol{x}$ is the one that minimizes $\|\boldsymbol{Ax} - \boldsymbol{b}\|_2$ [11]. Note that the value of $\boldsymbol{x}$ that minimizes $\|\boldsymbol{Ax} - \boldsymbol{b}\|_2$ is the same one that minimizes $(1/2)\,\|\boldsymbol{Ax} - \boldsymbol{b}\|_2^2$.

Let $f(\boldsymbol{x}) = \|\boldsymbol{Ax} - \boldsymbol{b}\|_2^2$. This is a differentiable function; the minimum occurs when $\nabla f(\boldsymbol{x}) = 0$ (here ∇ represents the gradient). Recall that $\|u\|_2^2 = u^H u$, where ^{H} represents conjugate transpose. (The H is used to denote the Hermitian, which means the conjugate transpose for complex matrices.) Then

$$f(x) = (1/2)\,(\boldsymbol{Ax} - \boldsymbol{b})^H(\boldsymbol{Ax} - \boldsymbol{b}) = (1/2)\left(\boldsymbol{x}^H\boldsymbol{A}^H\boldsymbol{Ax} - 2\boldsymbol{b}^H\boldsymbol{Ax} + \boldsymbol{b}^H\boldsymbol{b}\right).$$

Therefore, $\nabla f(\boldsymbol{x}) = \boldsymbol{A}^H\boldsymbol{Ax} - \boldsymbol{A}^H\boldsymbol{b}$. Setting the gradient to 0 yields

$$\boldsymbol{x} = \underbrace{\left(\boldsymbol{A}^H\boldsymbol{A}\right)^{-1}\boldsymbol{A}^H}_{\boldsymbol{A}^\dagger}\boldsymbol{b}. \tag{8}$$

Though there is an analytic expression for the pseudo-inverse of a matrix, it is numerically unstable to calculate it this way. There are far more numerically stable algorithms [12]. Luckily, most computer packages already have implementations of the pseudo-inverse implemented. For example, in Matlab, one can determine the estimate of x with the following command.

```
x = A \ b;
```

B Final Remarks

There are many more algorithms for parallel imaging and compressed sensing both in literature and in practice. The field is rich with accomplishment and ripe with opportunity for further progress. I hope that you have enjoyed this introduction. Please realize that it is *just* and introduction. There is much more work to be done. Hopefully, you're now a bit better prepared to help out.

References

[1] Daniel K Sodickson and Warren J Manning. Simultaneous acquisition of spatial harmonics (smash): fast imaging with radiofrequency coil arrays. *Magnetic resonance in medicine*, 38(4):591–603, 1997.

[2] Daniel Kevin Sodickson. Simultaneous acquisition of spatial harmonics (smash): ultra-fast imaging with radiofrequency coil arrays, June 8 1999. US Patent 5,910,728.

[3] Peter M Jakob, Mark A Grisowld, Robert R Edelman, and Daniel K Sodickson. Auto-smash: a self-calibrating technique for smash imaging. *Magnetic Resonance Materials in Physics, Biology and Medicine*, 7(1):42–54, 1998.

[4] Mark A Griswold, Peter M Jakob, Robin M Heidemann, Mathias Nittka, Vladimir Jellus, Jianmin Wang, Berthold Kiefer, and Axel Haase. Generalized autocalibrating partially parallel acquisitions (GRAPPA). *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 47(6):1202–1210, 2002.

[5] Jeffrey A Fessler. Model-based image reconstruction for MRI. *IEEE Signal Processing Magazine*, 27(4):81–89, 2010.

[6] Matteo Frigo and Steven G Johnson. FFTW: An adaptive software architecture for the FFT. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181)*, volume 3, pages 1381–1384. IEEE, 1998.

[7] Christopher C Paige and Michael A Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software (TOMS)*, 8(1):43–71, 1982.

[8] David Chin-Lung Fong and Michael Saunders. LSMR: An iterative algorithm for sparse least-squares problems. *SIAM Journal on Scientific Computing*, 33(5):2950–2971, 2011.

[9] Emmanuel J Candes, Justin K Romberg, and Terence Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 59(8):1207–1223, 2006.

[10] Michael Lustig, David Donoho, and John M Pauly. Sparse MRI: The application of compressed sensing for rapid MR imaging. *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 58(6):1182–1195, 2007.

[11] Christopher M Bishop. *Pattern recognition and machine learning*. Springer Science+ Business Media, 2006.

[12] Lloyd N Trefethen and David Bau III. *Numerical linear algebra*, volume 50. Siam, 1997.

Problem: In this problem, you will reconstruct MR imagery using GRAPPA and compressed sensing. The data is comes from `mridata.org`.

1) `grappaData2.mat` is a dataset where there is a calibration region with every other line retained. The missing data has values of exactly 0. Reconstruct the imagery by filling in the missing data using GRAPPA and then combining the resulting images using the square-root-sum-of-squares algorithm.

2) `grappaData3.mat` is a dataset where there is a calibration region with every third line retained. The missing data has values of exactly 0. Reconstruct the imagery by filling in the missing data using GRAPPA and then combining the resulting images using the square-root-sum-of-squares algorithm.

3) `csData.mat` is a single coil variable density dataset. Reconstruct the image using a compressed sensing algorithm. Try varying levels of coefficients to set to 0 in the Wavelet domain to see what happens. For this problem, you may wish to use the `wtHaar2` and `iwtHaar2` functions with a split of `split=zeros(4); split(1)=1;`. You can find those functions in the dworkLib github repository here: `https://github.com/ndwork/dworkLib`. You can view the wavelet transform of the image using the `wavShow` function in the same repository (though that function is dependent on other functions in the repository as well).