Bioengineering Imaging Track Qualifying Exam Study Book

Markus Foote and Blake Zimmerman

August 26, 2017

Contents

1. Affine Transform

1.1 General Transformation Concept

An *affine transform* is the combination of a linear transformation with a translation. In a linear algebra contest, a point in space is a column vector of individual values along each dimension, i.e. $\begin{bmatrix} x \\ y \end{bmatrix}$ or $\begin{bmatrix} x \\ y \\ z \end{bmatrix}$ in 2 or 3 dimensions, respectively. This location is noted simply as $\vec{x}$.

The affine transform can thus be written as a matrix-vector multiplication, followed by vector addition:

$$\vec{y} = \mathbf{A}\vec{x} + \vec{b} \tag{1.1}$$

The linear transformation matrix $\mathbf{A}$ has some important special cases:

Scaling This transformation scales each direction of a vector space by the corresponding c value:

$$\mathbf{A} = \begin{bmatrix} c_x & 0 \\ 0 & c_y \end{bmatrix} \text{ or } \begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & c_z \end{bmatrix}$$

Rotation This transformation rotates by an angle θ about the origin in 2 dimensions:

$$\mathbf{A} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

Translation This only translates all the points by a constant amount:

$$\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \text{ and } b = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

Skew

Others?

1.2 The Imperfect World

These transformations are all well and good, but with them so far we can only take points and apply some fun tricks to move them around in cute ways. What if there were pairs of points with *known* (or *assumed*) correspondence, and we want to find an affine transform that makes the point pairs match?

Using notation:

$\vec{x}_i$	source point(s)	(1.2)
$\vec{y}_i$	target point(s)	(1.3)
n	total number of point pairs	(1.4)
$\mathbf{A}$	linear transform matrix	(1.5)
T	translation vector	(1.6)

In a perfect world, all the points match exactly after being transformed, such that

$$\vec{y}_i = \mathbf{A}\vec{x}_i + T \tag{1.7}$$

is satisfied. This linear system of equations could be easily solved with some linear algebra.

However, this world is not perfect. Due to noise, human error, systematic error, and spilling coffee on samples before *carefully* wiping it up, there will be noise and the points will not exactly match. There will be some error. We must account for this error.

We want to estimate the best transformation given some number of point pairs that will not all match exactly. Thus, we should minimize the error in the matching using a euclidean distance for the error, and squared error so that this function is convex (which is important when very close to zero error):

$$\min_{\mathbf{A},T} \sum_i \left\| (\mathbf{A}\vec{x}_i + T) - \vec{y}_i \right\|^2 \tag{1.8}$$

1.3 Estimating a Solution

First, if we are given $\mathbf{A}$, then

$$\delta T = 2\sum_i^n (\mathbf{A}\vec{x}_i + T - \vec{y}_i) = 0 \tag{1.9}$$

The variation of T is set to zero because the first derivative of a function is zero at a minimum. Solving for T:

$$T = \frac{1}{n}\sum_{i=1}^{N} (\vec{y}_i - \mathbf{A}\vec{x}_i) \tag{1.10}$$

This translation is simply the difference of the centroids of the source and target points. Let us define

$$\tilde{y}_i = \vec{y}_i - \vec{c}_y \tag{1.11}$$

$$\tilde{x}_i = \vec{x}_i - \vec{c}_x \tag{1.12}$$

where c_x and c_y are the centroids of each point set. This modification of the problem implicitly removes T since both point sets will have the same centroid (the origin).

Assuming the points are centered the problem

$$\min_{\mathbf{A}} \sum_i \left\| \mathbf{A}\left(\vec{x}_i - \frac{1}{n}\sum_j \vec{x}_j\right) - \left(\vec{y}_i - \frac{1}{n}\sum_j \vec{y}_j\right) \right\|^2 \tag{1.13}$$

simplifies to

$$E = \min_{\mathbf{A}} \sum_i \left\| \mathbf{A}\tilde{x}_i - \tilde{y}_i \right\|^2 \tag{1.14}$$

Similar to the procedure performed to find and remove T, we now find the variation of $\mathbf{A}$ and set it equal to zero:

$$\frac{\partial}{\partial \mathbf{A}} = 2\sum_i (\mathbf{A}\tilde{x}_i - \tilde{y}_i)\ \tilde{x}_i^T = 0 \tag{1.15}$$

Then solving for $\mathbf{A}$:

$$\mathbf{A} = \left(\sum_i \tilde{y}_i \tilde{x}_i^T\right)\left(\sum_i \tilde{x}_i \tilde{x}_i^T\right)^{-1} \tag{1.16}$$

A special note here about the matrix $\sum_i \tilde{x}_i \tilde{x}_i^T$: A minimum of 3 non-co-linear points in 2D (or 4 non-coplanar points in 3D) are required for this matrix to be invertible.

2. Magnetic Resonance Imaging

2.1 Overview

Magnetic Resonance Imaging (**MRI**) is an imaging modality that expoits the phenomenon of Nuclear Magnetic Resonance to record information about the local chemical environment of specific atoms within a body. Commonly, these atoms are the hydrogen nuclei, but MRI is possible with any nuclei with a non-integer nuclear spin. Generally, the proceedure for MRI is summarized in the following steps, where only step 5 is specific to the Imaging modality over an NMR experiment:

1. Nuclei align with an applied, strong magnetic field $\vec{B}_0$.
2. Radio Frequency energy is applied to flip the nuclear magnetization off-axis from $\vec{B}_0$.
3. Interaction between $\vec{B}_0$ and nuclear magnetizations causes magnetizations to ***precess***.
4. Induced current from precessing magnetization is observed in an RF coil.
5. Speed of precession is spatially varied by a spatially-varying magnetic field gradient.

2.2 Magnetism and Nuclear Magnetic Resonance

Magnetism is an inherent property of matter which causes materials to interact with external magnetic fields to generate their own magnetic field. This phenomenon is characterized by the relation

$$\vec{H} = \chi \vec{B} \tag{2.1}$$

where $\vec{H}$ is the generated field, $\vec{B}$ is the applied field, and χ is the *magnetic susceptibility* of the matter. Matter is classified into three main categories based on χ:

Paramagnetic matter has *positive* χ, thus $\vec{H}$ points in the same direction as $\vec{B}$. **Ex:** Aluminum

Diamagnetic matter has *negative* χ, thus $\vec{H}$ points in the opposite direction as $\vec{B}$. **Ex:** Water

Ferromagnetic matter has *large* χ. **Ex:** Iron

Magnetism originates from the electrons of an atom, but Nuclear Magnetic Resonance originates from the nucleus and is still subject to electronic magnetism, or *chemical shift* phenomena.

Nuclear Magnetism arises from nuclei of atoms with odd-even pairing of neutrons and protons, resulting in nuclei with non-integer spin. E.g. ^{1}H with spin $I = 1/2$. Nuclear spin in motion gives rise to an angular momentum of the spin:

$$J = \hbar I = \frac{h}{2\pi} I \tag{2.2}$$

where J is the resulting angular momenta, h is Planck's constant, and $\hbar$ is the reduced Planck constant. The magnetic moment of matter is also determined by

$$\mu = \gamma J = \gamma \hbar I \tag{2.3}$$

where μ is the magnetic moment and γ is the *gyromagnetic ratio* that is specific to the nucleus in question.

Bulk magnetization (the collective net magnetization from a region of matter) is dependent upon the relative number of individual moments in the 'up' (parallel to $\vec{B}_0$) and 'down' (anti-parallel to $\vec{B}_0$) states:

$$\vec{m} = \mu(\Delta N) \tag{2.4}$$

where N is the number of magnetic moments in the up and down states such that ΔN is the net difference of moments in the anti-parallel direction. N follows a Boltzmann distribution based on the energy state, E, of the nuclei:

$$N \propto e^{-\frac{E}{kT}} \tag{2.5}$$

where k is the Boltzmann constant and T is the absolute temperature. The energy state of a magnetic moment is

$$E(\mu) = -\mu B_0 = -\gamma\hbar m B_0 \tag{2.6}$$

where $m = \{I \ldots {}^3\!/_2, {}^1\!/_2, -{}^1\!/_2, -{}^3\!/_2, \ldots - I\}$ are the permissible states for the spin I. Again assuming ^{1}H nuclei, the two energy states are

$$E\,(\text{up}) = -\gamma\hbar\,({}^1\!/_2)\,B_0 \tag{2.7}$$

$$E\,(\text{down}) = +\gamma\hbar\,({}^1\!/_2)\,B_0. \tag{2.8}$$

The ratio of spin alignment can then be calculated:

$$\frac{N\,(\text{up})}{N\,(\text{down})} = e^{+\frac{\Delta E}{kT}} \tag{2.9}$$

where $\Delta E = E\,(\text{down}) - E\,(\text{up})$. If we then suppose that there exist a total of N nuclei such that $N\,(\text{up}) + N\,(\text{down}) = N$, then from this and (2.9) we can solve for ΔN and thus M:

$$M = \mu\Delta N = \frac{N\gamma^2\hbar^2 B_0}{4kT} \tag{2.10}$$

This equation can be massaged to match the form of (2.1):

$$M = \chi_0 B_0 \qquad\qquad \chi_0 = \frac{N\gamma^2\hbar^2}{4kT} \tag{2.11}$$

where χ_0 is a matter-specific term that describes the resulting magnetic field from an applied field B_0. This χ_0 is very small, and results in a very low effective signal from a body of matter. However, there are engineering decisions in each of these terms that helps raise the signal:

$\mathbf{B_0}$ Make it big, but not too big. Like 1.5 T at least, probably 7 T is awesome.

N Effectively this is the voxel size. Cubic power. ${}^1\!/_2$ resolution $\rightarrow$ 8x larger voxel $\rightarrow$ 64x signal.

T Very limited range if you want the subject to be alive, but colder is better.

γ Pick nuclei with high γ, and isotope with high abundance. See Table 2.1.

Table 2.1: Relevant Properties of Some Common Nuclei

Nuclei	I	$\frac{\gamma}{2\pi}$ (MHz/T)	nat. abundance (%)
^{1}H	${}^1\!/_2$	**42.58**	99.98
^{13}C	${}^1\!/_2$	10.71	1.11
^{19}F	${}^1\!/_2$	40.05	100
^{31}P	${}^1\!/_2$	17.23	100
^{23}Na	${}^3\!/_2$	11.26	100

2.3 Spin Motion and the NMR Signal

Precession of the nuclear magnetization, or 'spin' is described by

$$\frac{d\vec{m}}{dt} = \gamma \vec{m} \times \vec{B}. \tag{2.12}$$

The resulting motion is analogous to a spinning gyroscope: the rotation does not change, but the *axis of rotation* changes. The precession of earth leads to changes in the classification of the north star, while the typical spinning of earth manifests as day/night.

There are four important cases of spin motion in the context of MRI/NMR:

1. First is the case when $\vec{m}$ is aligned with $\vec{B}$, for example $\vec{m} = m_0\ \hat{z}$ and $\vec{B} = B_0\ \hat{z}$. This results in the derivative term being zero from properties of the vector cross product. This case is stable (see Fig. 2.1a).

2. The second case is substantially more interesting. If we let

   $$\vec{m} = m_0\ \hat{x} \tag{2.13}$$

   and

   $$\vec{B} = B_0\ \hat{z}, \tag{2.14}$$

   then

   $$\frac{d\vec{m}}{dt} = -m_0 B_0\ \hat{y}\ . \tag{2.15}$$

   This results in the infinitesimal motion of the magnetization to move in the $-\hat{y}$ direction. As time progresses, the magnetization precesses *about* $-\hat{z}$ at the rate $\omega = -\gamma B_0$, known as the *Larmor* frequency (see Fig. 2.1b).

3. Similar to case 2, the precession about $-\hat{z}$ is stable at any angle α (see Fig. 2.1c).

4. Finally, the case of 'flipping' the magnetization from alignment with $\hat{z}$ to alignment with $\hat{x}$. Starting with

   $$\vec{m} = m_0\ \hat{z} \tag{2.16}$$

   the application of an infinitely short magnetic field pointing in the $-\hat{y}$ direction

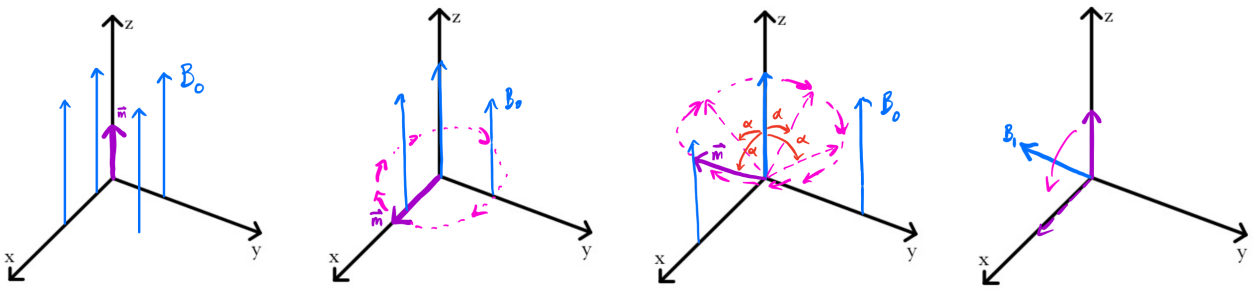
   $$\vec{B} = -\delta(t) B_1 \hat{y} \tag{2.17}$$

   results in changing the direction of $\vec{m}$ towards the $\hat{x}$ axis. Realistically, this applied field cannot be infinitely short, so after every bit of time, the $\vec{B}_1$ field must be updated to still be perpendicular to $\vec{m}$. This results in a field applied according to

   $$\vec{B} = B_1 \big(\cos(\omega_0 t) - \sin(\omega_0 t) \big) \tag{2.18}$$

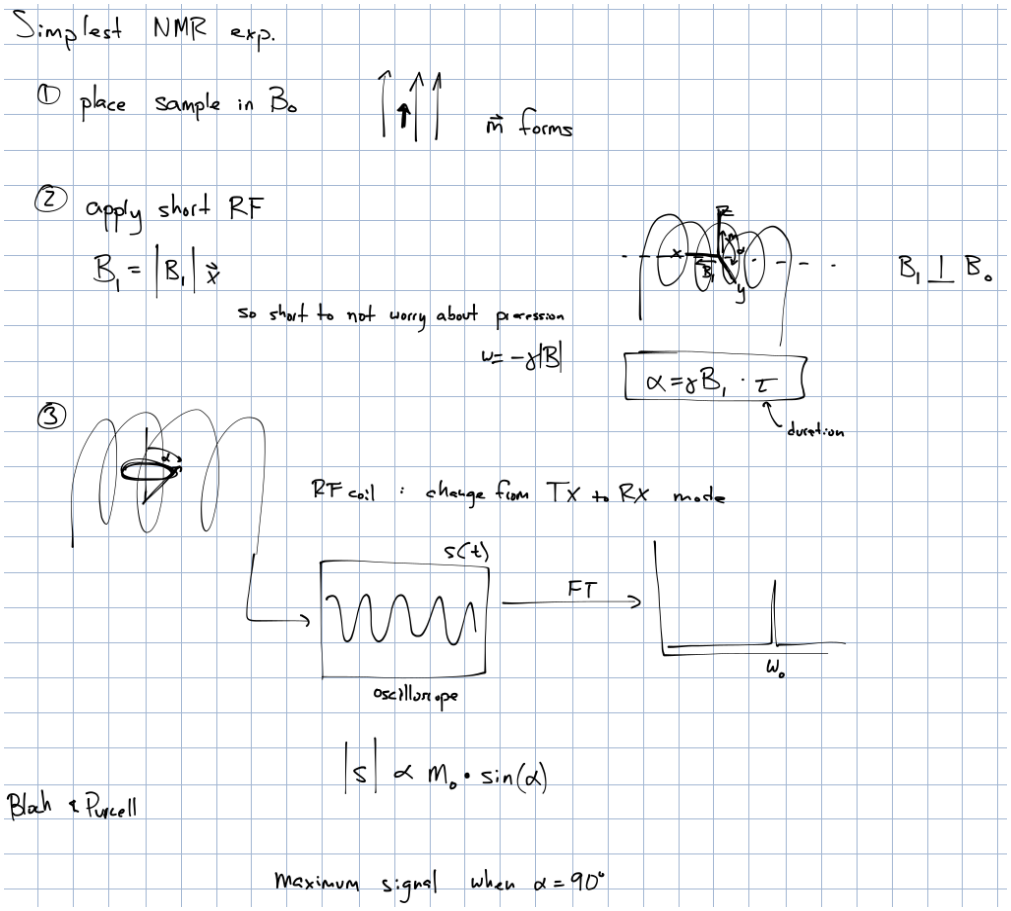
   giving a clockwise spin of the applied magnetic field (precession about the $-\hat{z}$ axis). This results in $\vec{m}$ precessing about the $\hat{z}$ axis while being progressively 'flipped' to the transverse ($\hat{x}$-$\hat{y}$) plane. The overall 'flip' procedure is parameterized by α, the resulting angle that $\vec{m}$ makes with the $+\hat{z}$ axis (see Fig. 2.1d).

The fundamentals presented thus far are sufficient to describe the most basic NMR experiment:



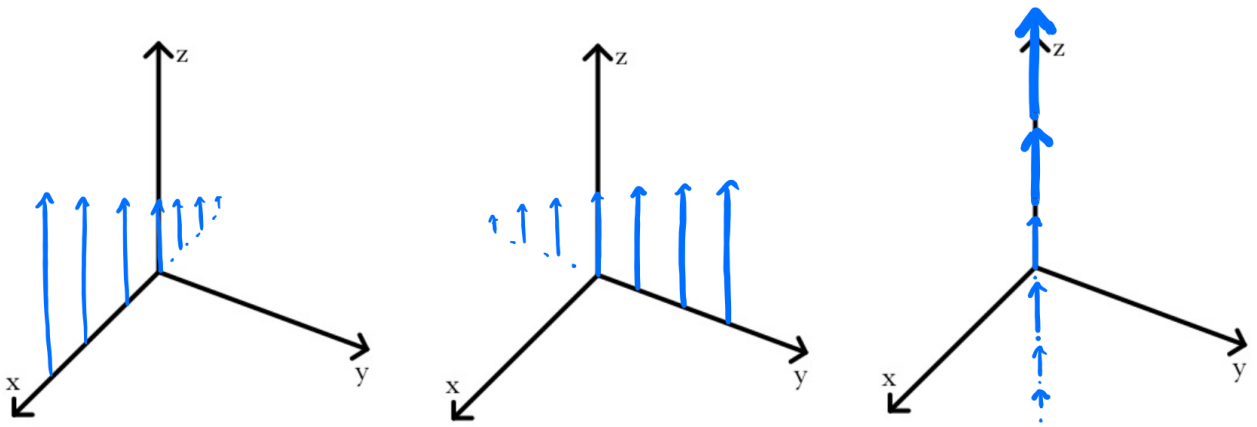
(a) Stable Magnetization (b) Spin Motion (c) Precession at an angle (d) Excitation

Figure 2.1: Canonical Cases of Spin Motion



2.4 Spatial Encoding

Magnetic Resonance *Imaging* is made possible by encoding spatial information in the NMR signal. This is accomplished by making B a linear function of space using *gradient fields*. These magnetic gradient fields are always parallel to the main B_0 field. By changing the magnitude of the magnetic field, the spins precess faster or slower to form a frequency histogram, which is a projection of the object. Example gradient fields are shown in Fig. 2.2.



(a) G_x (b) G_y (c) G_z

Figure 2.2: Gradient Fields

The Larmor frequency is thus dependent on the location of the spin:

$$\vec{G} = \begin{bmatrix} G_x \\ G_y \\ G_z \end{bmatrix} \tag{2.19}$$

$$\vec{r} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{2.20}$$

$$B = B_0 + \vec{G} \cdot \vec{r} \tag{2.21}$$

$$\omega(\vec{r}) = -\gamma B = -\gamma \left(B_0 + \vec{G} \cdot \vec{r} \right) \tag{2.22}$$

Using this relation and assuming that $m(x, y, z) = m(\vec{r})$ is the proton density image, each magnetization in space will precess at the frequency corresponding to its location by 2.22. The signal radiated by a single point is

$$m(\vec{r}) e^{i\omega(\vec{r})t}. \tag{2.23}$$

Moving into the rotating reference frame, the effect of the static B_0 field is eliminated, and we can consider only the change in frequency from the static B_0 that is imbued by the gradient fields:

$$\Delta\omega(\vec{r}) = -\gamma \vec{G} \cdot \vec{r}. \tag{2.24}$$

The signal observed in this rotating frame from a single point is

$$s(t) = m(\vec{r}) e^{i\Delta\omega(\vec{r})t}. \tag{2.25}$$

The total signal recorded by a coil surrounding the body is found by integrating over the entire volume:

$$s(t) = \iiint_{\Omega} m(\vec{r}) e^{-i\gamma \vec{G} \cdot \vec{r} t} d\vec{r} \tag{2.26}$$

A useful substitution is made by introducing the variable k:

$$\vec{k} = \frac{\gamma}{2\pi} \vec{G} t \tag{2.27}$$

$$s(\vec{k}) = \iiint_{\Omega} m(\vec{r}) e^{-i2\pi \vec{k} \cdot \vec{r}} d\vec{r} \tag{2.28}$$

Close inspection of this equation reveals that it is simply the forward Fourier Transform of the function of spatial magnetization $m(\vec{r})$. The original magnetization image can be easily obtained by the *inverse* Fourier Transform of the acquired signal.

2.5 Slice Selection

Spatial encoding of the Larmor frequency via gradient fields also enables selective excitation. The target nuclei within the 2D plane (or slab) we aim to excite are provisioned a common resonant frequency by applying a gradient in the direction orthogonal to that plane. The target slice can then be excited by delivering radiofrequency energy at the appropriate frequency via a modulated radiofrequency pulse, especially a sinc-modulated sinusoid.

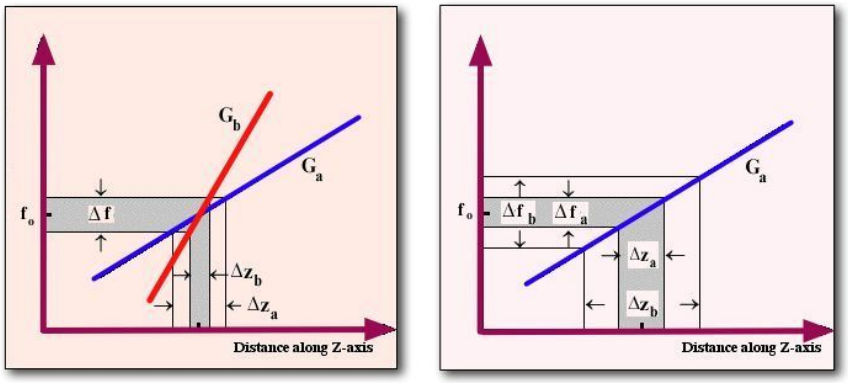


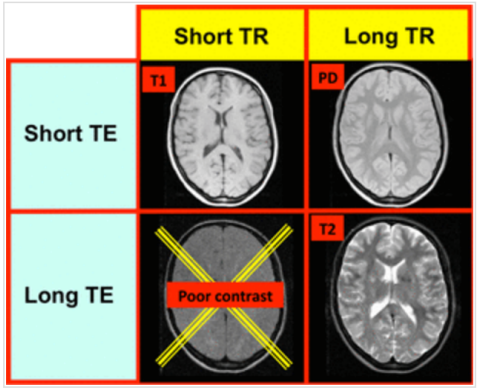
Figure 2.3: Diagram indicating effect of increased gradient field strength on slice selection.

2.6 MRI Contrast Mechanisms

The general signal equation behind MRI signal assuming a 90° pulse is the following:

$$S = M_z^0(1 - e^{-T_R/T_1})e^{-T_E/T_2} \tag{2.29}$$

and depending on the T_E and T_R times, the contrast of the image can be changed. The following is a table of the generic times for different contrasts:



2.6.1 Spin Lattice Relaxation - T1

$$\frac{dm_z}{dt} = \frac{m_0 - m_z}{T_1} \tag{2.30}$$

$$m_z(t) = \underbrace{m_{z0}\cos(\alpha)}_{\text{initial condition}} e^{\frac{-t}{T_1}} + \underbrace{m_0}_{\text{final condition}} \left(1 - e^{\frac{-t}{T_1}}\right) \tag{2.31}$$

add graph here

2.6.2 Spin-Spin Relaxation - T2

$$T_2 << T_1 \tag{2.32}$$

$$\frac{dm_{xy}}{dt} = \frac{-1}{T_1} m_{xy} \tag{2.33}$$

$$m_{xy}(t) = \underbrace{m_{z0} \sin(\alpha)}_{\text{initial condition}} e^{\frac{-t}{T_2}} \tag{2.34}$$

add graph here

2.6.3 Dephasing - T2*

Same as T_2, just faster, due to B_0 inhomogeneity, spins dephase in xy plane.

2.6.4 Bloch Equation

The T_1 and T_2 contrast mechanisms are incorporated together in the spin motion equation to fully describe the motion of a nuclear spin through time. In pieces, the equations are the same as what was introduced above in each section.

$$\frac{d\vec{m}}{dt} = \gamma \vec{m} \times \vec{B} \tag{2.35}$$

$$\frac{dm_z}{dt} = \frac{m_0 - m_z}{T_1} = \frac{m_z - m_0}{T_1} \tag{2.36}$$

$$\frac{dm_x}{dt} = \frac{-m_x}{T_1} \tag{2.37}$$

$$\frac{dm_y}{dt} = \frac{-m_y}{T_1} \tag{2.38}$$

These equations are written together as a single vector equation:

$$\frac{d\vec{m}}{dt} = \gamma \vec{m} \times \vec{B} - \begin{bmatrix} 1/T_2 & 0 & 0 \\ 0 & 1/T_2 & 0 \\ 0 & 0 & 1/T_1 \end{bmatrix} (\vec{m} - \vec{m}_0) \tag{2.39}$$

2.6.5 Diffusion

Zeroth moment of gradients (area under curves) encodes location. Take it a step further and calculate (well, design pulse sequence that has) large First Moment of gradient, which encodes velocity.

2.6.6 FatSat

Use the fat-specific chemical shift of protons on fat to excite them (90° flip), then spoil gradient to dephase.

2.7 Steady-State Magnetization

The magnetization after many α pluses is given by the following:

$$M_z(t_{n+1}) = M_z(T_n)\ \cos(\alpha)\, e^{-T_R/T_1} \quad + \quad M_0(1 - e^{-T_R/T_1}) \tag{2.40}$$

where $M_z(T_n)$ represents the magnetization after the n pulse. If we want to reach steady state, that is when $M_z(T_{n+1}) = M_z(T_n) = M_{ss}$. If we plug this idea into 2.40:

$$M_{ss} = \frac{M_0(1 - e^{-T_R/T_1})}{1 - \cos(\alpha)e^{-T_R/T_1}} \tag{2.41}$$

We can then look at the transverse magnetization resulting from this flip:

$$M_{\perp}(TE) = M_{ss}\ \sin(\alpha)\ e^{-T_E/T_2^*} = \frac{M_0(1 - e^{-T_R/T_1})}{1 - \cos(\alpha)e^{-T_R/T_1}}\ \sin(\alpha)\ e^{-T_E/T_2^*} \tag{2.42}$$

with this we can then take the derivative with respect to α and set that equal to zero to find the Earsnt angle, or the angle that will maximize signal for a given T_R and T_1:

$$\cos(\alpha) = e^{-T_R/T_1} \tag{2.43}$$

2.8 MRI Field of View

The field of view in an MRI acquisition follows a general inverse relationship with the sampling frequency of the acquisition:

$$\Delta k_{x,y,z} = \frac{1}{FOV_{x,y,z}} \tag{2.44}$$

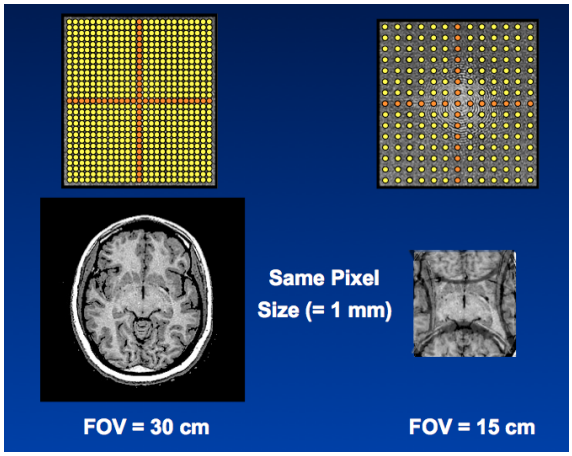
The sampling of k-space relates to FOV and the Gradient strength by

$$\Delta k_{ro,freq,x} = \frac{\gamma}{2\pi} \cdot \frac{1}{SW} \cdot G_{ro,freq,x} \tag{2.45}$$

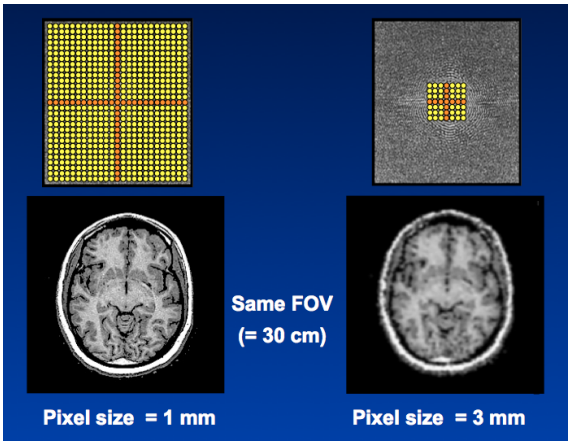
where SW is the 'sweep width'/sampling frequency of the digitizer in the MRI system. Similarly for the phase-encoding direction:

$$\Delta k_{phase,y} = \frac{\gamma}{2\pi} \cdot \tau \cdot \Delta G_{phase,y} \tag{2.46}$$

where τ is the time that the gradient is turned on before read-out, and $\Delta G_{phase,y}$ is the step-distance in the 'ladder' for each excitation. See the shamelessly stolen images that demonstrate this inverse relationship in Fig. 2.4 .



(a) Inverse relationship between spacing of data samples (Δk) and image field-of-view (FOV)



(b) Inverse relationship between pixel width ($\Delta\omega$) and range of sampled k-space frequencies (k_{FOV})

Figure 2.4: MRI FOV Relationships

2.9 Basic Pulse Sequences

The most basic form of imaging is spin echo imaging where the transverse magnetization is re-phased with a 180° flip and then read out. A basic pulse sequence for spin echo can be seen in Figure 2.5.

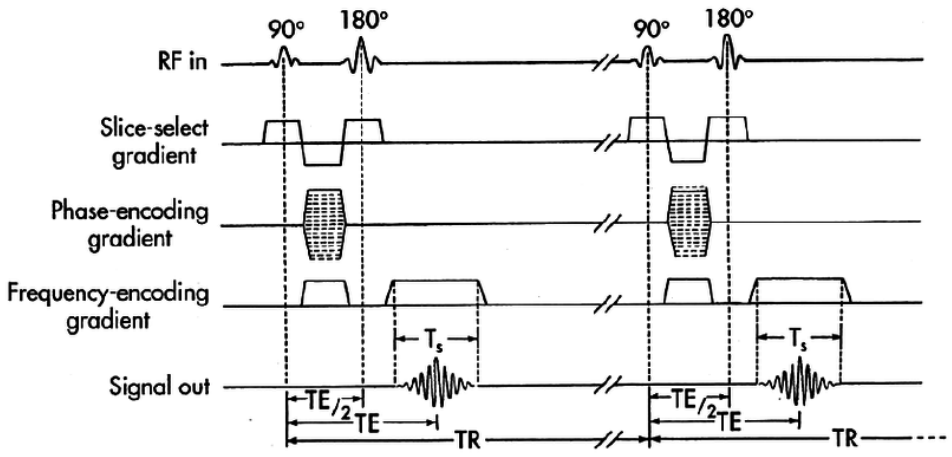


Figure 2.5: Spin Echo Pulse Sequence

One thing that is not labeled is the negative lobe in the slice-select gradient line between the 90° and 180° flip. This re-phases the slice that was excited because the slice select gradient caused there to be a frequency shift across the slice. It is important to note here that the signal has been attenuated due to T_2 decay because we use a 180° pulse to re-phase the transverse magnetization. Alternatively, there is gradient echo imaging for which a standard pulse sequence can be seen in Figure 2.6.

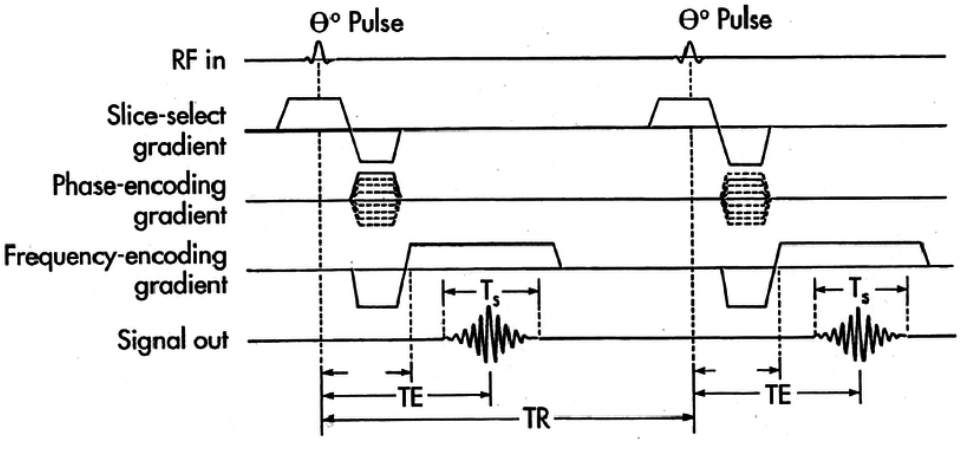
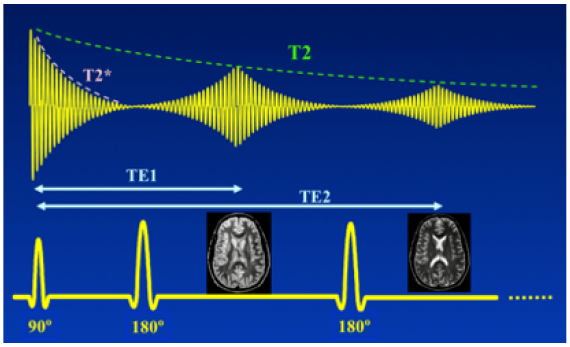
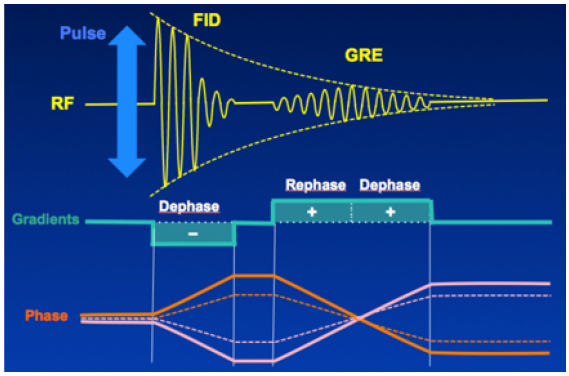


Figure 2.6: Gradient Echo Pulse Sequence

Here it is important to note that the signal is attenuated due to T_2^* decay because the transverse magnetization is not being re-phased. To clearly demonstrate this, Figure 2.7 has been added to juxtapose the signal for a spin echo sequence, Figure 2.7a, and the gradient echo signal, Figure 2.7b. Another important feature of Figure 2.7a is that it shows the contrast difference if the T_E is longer. Figure 2.8 was added to illustrate how spin echo is done qucikly and to just show the overall sequential pulses of spin echo.



(a) Signal for a spin echo pulse sequence



(b) Signal for a gradient echo pulse sequence

Figure 2.7: MRI Signal Relationships

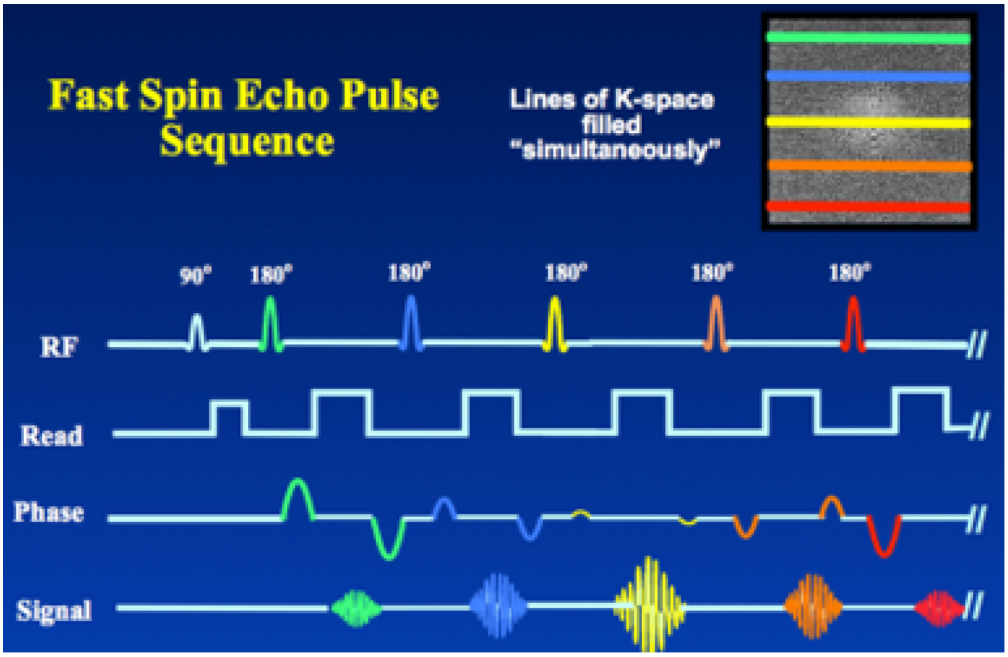


Figure 2.8: Gradient Echo Pulse Sequence

3. Landmark Thin Plate Splines Registration

The name thin plate spline refers to a physical analogy involving the bending of a thin sheet of metal. Just as the metal has rigidity, the TPS fit resists bending also, implying a penalty involving the smoothness of the fitted surface. In the physical setting, the deflection is in the z direction, orthogonal to the plane. In order to apply this idea to the problem of coordinate transformation, we interpret the lifting of the sheet of metal as a displacement of the x or y coordinates within the plane. Thin plate splines work when there are a finite number of point correspondences with irregular spacing. For simplicity sake, we will only consider the 2-D case here. The equation for the least bent surface that exactly matches the given point correspondences is:

$$f(x,y) = a_1 + a_2 x + a_3 y + \sum_{i=1}^{n} w_i U(r) \tag{3.1}$$

where $r = \sqrt{(x-x_i)^2 + (y-y_i)^2}$ or the euclidean distance between a point and its corresponding target point and $U(r)$ is a radial basis function. The first three terms correspond to the linear part which defines a flat plane that best matches all control points (this can be seen as a least square fitting). In the case that there are only 3 correspondence points, they can all be matched without bending the plane. The last term corresponds to the bending forces provided by n control points and there is a coefficient w_i for each control point. In the particular case of thin plate spline, this radial basis function, $U(r)$, is defined as the following:

$$U(r) = r^2 ln(r) \tag{3.2}$$

In order to use thin plate splines, we must first solve for the coefficients w_i and a_i. The general equation to solve for the coefficients is the following:

$$\begin{bmatrix} K & P \\ P^T & 0 \end{bmatrix} \begin{bmatrix} w \\ a \end{bmatrix} = \begin{bmatrix} v \\ 0 \end{bmatrix} \tag{3.3}$$

where the pieces of the first matrix are the following:

$$K_{ij} = \begin{bmatrix} U(r_{11}) & U(r_{12}) & \dots \\ U(r_{21}) & U(r_{22}) & \dots \\ \dots & \dots & U(r_{ii}) \end{bmatrix} \tag{3.4}$$

$$P = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ & \vdots & \\ 1 & x_i & y_i \end{bmatrix} \tag{3.5}$$

w are the radial basis function weights, a are the affine coefficients, and v are the heights of the control points off the plane. Equation 3 can be solved for the w and a matrix to obtain the thin plate spline coefficients specific to the input control points. Once these coefficients have been found, all of the other points in the image can be evaluated by passing them through equation 3.1 to obtain the final image.

4. Image Processing: Hough Transform

In image processing the **Hough Transform** is useful for *global* filtering, especially to find sets of pixels that lie along curves of a specified shape, i.e. finding lines, circles, ellipses, spheres, hypercubes, etc. It provides an elegant solution using a *discretized parameter space*, while the naïve, brute-force approach quickly becomes daunting. In the case of finding lines in an image of n points, the naïve approach involves iteration over all $\sim n^2$ possible lines and performing $\sim n^3$ comparisons for every point to each line. The $O(n^3)$ complexity exponentially worsens for shapes with higher dimensional parameter spaces. This approach is computationally prohibitive for non-trivial applications.

Instead, the Hough Transform accumulates non-background points in a discretized parameter space. The dimensionality of the parameter space is equal to the number of the parameters that describe the desired shape. In the case of a line, there are two parameters. However, the parameters must be defined with care. The naïve parameter choices for a line might be *slope* and *intercept*, but if the objective might include finding vertical (or nearly vertical) lines, a method to accurately discretize/store infinity (or very large values) would be required. A better approach is to parameterize lines by their *normal* representation:

$$x\cos(\theta) + y\sin(\theta) = \rho\ . \tag{4.1}$$

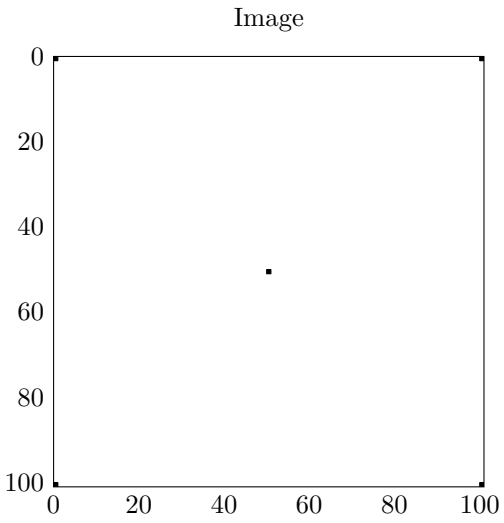
In this parameterization, all parameters are nicely bounded: $-\pi/2 \leq \theta \leq \pi/2$ and $-D \leq \rho \leq D$ for an image with D length between the two most distant corners.

The Hough Transform is a general algorithm, though in some cases it is analogous to formal mathematical transforms (eg. the Hough transform of lines with 'normal' parameterization is analogous to the Radon Transform). A typical pre-processing step for the image in question is edge detection to form a binary image, though this is not strictly necessary. The algorithm proceeds as follows:

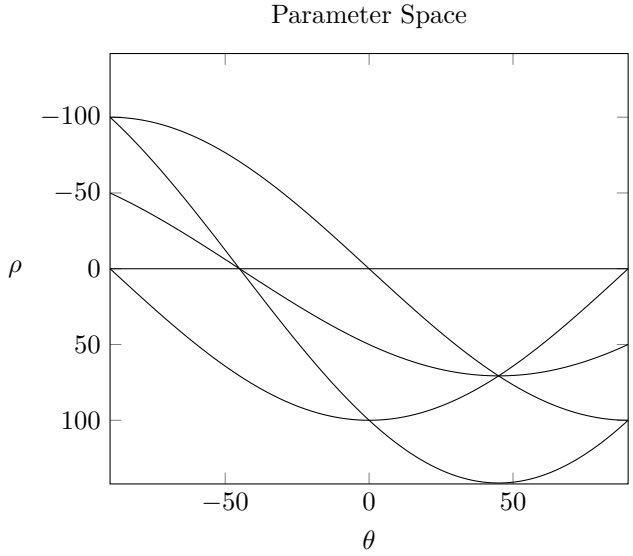
1. Allocate an accumulator image, selecting the appropriate number and width of bins for each parameter.
2. Select a foreground pixel.
3. Select center-of-bin values for (the same) $n-1$ of the n parameters.
4. Solve for the final parameter value to satisfy the reference equation at the selected pixel.
5. Increment the accumulator image at the chosen- and solved- parameter location.
6. Repeat steps 3 - 5 for all possible chosen parameter values.
7. Repeat steps 2 - 6 for all foreground pixels in the image.
8. Find parameters for prominent features by the location of accumulator maximum, or thresholding.

In the case of non-binary images, the accumulator simply need not be discrete, such that foreground (or all) pixels just have a partial-voting effect. This algorithm simply extends to higher dimensional images, and higher dimensional parameter spaces. In general, the complexity depends on the number of foreground pixels and the number of parameters; the complexity *increases* by $O(A^{m-2})$ with each additional parameter, where A is the size of the image space, and m is the number of parameters.

As a concrete example, consider the binary image (with foreground as black)



By performing a Hough Transform with (4.1) defining the parameters, and with the range of $\theta = \pm 90°$ and $\rho = \pm\sqrt{2}C$, the resulting accumulator image is



(Realistically these sinusoids are just the patterns of the discrete bins, not actual sinusoidal curves, but I'm lazy and made 5 curves instead of 101^2 boxes.) The locations where these curves intersect have accumulator values of 2 or 3 (however many curves intersect). From the intersection of the three curves at $(45, 70.7)$ and $(-45, 0)$, we conclude that there are three foreground points that lie on each lines diagonally across the image. Also notice how this special case of the Hough Transform is equivalent to the Radon Transform.

As a brief peek at generalization, the Hough Transform for a circle might be based on the parameterization

$$(x-a)^2 + (y-b)^2 = r^2 \tag{4.2}$$

where the center lies at (a, b) with radius r. With three parameters, this algorithm is significantly more complex, though it can be decreased by knowing the radius of circle which is desired *a priori*. Additionally, finding a circle presents interesting considerations, such as the *center* of the circle not being within the original image space while the pixels that contribute to that circle do.

5. Image de-noising/Wiener Filter

Wiener filtering is commonly used in photography for motion correction/de-blurring. It can also be used for de-noising when the spectral content of the noise is known, even if a system function is not applied.

5.1 Definition

The general assumption of Wiener filtering is that an acquisition is described by

$$y(t) = (h \circledast x)(t) + n(t) \tag{5.1}$$

where $\circledast$ denotes convolution and:

- $x(t)$ is some original signal (unknown) at time t.
- $h(t)$ is the known impulse response of a linear time-invariant system
- $n(t)$ is some unknown additive noise, independent of $x(t)$
- $y(t)$ is our observed signal

Our goal is to find some $g(t)$ so that we can estimate $x(t)$ as follows:

$$\hat{x}(t) = (g \circledast y)(t) \tag{5.2}$$

where $\hat{x}(t)$ is an estimate of $x(t)$ that minimizes the mean square error.

The Wiener deconvolution filter provides such a $g(t)$. The filter is most easily described in the frequency domain:

$$G(f) = \frac{H^*(f)S(f)}{|H(f)|^2 S(f) + N(f)} \tag{5.3}$$

where:

- $G(f)$ and $H(f)$ are the Fourier transforms of g and h, respectively at frequency f.
- $S(f)$ is the mean power spectral density of the original signal $x(t)$
- $N(f)$ is the mean power spectral density of the noise $n(t)$
- the superscript * denotes complex conjugation.

The filtering operation may either be carried out in the time-domain, as above, or in the frequency domain:

$$\hat{X}(f) = G(f)Y(f) \tag{5.4}$$

5.2 Example

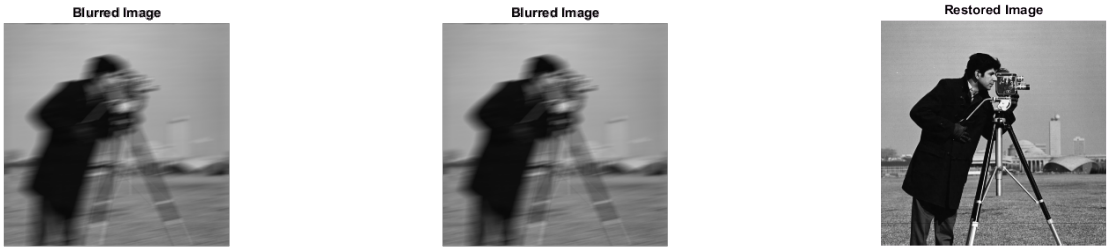
The Wiener Filter can be easily implemented in MATLAB, first without noise:

```
I = im2double(imread('cameraman.tif'));
imshow(I);
title('Original Image (courtesy of MIT)');

LEN = 21;
THETA = 11;
PSF = fspecial('motion', LEN, THETA);
blurred = imfilter(I, PSF, 'conv', 'circular');
imshow(blurred);
title('Blurred Image');

wnr1 = deconvwnr(blurred, PSF, 0);
imshow(wnr1);
title('Restored Image');
```

See Figure 5.1 to see the result.



(a) (b) (c)

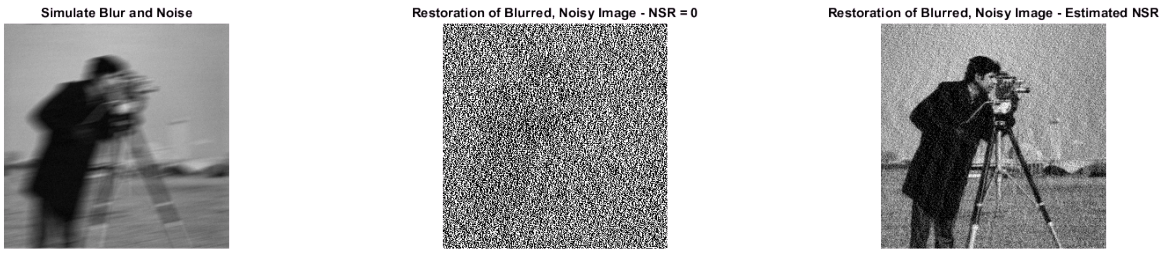
Figure 5.1: Wiener Filter Example

The addition of noise makes the deconvolution significantly more temperamental:

```
noise_mean = 0;
noise_var = 0.0001;
blurred_noisy = imnoise(blurred, 'gaussian', ...
noise_mean, noise_var);
imshow(blurred_noisy)
title('Simulate Blur and Noise')

wnr2 = deconvwnr(blurred_noisy, PSF, 0);
imshow(wnr2)
title('Restoration of Blurred, Noisy Image - NSR = 0')

signal_var = var(I(:));
wnr3 = deconvwnr(blurred_noisy, PSF, noise_var / signal_var);
imshow(wnr3)
title('Restoration of Blurred, Noisy Image - Estimated NSR');
```



(a) (b) (c)

Figure 5.2: Wiener Filter Example with Noise

6. Computed Tomography: Physics and Reconstruction

6.1 X-ray Summary

X-rays are generated from the conversion of kinetic energy of electrons into electromagnetic radiation. To create the kinetic energy of the electrons, a element (tungsten cathode) is heated to release electrons into an electric potential field where a potential difference accelerates the electron toward the target anode and collides with the target material (tungsten).

When the electron hits that material and interacts with its nucleus, X-rays are formed. If the electron hits the nucleus, the x-ray that is produced has maximal energy. If the electron passes within close proximity to a nucleus, an x-ray is still generated but the energy is lower. The farther the electron is from a nucleus, the lower the energy. The spectrum of the energy distribution is called the bremsstrahalung spectrum, seen in figure 6.1. The spikes in this figure labeled characteristic radiation come from when an incident electron has enough energy to remove an electron from a target material atom, the creating an empty electron spot. Another electron then moves into this empty location from an outer shell, which releases characteristic radiation which is specific to the location of the electron that was removed.

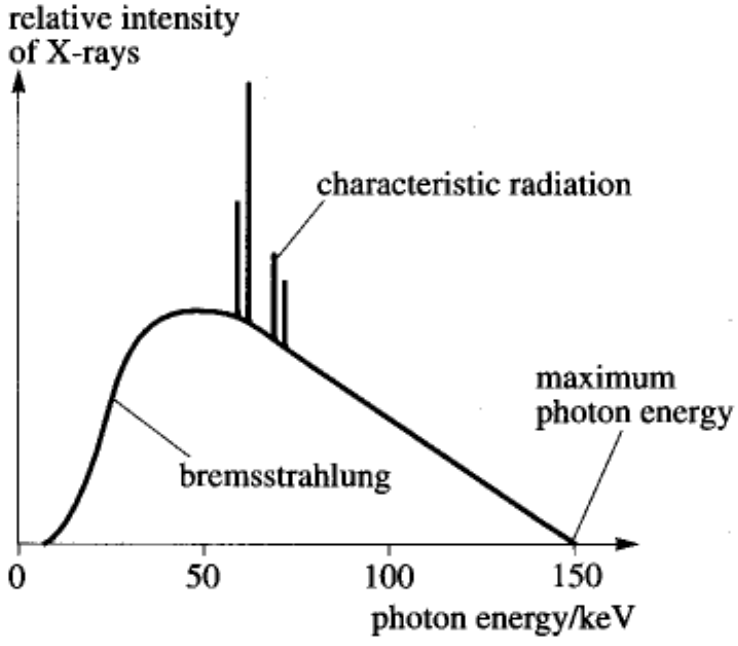


Figure 6.1: Bremsstrahalung Spectrum

6.2 Projections and Reconstructions

The basic equation for taking a projection of an object at a given angle is the following:

$$g(\rho_i, \theta_k) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} I(x,y)\ \delta\big(xcos(\theta) + ysin(\theta) - \rho\big)\ dx\ dy \tag{6.1}$$

where $g(\rho_i, \theta_k)$ is the radon transform for given sets $\{\rho_i\}$ and $\{\theta_k\}$, $I(x,y)$ is the image that is desired image, or the object, and δ is the dirac delta function. Back projection then uses these individual projections to create a final image. For a given θ the back-projected image is the following:

$$f_\theta(x,y) = g(xcos(\theta) + ysin(\theta), \theta) \tag{6.2}$$

Summation of these individual back-projected images generates a final image of the original object:

$$I(x,y) = \sum_{\theta=0}^{\pi} f_\theta(x,y) = \sum_{\theta=0}^{\pi} g(xcos(\theta) + ysin(\theta), \theta) \tag{6.3}$$

However this causes blurring due to multiple projections adding up in the middle (over-sampling), which means that the image obtained is not correct. What is needed is filtered back projection. A key theorem for this is the central slice theorem which states that the 1-D Fourier transform of a radon transform is a single slice of the 2-D Fourier transform of the image that passes through the origin at angle θ. To prove this, let us consider a single radon transform at angle θ and take the 1-D Fourier transform of it:

$$G(w,\theta) = \int_{-\infty}^{\infty} g(\rho,\theta)\ e^{i2\pi w\rho}\ d\rho \tag{6.4}$$

substituting 6.1 into 6.4 yeilds the following:

$$G(w,\theta) = \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} I(x,y)\ \delta\big(xcos(\theta) + ysin(\theta) - \rho\big)\ dx\ dy \right] e^{i2\pi w\rho}\ d\rho \tag{6.5}$$

rearranging this gives:

$$G(w,\theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} I(x,y) \underbrace{\left[\int_{-\infty}^{\infty} \delta\big(xcos(\theta) + ysin(\theta) - \rho\big) e^{i2\pi w\rho}\ d\rho \right]}\ dx\ dy \tag{6.6}$$

using the properties of the dirac delta, the equation can be simplified:

$$G(w,\theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} I(x,y) e^{i2\pi w\left(xcos(\theta)+ysin(\theta)\right)}\ dx\ dy\ =\ F(wcos(\theta), wsin(\theta)) \tag{6.7}$$

Note that this is the a slice of the 2-D Fourier transform along a line w at angle θ. Let us now look at reconstructing the desired image from the Fourier space:

$$I(x,y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(u,v) e^{i2\pi\left(ux+vy\right)}\ du\ dv \tag{6.8}$$

Now we can convert this to polar coordinates:

$$I(x,y) = \int_{0}^{2\pi} \int_{-\infty}^{\infty} w\ F(wcos(\theta), wsin(\theta)) e^{i2\pi w\left(xcos(\theta)+ysin(\theta)\right)}\ dw\ d\theta \tag{6.9}$$

Using the Fourier slice theorm from 6.7 we can write the following:

$$I(x,y) = \int_{0}^{2\pi} \int_{-\infty}^{\infty} w\ G(w,\theta) e^{i2\pi w\left(xcos(\theta)+ysin(\theta)\right)}\ dw\ d\theta \tag{6.10}$$

Here we can note that $G(w, \theta + \pi) = G(-w, \theta)$ so we can write the following:

$$I(x,y) = \int_{0}^{\pi} \int_{-\infty}^{\infty} |w|\ G(w,\theta) e^{i2\pi w\left(xcos(\theta)+ysin(\theta)\right)}\ dw\ d\theta \tag{6.11}$$

What this shows is that the desired image needs to have the 1-D Fourier transforms of the projections each filtered by $|w|$ in order properly reconstruct the image. This is called filtered back projection and now that the individual projections have been scaled, the reconstructed image is now correct.

6.3 Sampling Criteria

Assume that the X-ray beam that we are using to generate the projections has a width lets call Δs and that the detector that we are using has a certain distance between detector element centers Δr. We can now use these distances to define the sampling criteria for CT. Lets say we had a perfect theoretical projection $p_\theta(r)$, which never happens, like shown in row 1 of Figure 6.2. Because the X-ray beam that we are using to generate this projection has some witdth in the real world, we must convolve our prefect projection with a $rect()$ function that has the same width as the beam. Well this is terrible because the fourier relationship of a $rect()$ function is a $sinc()$ function, which sucks because sinc functions go on forever. Well someone decided to draw this awful relationship and that can be seen in row 2 of Figure 6.2. Anyways, the first lobe of the $sinc()$ function is at $\frac{1}{\Delta s}$ in the frequency domain and since we convolved in the spatial domain, we have to multiply the $sinc()$ function in the frequency domain, thus making our smoothed projection look like it does in row 3 of Figure 6.2.

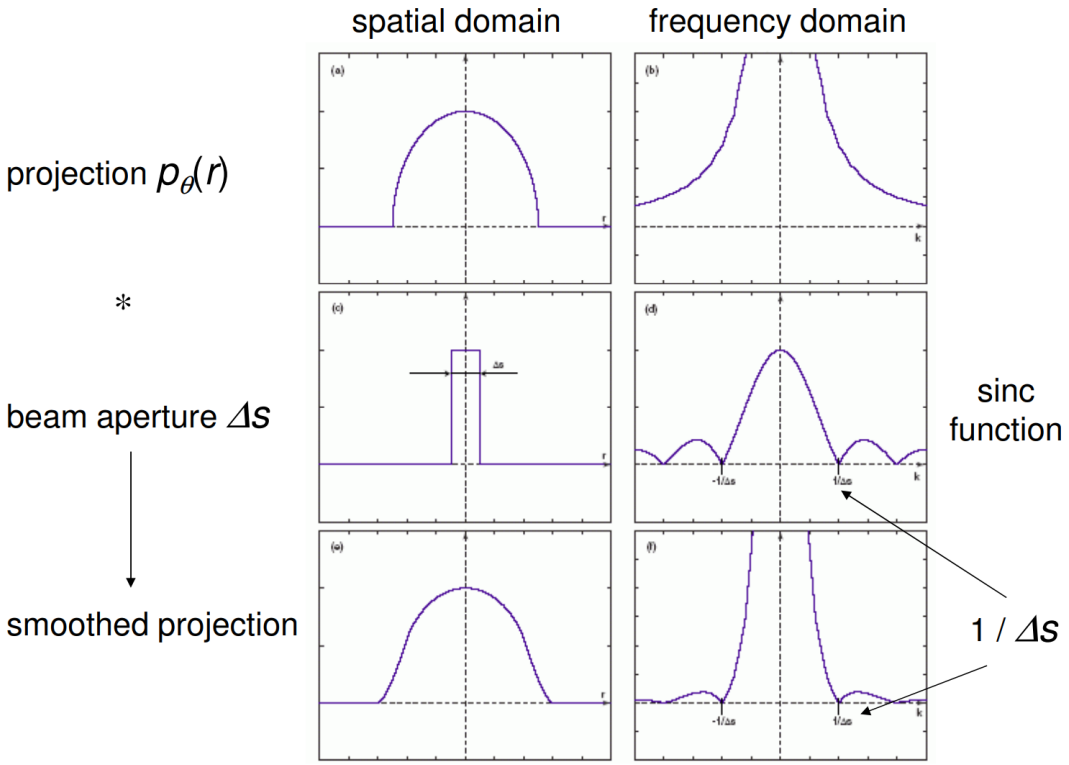


Figure 6.2: Affect of beam width on projections

So remember that Nyquist theory thing? Well it's gonna be important now because if we have samples that are Δr apart, which is what we have because that is how far apart our detectors are, then that correlates to $\frac{1}{\Delta r}$ in the frequency domain as can be seen in row 2 of Figure 6.3. Because we multiply by the sampling function in the spatial domain, we have to convolve in the frequency domain, so our copies can't overlap in the frequency domain. So guess what? No, you are wrong, according to Nyquist, the following must be true so we don't get aliasing:

$$\frac{1}{\Delta r} \leq \frac{2}{\Delta s} \quad \Longrightarrow \quad \Delta r \leq \frac{\Delta s}{2} \tag{6.12}$$

Holy shit! That is at least 2 samples per beam. If this is true, the samples in the frequency domain look like row 3 of Figure 6.3. Now we need to make sure that we get enough different angle projections to get our image back, but what should $\Delta\theta$ be? Well we searched the good old internet and find the "rule of thumb" of CT sampling which is to make the maximum angular sampling the same as the projection sampling ($\Delta\theta = \Delta r$):

$$\Delta\theta = \frac{\pi k_{max}}{M} \tag{6.13}$$

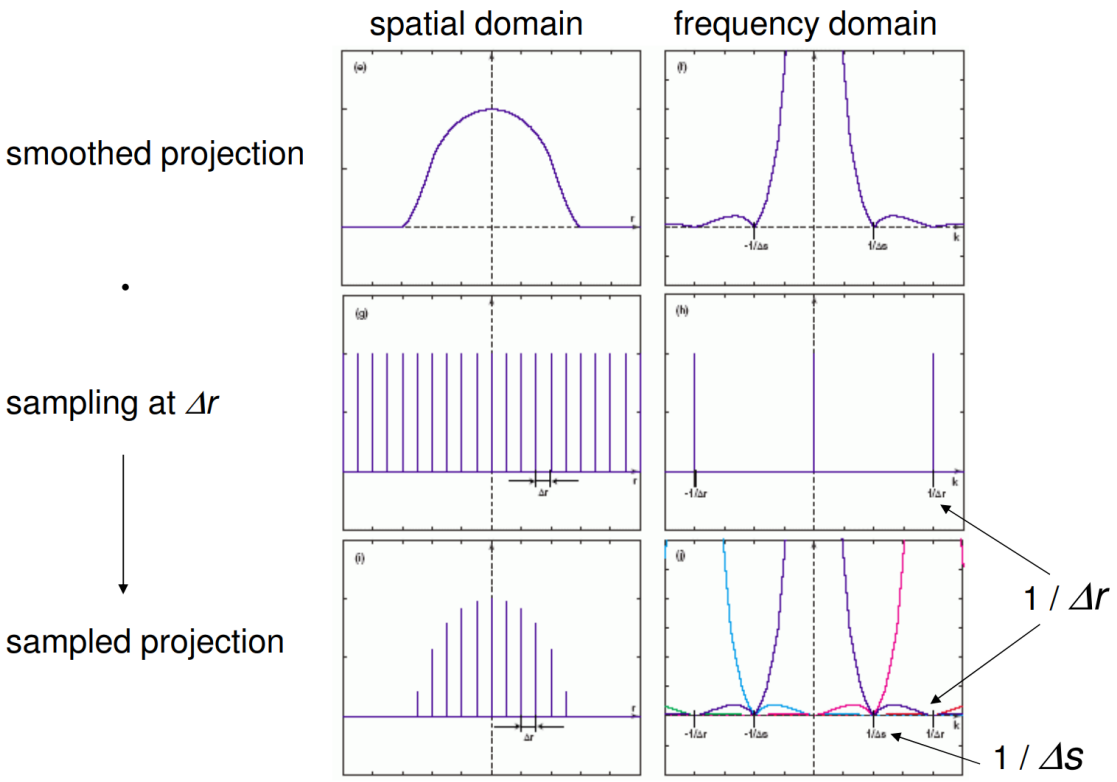


Figure 6.3: Affect of beam width on projections

$$\Delta r = \frac{k_{max}}{\frac{N}{2}} \tag{6.14}$$

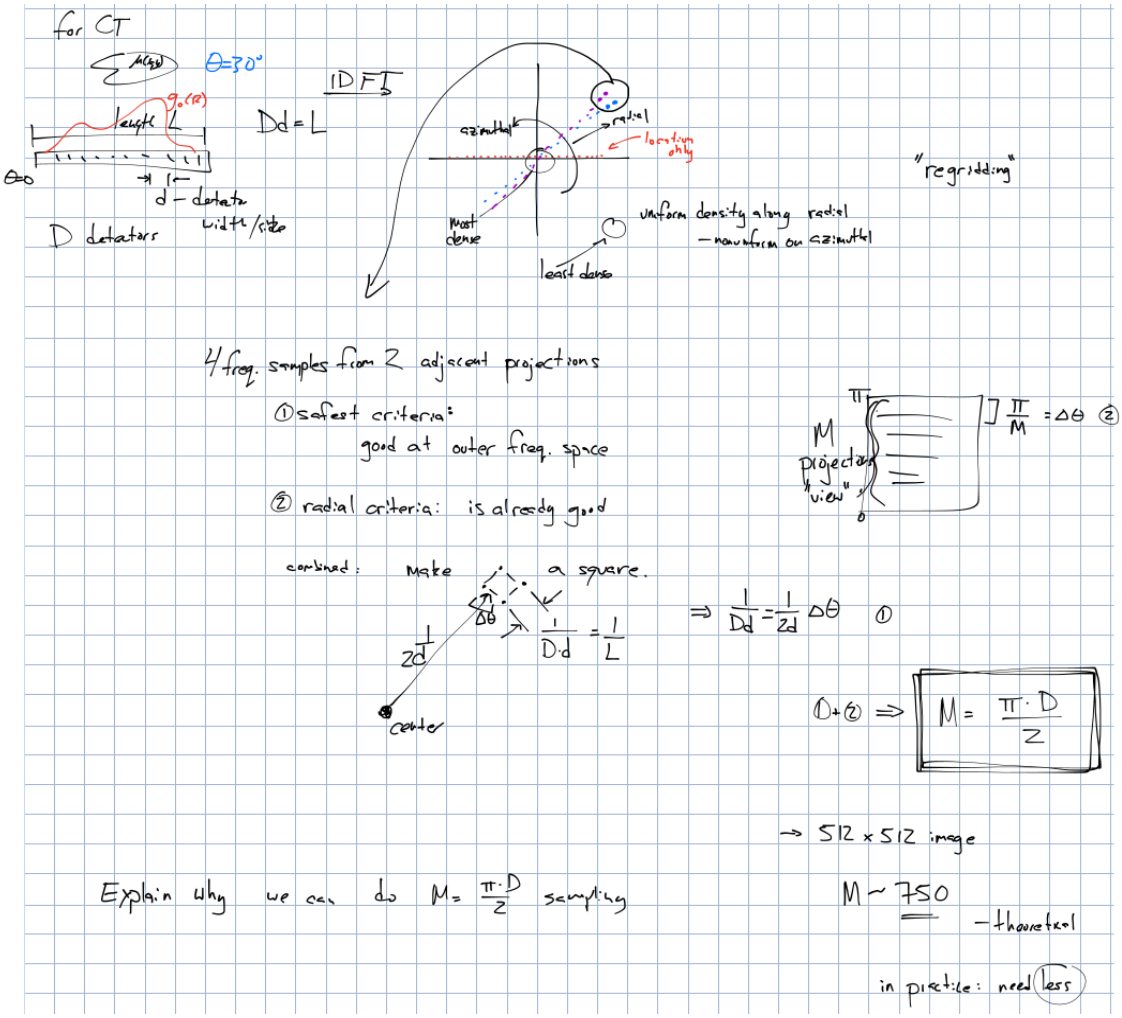
$$\Delta\theta = \Delta r \quad \Longrightarrow \quad M = \frac{\pi N}{2} \tag{6.15}$$

where N is the number of detectors, M is the number of radial samples, and k_{max} is the maximum frequency.

As an example, assume we have the objective to obtain with a scanner that has a bank of 512 detectors:

$$M = \frac{\pi\, N}{2} = \frac{\pi\, 512}{2} =\sim 804 \tag{6.16}$$

thus we need approximately 804 (evenly angularly spaced) projections.


for CT
θ=30°
1D FT
$g_\theta(t)$
length L
Dd = L
d – detector width/size
D detectors
azimuthal
radial
location only
most dense
least dense
uniform density along radial
– nonuniform on azimuthal
"regridding"
4 freq. samples from 2 adjacent projections
① safest criteria:
good at outer freq. space
② radial criteria: is already good
combined: Make a square.
$\Delta\theta$
$\frac{1}{2d}$
$\frac{1}{D \cdot d} = \frac{1}{L}$
center
$\Rightarrow \frac{1}{Dd} = \frac{1}{2d}\Delta\theta$ ①
M projection "view"
π
$\frac{\pi}{M} = \Delta\theta$ ②
①+② $\Rightarrow$ $M = \frac{\pi \cdot D}{2}$
→ 512 × 512 image
Explain why we can do $M = \frac{\pi \cdot D}{2}$ sampling
$M \sim 750$
– theoretical
in practice: need less

7. Compressed Sensing

MRI is a good candidate for CS because the image is naturally formed in a sparse frequency domain. Given the physics of MRI, it is ideal to only downsample in the phase encoding direction, as this direction is time consuming and sampling the frequency encoding direction is cheap. The trivial inverse Fourier transform reconstruction of a sub-sampled, sparse frequency domain causes significant image artifacts. The most commonly used penalty in Compressed sensing MRI is the total variation penalty, which is the following minimization problem:

$$E(I(x)) = \underset{I}{\operatorname{argmin}} \int \frac{1}{2}|| \left(F(I(x)) - d\right) Mg||^2 \, dx + \int \lambda \left|\nabla I(x)\right| \, dx \tag{7.1}$$

where F$(\cdot)$ denotes the Fourier transform, $I(x)$ is the desired image, $x \in X$ denotes a location in a vector space, λ is a scalar penalty parameter, and M is a mask matrix that nullifies the comparison between the desired image and where there is no data in the frequency domain. For convenience, the Fourier transform will be denoted a matrix, A, throughout the calculations. Substituting A for the Fourier transform and using the dual space formulation of Total Variation, Equation **??** can be re-written as:

$$D(I(x), w(x)) = \underset{I}{\operatorname{argmin}} \int \frac{1}{2}||(AI(x) - d)M||^2 \, dx + \left| - \sup_{||w_\infty||=1} \lambda \int I(x) div(w(x)) \, dx \right| \tag{7.2}$$

where D is the energy measured with the dual space of Total Variation. As the first term is not dependent on w, the *sup* operator can be placed in front of the *min* operator, giving:

$$D(I(x), w(x)) = \sup_{||w_\infty||=1} \left[\underset{I}{\operatorname{argmin}} \int \frac{1}{2}||(AI(x) - d)M||^2 \, dx + \lambda \int I(x) div(w(x)) \, dx \right] \tag{7.3}$$

The minimization problem inside the *sup* operator can now be solved by taking the Gateaux derivative with respect to $I(x)$, and assuming a fixed $w(x)$, which is:

$$\frac{\partial}{\partial \epsilon} D(I(x)+\epsilon h, w(x))\Big|_{\epsilon=0} = \frac{\partial}{\partial \epsilon} \left[\underset{I}{\operatorname{argmin}} \int \frac{1}{2}||(A(I(x) + \epsilon h) - d)M||^2 \, dx + \lambda \int (I(x) + \epsilon h) div(w(x)) \, dx \right] \Big|_{\epsilon=0} \tag{7.4}$$

$$\frac{\partial}{\partial \epsilon} \left[\underset{I}{\operatorname{argmin}} \int \frac{1}{2}||(AI(x) + A\epsilon h - d)M||^2 \, dx + \lambda \int I(x) div(w(x)) + \epsilon h div(w(x)) \, dx \right] \Big|_{\epsilon=0} \tag{7.5}$$

$$\int (AI(x) + A\epsilon h - d)MAh \, dx + \int \lambda h div(w(x)) \, dx \Big|_{\epsilon=0} \tag{7.6}$$

$$\delta_{I(x)} = \int (AI(x) - d)MAh \, dx + \int \lambda h div(w(x)) \, dx \tag{7.7}$$

This can be re-written as the following using the definition of inner product for functions $\left(\langle f(x), g(x)\rangle = \int f(x)g(x)\right)$:

$$\delta_{I(x)} = \langle (AI(x) - d)M, Ah\rangle + \langle h, \lambda div(w(x))\rangle \tag{7.8}$$

The inner products can be re-written and combined in the following manner based on the properties of inner products:

$$\delta_{I(x)} = \langle A^t(AI(x) - d)M, h\rangle + \langle \lambda div(w(x)), h\rangle \delta_{I(x)} = \langle A^t(AI(x) - d)M + \lambda div(w(x)), h\rangle \tag{7.9}$$

It is necessary for the inner product to equal zero $\forall h \in X$ for the energy D to have an extremum in X. The only case that satisfies this condition is that the first argument of the inner product equals 0:

$$\delta_{I(x)} = A^t M \left(AI(x) - d\right) + \lambda div(w(x)) = 0 \tag{7.10}$$

Typically, this variation would be used to find a closed form solution for $I(x)$. However, in this case, it would result in an exceptionally complicated equation for subsequent optimization steps. Consequently, the variation of $I(x)$ can be used in a gradient descent algorithm in conjunction with the variation of $w(x)$. The variation of $w(x)$ can be solved for in a similar fashion to that of $I(x)$. First off, Equation 7.3 can be written as the following using that the adjoint of divergence is the negative gradient:

$$D(I(x), w(x)) = \sup_{||w_\infty||=1} \left[\min_I \int \frac{1}{2} ||(AI(x) - d)M||^2 dx + \lambda \int -\nabla I(x) w(x)\ dx \right] \tag{7.11}$$

Assuming that $I(x)$ is constant and is the solution to the minimization problem, and using the Gateaux derivative with respect to $w(x)$, we can solve for $\delta_{w(x)}$ in a similar manner as above for $\delta_{I(x)}$ to result in the following:

$$\delta_{w(x)} = -\lambda \nabla I(x) \tag{7.12}$$

Another penalty that is a candidate for reconstruction is the H^1 penalty, from which the following minimization problem arises:

$$E(I(x)) = \underset{I}{\operatorname{argmin}} \int \frac{1}{2} ||F(I(x)) - d||^2 dx + \int \frac{\alpha}{2} ||\nabla I(x)||^2 dx \tag{7.13}$$

when we carry out the math in a similar fashion as before to solve for $\delta_{I(x)}$ we get the following equation:

$$\delta_{I(x)} = A^t (AI(x) - d)M - \alpha \Delta I(x) \tag{7.14}$$

At this point, the equation could be carried out to an explicit solution for $I(x)$. However, inspecting the aforementioned equation will show that this is not necessary. One of the constraints that has to be met for compressed sensing is the reconstruction technique had to be non-linear. In this instance, the penalty term is the Laplacian, which is a linear operator. If this was implemented in the frequency domain, the Laplacian would just be a multiplier on the zero filled image data. This means that none of the frequencies that were estimated at zero before reconstruction can be filled in, so the places where frequencies were not sampled, there will still be no data. This is not to say that H^1 penalty will not have an effect on the image, but it will not estimate the frequencies for the places where the k-space was not sampled.

8. General Image Processing

8.1 Convolution and Fourier Transform

A fundamental property of the Fourier transform is that convolution in time (or space) is equivalent to multiplication in the temporal (or spatial) frequency domain. The inverse, that multiplication in physical space is equivalent to convolution in frequency, is also true.

$$x(t) \circledast h(t) \Longleftrightarrow X(f) \cdot H(f) \tag{8.1}$$

Discrete Implementation

This property can be useful to speed up image processing, as the computationally-expensive convolution operation can be implemented as a simple element-wise multiplication, with overhead only of a forward and inverse Fourier transform. This is accomplished via the following proceedure:

1. Given a kernel h and a signal x, both in physical space.
2. Pad the smaller of the two arrays to the size of the larger.
3. Calculate the Fourier transform of both arrays, H and X.
4. Multiply element-wise the two arrays, so that $Y = H \cdot X$.
5. Calculate the inverse transform $y = \mathcal{F}^{-1}(Y)$.

8.2 Filtering with Kernels

Many filters exist that can perform useful operations on images. One such kernel that is popular for denoising is the Gaussian Blur. This filter simply uses a kernel that represents a Gaussian distribution function. Once the kernel is properly sized for the size of the Gaussian, it is convolved with the image. A manual-approximation estimate for one such kernel might be

$$h = \frac{1}{159}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} \tag{8.2}$$

8.2.1 Edge Detection

Edges in an image can be noticed by filtering with a kernel. Some popular kernels for edge detection include Sobel, Prewitt, and Roberts operators. Each of these produce a Gradient Magnitude and a Direction image.

Sobel

$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} \quad \mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{8.3}$$

$$\mathbf{G} = \sqrt{{\mathbf{G}_x}^2 + {\mathbf{G}_y}^2} \tag{8.4}$$

$$\mathbf{\Theta} = \operatorname{atan}\left(\frac{\mathbf{G}_y}{\mathbf{G}_x}\right) \tag{8.5}$$

Prewitt

$$\mathbf{G_x} = \begin{bmatrix} +1 & 0 & -1 \\ +1 & 0 & -1 \\ +1 & 0 & -1 \end{bmatrix} \quad \mathbf{G_y} = \begin{bmatrix} +1 & +1 & +1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \tag{8.6}$$

$$\mathbf{G} = \sqrt{{\mathbf{G}_x}^2 + {\mathbf{G}_y}^2} \tag{8.7}$$

$$\mathbf{\Theta} = \operatorname{atan2}\left(\mathbf{G}_y, \mathbf{G}_x\right) \tag{8.8}$$

8.2.2 Discrete Derivatives

Edge detection is an approximation of the first derivative, or gradient, of an image. The second derivative, of Laplacian, of an image is often calculated by applying the above gradient kernels twice. This can be done by applying the kernel to themselves, for example, the center-difference Sobel operator for the second derivative can be written

$$\mathbf{D}^2_{xy} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{8.9}$$

8.3 Median Filter

The median filter is a useful filter for de-noising and image that **cannot** be implemented with convolution. This filter finds the median intensity value for a set region (i.e. kernel size/shape) and replaces the current pixel with that median value. This filter is particularly effective against salt-and-pepper noise.