LECTURE 6

BASIC IMAGE RECONSTRUCTION

Lecture Notes by Assaf Tal

K-SPACE: A MATHEMATICAL DESCRIPTION OF SIGNAL ENCODING IN MRI

The Time Evolution of the Magnetization in the Presence of Gradients

We've previously seen that our acquired signal is

$$s(t) \propto \omega_0 \int_{\text{body}} \overline{B_{xy}^{(rec)}(\mathbf{r})} M_{xy}^{(\text{rot})}(\mathbf{r},t)\, dV$$

with (omitting the *rot* superscript for simplicity):

$$M_{xy}(\boldsymbol{r},t) = M_{xy}(\boldsymbol{r},0) e^{-i\int_0^t \Delta\omega(\boldsymbol{r},t')dt'} e^{-\frac{t}{T_2}}$$

where $\Delta\omega(\boldsymbol{r},t)$ is the offset of the spins (i.e. the z-component of the effective field in the rotating frame). Neglecting field imperfections for the time being, and in the absence of RF irradiation, the effective field in the rotating frame in the presence of gradients is:

$$\boldsymbol{B}_{eff} = \begin{pmatrix} 0 \\ 0 \\ \boldsymbol{G}(t)\cdot\boldsymbol{r} \end{pmatrix} \equiv \begin{pmatrix} 0 \\ 0 \\ \frac{\Delta\omega(\boldsymbol{r})}{\gamma} \end{pmatrix}.$$

Thus, $\Delta\omega(\boldsymbol{r},t) = \gamma \boldsymbol{G}(t)\cdot\boldsymbol{r}$. Substituting into M_{xy}, we obtain:

$$M_{xy}(\boldsymbol{r},t) = M_{xy}(\boldsymbol{r},0) e^{-i\gamma\left(\int_0^t \boldsymbol{G}(t')dt'\right)\cdot\boldsymbol{r}} e^{-\frac{t}{T_2}}$$

We now **define** a new variable, which will be of great importance in future discussions (recall $\gamma = 2\pi\gamma$):

$$\boldsymbol{k}(t) \equiv \gamma \int_0^t \boldsymbol{G}(t')dt'$$

such that:

$$M_{xy}(\boldsymbol{r},t) = M_{xy}(\boldsymbol{r},0) e^{-2\pi i \boldsymbol{k}(t)\cdot\boldsymbol{r}} e^{-\frac{t}{T_2}}$$

We are going to assume we excite our spins from thermal equilibrium onto the x-axis of the transverse plane. At thermal equilibrium, the magnetization at each point is simply given by $M_0(\mathbf{r})$, meaning

$$M_{xy}(\boldsymbol{r},0) = M_0(\boldsymbol{r})$$

Finally, we are going to assume we acquire our signal over timescales which are much shorter than T_2 of water, so $t \ll T_2 \sim 50-100$ ms. Under these assumptions:

$$s(t) \propto \int_{body} M_0(\boldsymbol{r}) e^{-2\pi i \boldsymbol{k}(t)\cdot\boldsymbol{r}} d\boldsymbol{r}$$

As you see, the signal does not *explicitly* depend on time, but rather on **k** (which depends on time). We will therefore write:

$$s(\boldsymbol{k}(t)) \equiv \hat{s}(\boldsymbol{k}) \propto \int_{body} M_0(\boldsymbol{r}) e^{-2\pi i \boldsymbol{k}(t)\cdot\boldsymbol{r}} d\boldsymbol{r}$$

The MRI Signal Is Acquired In The Fourier Space ("k-Space") Of The Image

Once we neglect T_2, the signal depends only on **k**(t), which we can control by varying the gradient **G**(t). Thus, we can think of $\hat{s}(\boldsymbol{k})$ as being acquired in some 3D k-space.

Given $M_0(\mathbf{r})$, it is computationally straightforward to compute $\hat{s}(\boldsymbol{k})$, and a sample calculation is shown below:

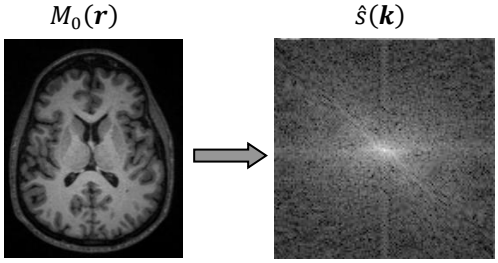


Image space k-space

The question we would like to pose is the opposite: given $\hat{s}(\boldsymbol{k})$, can we recover $M_0(\boldsymbol{r})$? The answer is yes, and is related to perhaps the most famous transform in mathematics, the Fourier transform.

k-Space and Image Space Are Related Via A Continuous Fourier Transform

Given a function $g(\boldsymbol{r})$, we can define its **continuous Fourier transform** (CFT) $\hat{g}(\boldsymbol{k})$, which is another function, as:

$$\hat{g}(\boldsymbol{k}) \equiv \int_{-\infty}^{\infty} g(\boldsymbol{r}) e^{-2\pi i \boldsymbol{k}\cdot\boldsymbol{r}} d\boldsymbol{r}$$

(this is a 3D integral over all of space)

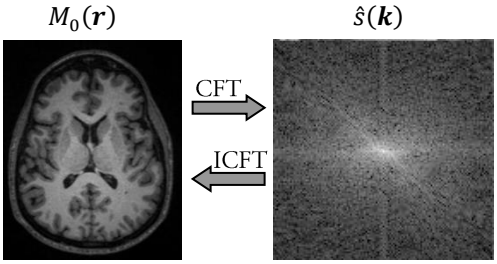
This is just a definition. However, it is now possible to prove that, if we know $\hat{g}(\boldsymbol{k})$ at each value of $\mathbf{k}$, we can use that to completely recover $g(\boldsymbol{r})$, using the **inverse continuous Fourier transform** (ICFT):

$$g(\boldsymbol{r}) = \left(\frac{1}{2\pi}\right)^3 \int_{-\infty}^{\infty} \hat{g}(\boldsymbol{k}) e^{2\pi i \boldsymbol{k}\cdot\boldsymbol{r}} d\boldsymbol{r}$$

There are multiple proofs of this well-known and non-trivial theorem, and while not difficult we will not present any here since we are not interested in the mathematical foundations of signal processing. Instead, we will just use it for our purposes. This theorem has a one-dimensional analogue[1]:

$$\begin{cases} \hat{s}(k) = \int_{-\infty}^{\infty} s(x) e^{-2\pi i k x} dx & \text{(CFT)} \\ s(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \hat{s}(k) e^{+2\pi i k x} dx & \text{(ICFT)} \end{cases}$$

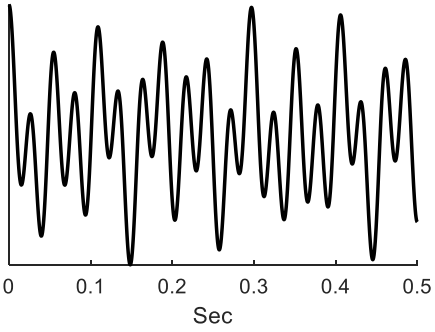
If we make the correspondence $s(x) = M_0(x)$, then we immediately see that the acquired signal $\hat{s}(k)$ is the CFT of the image $M_0(x)$ (and, in 3D, $\hat{s}(\boldsymbol{k})$ is the CFT of $M_0(\boldsymbol{r})$):



Note: since the CFT and ICFT definitions are so similar, authors often switch between the CFT and ICFT, and might call our ICFT a CFT. There really is very little difference apart from a scaling factor ($1/2\pi$) and flipping the resulting transform (which has –k instead of k in the exponent). Our particular choice of names stems from trying to mimic as closely as possible the way the popular academic programming language MATLAB handles Fourier transforms.

We have already introduced the concept of a Fourier transform in the previous lecture as a "magical box" that can detect and tell apart frequency components in the signal. For example, if we take the function

$$f(t) = \cos(2\pi\nu_1 t) + 2 \cdot \cos(2\pi\nu_2 t) + 3 \cdot \cos(2\pi\nu_3 t)$$

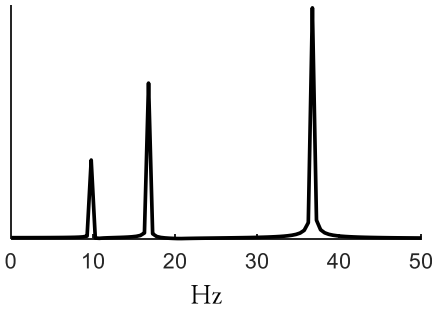


with $\nu_1 = 10$ Hz, $\nu_2 = 17$ Hz, $\nu_3 = 37$ Hz, then its Fourier transform,

$$\hat{f}(\nu) = \int_{-\infty}^{\infty} f(t) e^{-2\pi i \nu t} dt$$

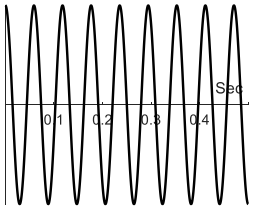
will look like this:

[1] The signs and $1/2\pi$ factors differ between different textbooks and papers. This is a consequence of the Fourier transform theorem: one only needs the signs in the exponential in the CFT and ICFT to be opposite. Similarly, one needs the factors of the integrals to equal $\frac{1}{2\pi}$, so in some books both the CFT and ICFT have a $\frac{1}{\sqrt{2\pi}}$ factor in front of them, and in some books the CFT has the $\frac{1}{2\pi}$ factor in front of it.



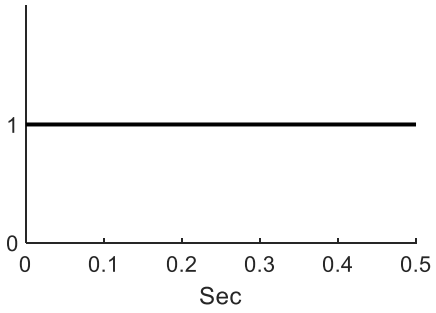
How does it work? If you look at the value of the Fourier transform at a given frequency – say, $\bar{\nu}$ – then it is given by taking a plane way, $e^{-2\pi i\bar{\nu}t}$, multiplying it by the function f(t), and summing across time. If f(t) is another plane wave with a frequency that is not $\bar{\nu}$, then the two plane waves will interfere destructively, yield another plane wave, with an average area of zero (it will have negative and positive lobes which cancel out):

$$\mathrm{Re}(e^{-2\pi i\bar{\nu}t}e^{-2\pi i\nu t})$$
$$\nu \neq \bar{\nu}$$



However, if the frequencies are equal – that is, if you look at the value of $f(\nu)$ at the frequency $\bar{\nu}$, and that frequency happens to coincide with the frequency of the plane wave f(t) – the two plane waves will completely cancel out and the remaining constant function will have a non-zero total area:

$$\mathrm{Re}(e^{-2\pi i\bar{\nu}t}e^{-2\pi i\nu t})$$
$$\nu = \bar{\nu}$$



If f(t) has several frequency components – i.e., it is the sum of several terms with different frequencies – then its Fourier transform $\hat{f}(\nu)$ will become non-zero and you will obtain a peak whenever ν coincides with one of the frequencies in the signal f(t).

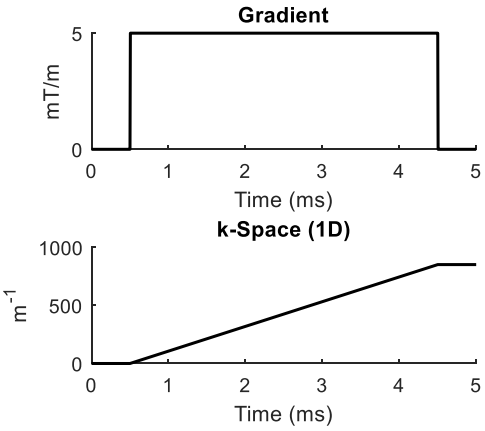
Sampling k-Space

We Can "Take A Walk" In k-Space By Varying The Gradient

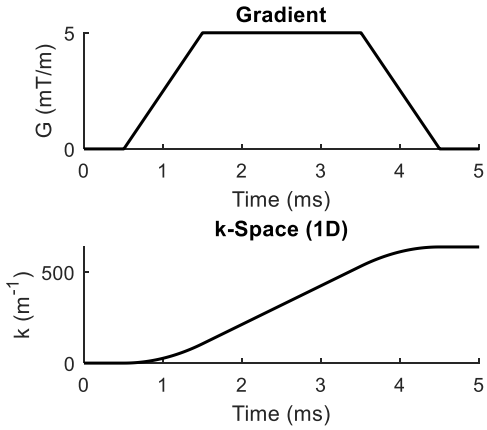
We manipulate $\mathbf{k}(t)$ by changing the gradient. According to its definition,

$$\mathbf{k}(t) = \gamma\int_0^t \mathbf{G}(t')dt',$$

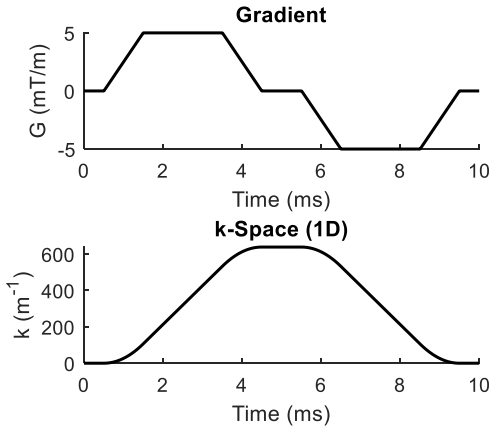
it is equal (per-component) to the total area underneath the gradient. In 1D, for example:



Of course, a realistic gradient would need non-zero ramp up and down times:



Since we can apply negative gradient values we can also "rewind" **k** back to the origin:



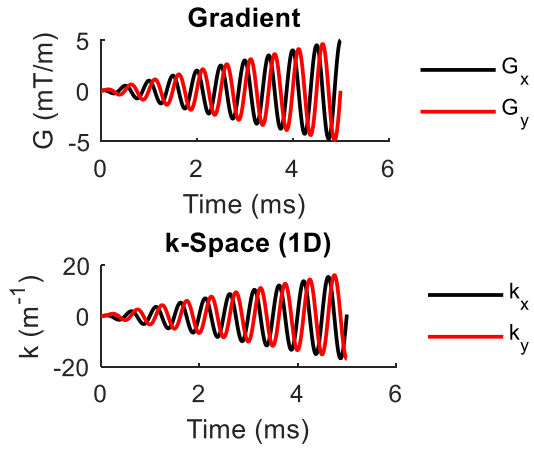
As another example, consider the following 2D gradient waveform function:

$$G_x(t) = G_0 \cdot \tfrac{t}{T} \cdot \cos(\omega_g t)$$
$$G_y(t) = G_0 \cdot \tfrac{t}{T} \cdot \cos(\omega_g t)$$

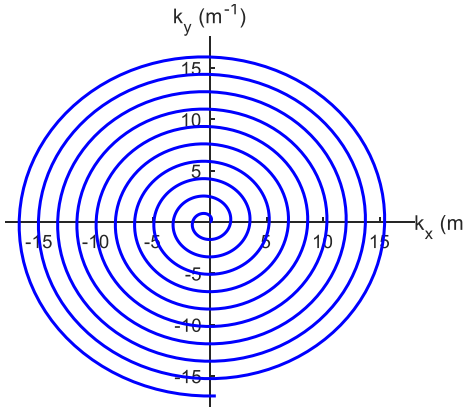
with T = 5 ms, G_0 = 5 mT/m and ω_g =2π·2 Hz. We can find **k**(t) easily by integrating:

$$k_x = \gamma \int_0^t G_x(t')\,dt'$$
$$k_y = \gamma \int_0^t G_y(t')\,dt'$$

We will not actually carry out the integration analytically as it is not particularly interesting. Instead, I've used a plotting software package to plot both G_x, G_y and k_x, k_y as a function of time:



We can think of **k**(t) as tracing a "path" in the k_x-k_y plane, by plotting it parametrically (that is, plotting the position of the vector **k**(t) as a function of the variable *t*):



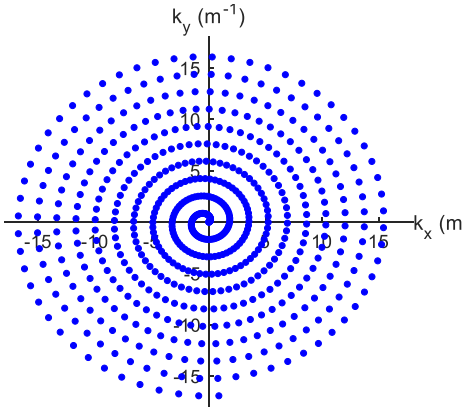
In effect, we are sampling data along the this trajectory in k-space (that is, we are measuring s(**k**) along the trajectory).

Our Sampling Of k-Space Is Limited by Both The Total (1) Acquisition Time And (2) Sampling Rate

Although $\hat{s}(\boldsymbol{k})$ and $M_0(\boldsymbol{r})$ are related via a CFT, and although $M_0(\boldsymbol{r})$ can be recovered from $\hat{s}(\boldsymbol{k})$ completely by carrying out an ICFT, we are faced with a problem: We cannot measure $\hat{s}(\boldsymbol{k})$ at every point in k-space, but only along the "trajectory" of **k** in k-space. It should be quite clear to the reader at this point that traversing k-space by varying the gradients takes time, and that we do not have

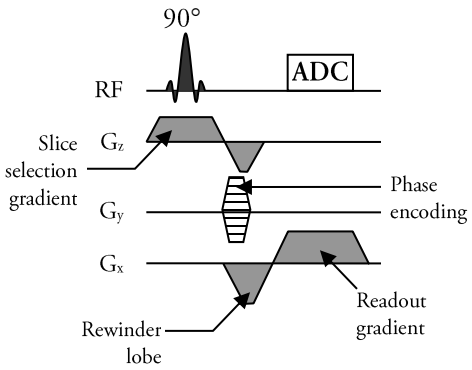
infinite acquisition time, seeing as the magnetization decays in the xy-plane with a time constant T_2. Hence, the extent to which we can cover k-space is limited.

There is another factor limiting our coverage of k-space, which is the sampling rate of the analog to digital converter (ADC). The ADC samples at a constant rate in time steps Δt called the **dwell time**. Thus we do not measure data along a continuous **k** variable but only at discrete points, i.e. a discrete trajectory. In the spiral trajectory we've seen above, we would actually only sample $\hat{s}(\boldsymbol{k})$ on a non-rectilinear grid:

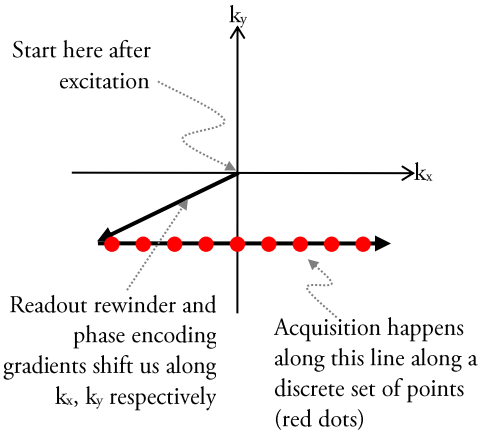


Phase and Frequency Encoding Are Often Combined To Read Out A Cartesian Data Set In k-Space

We have previously outlined a 2D pulse sequence which combines phase and frequency encoding, but have not delved into its workings. We will take a closer look at it now, with a small modification: we will insert a **rewinder gradient** before the readout gradient, for a reason that will become clear in a moment:

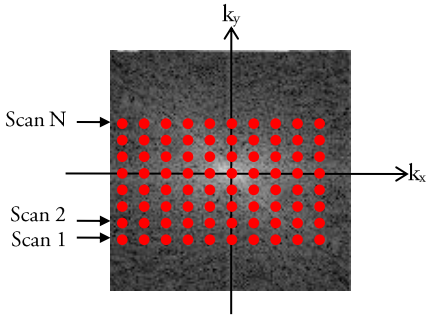


The above sequence is repeated multiple times, each time incrementing the phase encoding gradient from some initial negative value to some final positive value. A particular scan and its corresponding dataset in k-space would look like this:



Thus the purpose of the rewinder gradient is to shift us back along the readout axis (k_x) so we read out data symmetrically in k-space.

Repeating the sequence multiple times, varying the phase encoding gradient each time, yields a 2D data set of values of s(**k**) sampled on a cartesian grid of **k** points. Each line in k-space will result from an excitation-rewinding-acquisition block:



To summarize:

> Data $\hat{s}(\boldsymbol{k})$ in MRI is acquired in **k**-space by varying the gradient as a function of time. The data is proportional to the continuous Fourier transform of the image we wish to reconstruct, sampled on a discrete (although not necessarily rectilinear) grid of points.

In the next section we will examine the problem of reconstructing the image f(**r**) from this discrete sampled dataset, assuming the grid is rectilinear.

The Discrete Fourier Transform

Discrete Sampling Leads To Aliasing, Blurring and Ringing

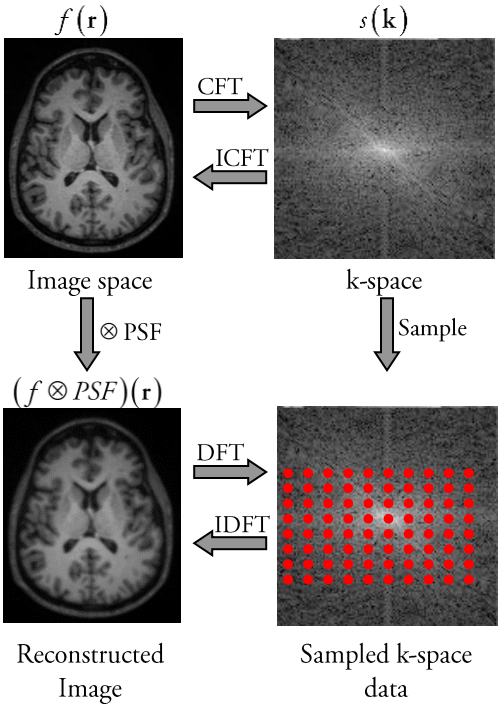
In the remainder of this lecture we will concern ourselves with two questions:

1. How do we reconstruct our image from the discretely sampled data?
2. What effect will this have on the image? That is, how will our reconstructed image compare to the "true" image f(**r**)?

The answers to these two questions will be:

1. Reconstruction takes place via a **Discrete Fourier Transform** (DFT).
2. The sampling will yield an image that is obtained by convolving the true image with a point spread function and sampling it on a discrete grid. The convolution will lead to three effect:
   a. **Blurring**. the reconstructed image will look like a blurred version of $M_0(\boldsymbol{r})$. This will be a result of the limited extent of measurement in k-space – that is, the maximal value of **k** we will sample.
   b. **Aliasing**. Copies of the original image will appear if the spacing between the acquired points in k-space is not kept small enough.
   c. **Ringing**. Sharp edges in the original image will appear to "ring", namely, create small "waves" around them. This will be an unavoidable feature which will grow less substantial as the number of points increases.

Graphically:



Statement Of The Problem

Let us confine ourselves to a one dimensional problem. Given an "image" $s(x) = M_0(x)$, we form its continuous Fourier transform:

$$\hat{s}(k) = \int_{-\infty}^{\infty} M_0(x) e^{-2\pi i k x} dx$$

We sample $\hat{s}(k)$ on a discrete grid of N points, starting from some initial value $-k_{max}/2$ and advancing in steps $\Delta k = k_{max}/N$:

$$k_n = -\frac{k_{max}}{2} + n \cdot \Delta k$$
$$n = 0,1,\ldots,N-1$$

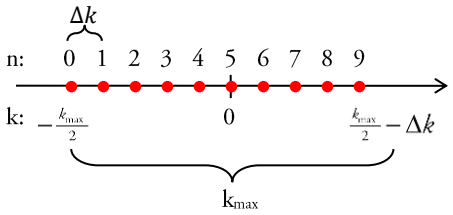
If the readout gradient is constant in magnitude, the interval Δk is determined by the gradient strength, G, and the ADC dwell time, Δt:

$$\Delta k = \gamma G \Delta t$$
$$N \Delta k = k_{max}$$

However, it will not matter to us ***how*** the points in k-space were sampled, only that they ***were*** sampled.

The N points k_n in this particular sampling pattern are not symmetrically placed around 0: there is a last point missing at $k_{max}/2$. However, if

you think in terms of *intervals*, it is symmetrically placed. Here is an example with N=10:



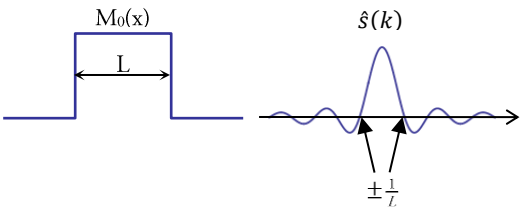
We now pose the question: given the set $\hat{s}_n \equiv \hat{s}(k_n)$, how do we recover $M_0(x)$? It would be instructive to take a particular example. Suppose our "image" $M_0(x)$ is an uninteresting boxcar:

$$M_0(x) = \begin{cases} 1 & |x| \le \frac{L}{2} \\ 0 & \text{elsewhere} \end{cases}$$

It is actually feasible to compute its CFT analytically:

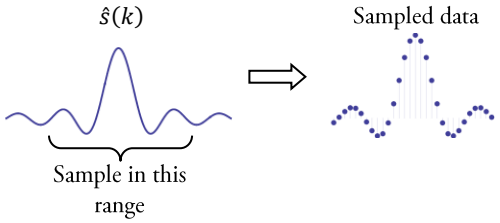
$$\begin{aligned} \hat{s}(k) &= \int_{-\infty}^{\infty} M_0(x) e^{-2\pi i k x} dx \\ &= \int_{-\frac{L}{2}}^{\frac{L}{2}} e^{-2\pi i k x} dx \\ &= \frac{e^{-2\pi i k x}}{-2\pi i k}\Big]_{-\frac{L}{2}}^{\frac{L}{2}} \\ &= \frac{1}{\pi k}\frac{\left(e^{\frac{2\pi i k L}{2}} - e^{-\frac{2\pi i k L}{2}}\right)}{2i} \\ &= L\frac{\sin(\pi k L)}{\pi k L} = L \cdot \mathrm{sinc}(\pi k L) \end{aligned}$$

where we have used $\mathrm{sinc}(x) = \frac{\sin(x)}{x}$ and $\sin(x) = \frac{e^{ix} - e^{-ix}}{2i}$. The two functions are plotted next to each other below:



This highlights a general feature of Fourier transforms, namely that the width of the function in one domain is approximately equal to the inverse of its transform in the Fourier domain.

Due to our discrete sampling, we only know the values of $\hat{s}(k)$ on our discretely sampled grid:



We hope that, if we sample "enough" points, we should be able to approximate the continuous Fourier integral by a discrete sum. This motivates us to **define** the discrete Fourier transform and its inverse:

Given a discrete set of (possibly complex) points $s_0, s_1, \ldots, s_{N-1}$, we define their **discrete Fourier transform** (DFT) as the set of points $\hat{s}_0, \hat{s}_1, \ldots, \hat{s}_{N-1}$:

$$\hat{s}_m = \frac{1}{N}\sum_{n=0}^{N-1} \hat{s}_n e^{-\frac{2\pi i n m}{N}} \quad \text{(DFT)}$$

Then it is possible to prove that, given $\hat{s}_0, \hat{s}_1, \ldots, \hat{s}_{N-1}$, the numbers $s_0, s_1, \ldots, s_{N-1}$ can be recovered by computing the **inverse discrete Fourier transform**:

$$s_n = \frac{1}{N}\sum_{m=0}^{N-1} \hat{s}_m e^{\frac{2\pi i n m}{N}} \quad \text{(IDFT)}$$

Compare this to the CFT and its inverse!

The Point Spread Function Of Cartesian Sampling Is The "Dirichlet Kernel"

We would *like* to carry out an ICFT of the full k-space data, $\hat{s}(k)$, and, instead, we're going to carry out an IDFT of the partial, discrete data $\hat{s}_m \equiv \hat{s}(k_m)$:

$$s_n = \frac{1}{N}\sum_{m=0}^{N-1} \hat{s}_m e^{\frac{2\pi i n m}{N}} \qquad (n = 0,1,\ldots,N-1)$$

We are going to hope the computed values $s_0, \ldots, s_{N-1}$ are going to recover our image $M_0(x)$.

In what follows we're going to carry out an ugly calculation – you can skip it if you'd like and just jump down to the final expression.

Let's follow this rabbit down the rabbit hole and see where it leads us. First, we note that the k-space data is given by

$$\hat{s}(k) = \int_{-\infty}^{\infty} M_0(x) e^{-2\pi i k x} dx$$

and therefore our sampled data points at $k_m = -\frac{k_{max}}{2} + m \cdot \Delta k$ are given by:

$$\begin{aligned} \hat{s}_m &\equiv \hat{s}(k_m) \\ &= \int_{-\infty}^{\infty} M_0(x) e^{-2\pi i k_m x} dx \\ &= e^{i\pi k_{max} x} \int_{-\infty}^{\infty} M_0(x) e^{-2\pi i m \Delta k x} dx \end{aligned}$$

for $m = 0,1,\dots,N-1$. We get, upon substitution and interchange of the summation and integration signs:

$$\begin{aligned} s_n &= \frac{1}{N} \sum_{m=0}^{N-1} \hat{s}_m e^{\frac{2\pi i m n}{N}} \\ &= \int_{-\infty}^{\infty} M_0(x) \frac{e^{i\pi k_{max} x}}{N} \left(\sum_{m=0}^{N-1} e^{-2\pi i m \Delta k x} e^{\frac{2\pi i m n}{N}} \right) dx \end{aligned}$$

We are going to simplify this by combining the exponents inside the summation (since $e^a e^b = e^{a+b}$):

$$s_n = \int_{-\infty}^{\infty} M_0(x) \frac{e^{i\pi k_{max} x}}{N} \left(\sum_{m=0}^{N-1} e^{-2\pi i m \Delta k \left(x - \frac{n}{N\Delta k} \right)} \right) dx$$

We can write

$$\begin{aligned} e^{i\pi k_{max} x} &= e^{i\pi k_{max} \left(x - \frac{n}{N\Delta k} \right)} e^{\frac{i\pi k_{max} n}{N\Delta k}} \\ &= e^{i\pi n} e^{i\pi k_{max} \left(x - \frac{n}{N\Delta k} \right)} \\ &= (-1)^n e^{i\pi k_{max} \left(x - \frac{n}{N\Delta k} \right)} \end{aligned}$$

because $e^{i\pi n} = \cos(\pi n) + \iota \sin(\pi n) = (-1)^n$. Putting this back, we see that we can write s_n as

$$s_n = (-1)^n \int_{-\infty}^{\infty} M_0(x) PSF\left(x - \frac{n}{k_{max}} \right) dx$$

with

$$PSF(x) = e^{i\pi k_{max} x} \sum_{m=0}^{N-1} e^{-2\pi i m \Delta k x}$$

The only thing we need to do is carry out the summation. We can do this by noting that, in general,

$$e^{xy} = (e^x)^y$$

so

$$e^{-2\pi i m \Delta k x} = (e^{-2\pi i \Delta k x})^m \equiv a^m$$

with

$$a \equiv e^{-2\pi i \Delta k x}$$

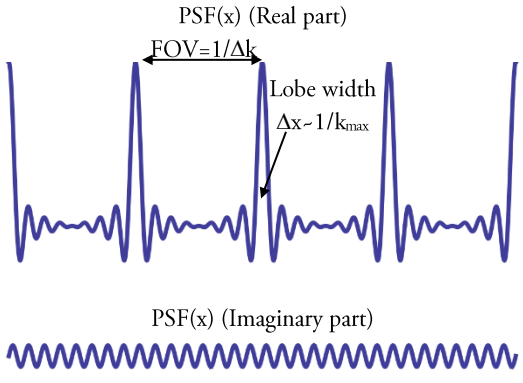
This just means the summation is the sum of a geometric series (you can find this in Wikipedia if you don't know how to derive it):

$$\sum_{m=0}^{N-1} a^m = \frac{(1 - a^N)}{1 - a}$$

Plugging in our expression for a and simplifying, we obtain:

$$\begin{aligned} PSF(x) &= e^{i\pi \Delta k x} \frac{\sin(\pi k_{max} x)}{\sin(\pi \Delta k x)} \\ k_{max} &= N \cdot \Delta k \end{aligned}$$

This is known as the Dirichlet kernel. We plot it and note some crucial features:

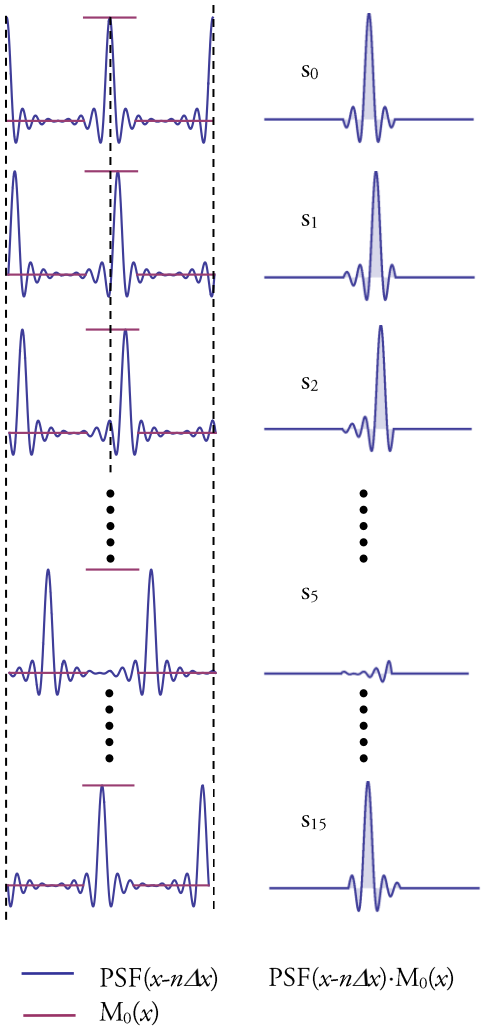


It is periodic with a period $1/\Delta k$, and its lobes have a width approximately given by $1/k_{max}=1/(N\Delta k)$, i.e. N times smaller than its periodicity. We define two quantities known as the Field Of View (FOV) and spatial resolution (Δx):

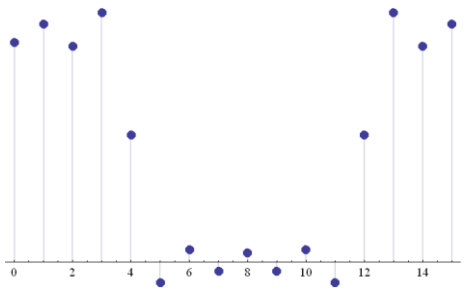
$$FOV = \frac{1}{\Delta k}$$
$$\Delta x = \frac{1}{k_{max}} = \frac{1}{N \cdot \Delta k} = \frac{FOV}{N}$$

The Point Spread Function "Scans" The Image From –FOV/2 To FOV/2, In Steps Of Δx

To visualize the imaging process, I've plotted the boxcar function $M_0(x)$ with a width of unity, and have chosen the following sampling parameters: FOV = 2, N = 16 so $\Delta x=0.125$ and $\Delta k=0.5$. Since N=16, we have 16 k-space samples $\hat{s}_0, \hat{s}_1, \ldots, \hat{s}_{15}$ from which we construct 16 image coefficients $s_0, s_1, \ldots, s_{15}$. The j^{th} coefficient is obtained by multiplying the true distribution of spins, f(x), by $PSF(x - n \cdot \Delta x)$, where $\Delta x = 1/k_{max}$, and integrating (i.e. calculating the area). This is shown in the following diagram:

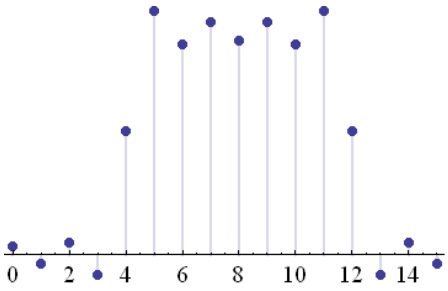


We see that the PSF "moves" in small steps of Δx. When the main lobe moves outside the spin distribution the area becomes negligible, and when it moves back in the signal grows back up. If we actually calculate s_j and plot the result, we obtain:

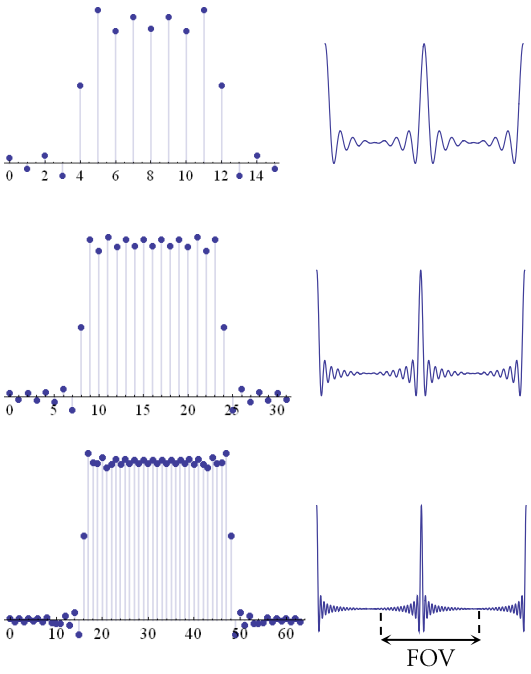


This looks like a "mirrored" version of $M_0(x)$ about x=0. This mirroring is the result of the PSF starting from the center of the image at n=0 and not from

the far left edge. This implies that we need to "switch" the right and left parts of the image, which yields:



This is not a perfect image of the boxcar function but it's not too bad. Here are the coefficients s_k when we increase N to 32, and then 64 (shown on the left) and the corresponding changes to the PSF on the right:

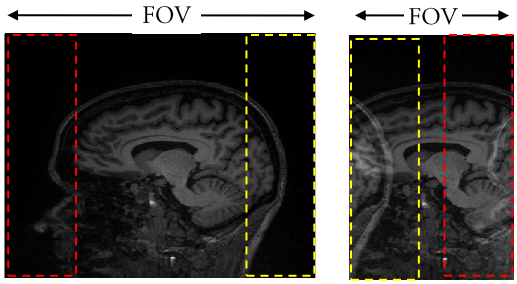


Aliasing Is Caused When The Steps In k-Space Are Not Small Enough

The Dirichlet kernel has a periodic structure with periodicity given by the Field of View:

$$PSF(x + FOV) = PSF\left(x + \frac{1}{\Delta k}\right) = PSF(x)$$

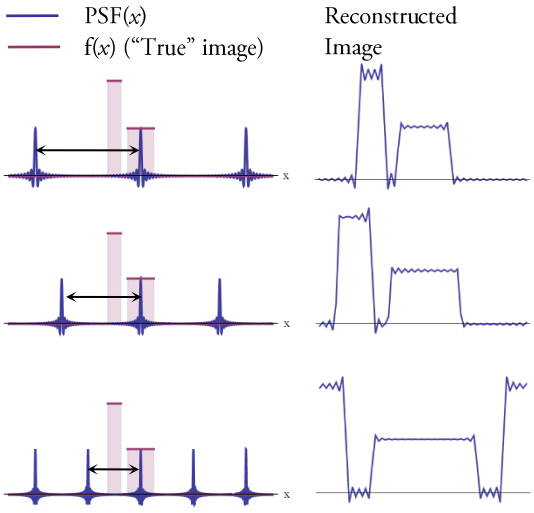
If the FOV is smaller than the imaged object then we will get **aliasing**. This describes a phenomena in which the distance between adjacent lobes, given by the FOV, becomes smaller than the object. Thus, as the PSF "scans" the object, non-central lobes will re-enter the image, effectively causing multiple copies of the object to appear in the reconstructed image. This has a very classic appearance in MRI, in which one side of the image "wraps" into the opposite side:



Aliasing is removed by keeping the FOV larger than the object:

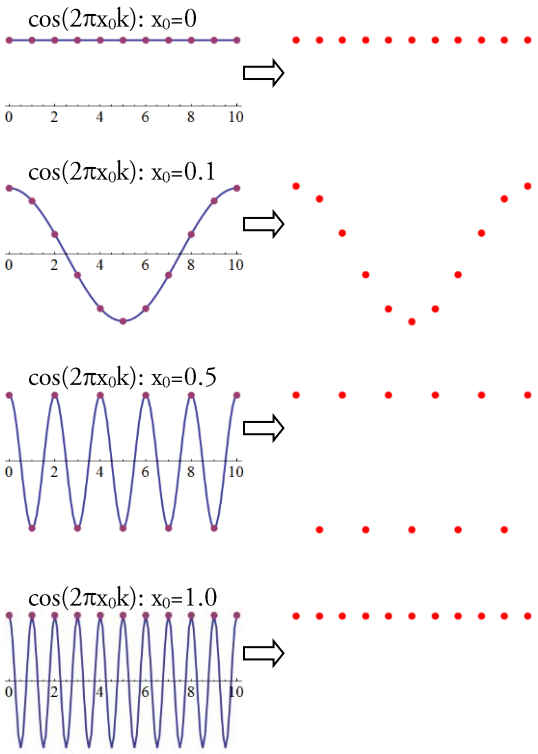
To avoid aliasing, keep FOV=1/Δk bigger than the object's dimensions, where $\Delta k = \gamma G \Delta t$, G is the gradient strength, and Δt is the dwell time.

This is illustrated in the following diagram, showing an object (f(x), two boxcars) of size 3 arbitrary units, and a point spread function with FOV = 4.0, 3.0 and 2.0. In the last case aliasing will occur (N=64).



Another Way To Understanding Aliasing

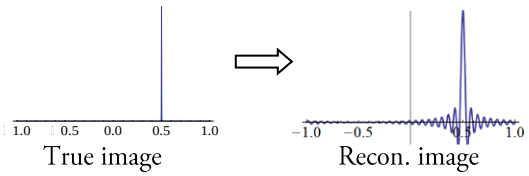
Aliasing can also be understood from a different perspective: we are sampling the image in k-space, which is its Fourier space. This space represents the frequencies of the object. Once we only consider discrete frequencies in jumps of Δk, we lose the ability to tell apart positions x and $x+1/\Delta k$, as shown in the following diagram, in which $\Delta k=1$ (in arbitrary units) and $\cos(2\pi x_0 k)$ and its sampled version are both plotted for x_0=0, 0.1, 0.5 and 1.0, showing that, for x_0=0 and 1, the set of sampled points coincides and the two positions cannot be discerned:



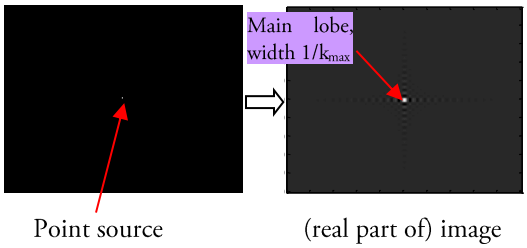
Blurring Occurs When We Don't Go Far Enough In k-Space

The width of the Dirichlet kernel's main lobe sets the *resolution* of the image. Any point image $\delta(x-x_0)$ will be replaced by the PSF, centered at x_0:

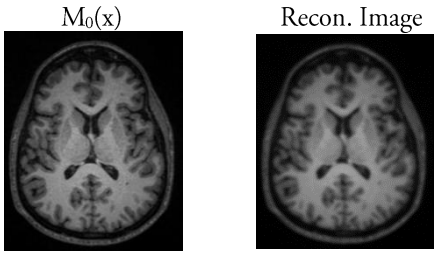
$$\delta(x-x_0) \rightarrow \int \delta(x'-x_0)PSF(x-x')dx' = PSF(x-x_0)$$



This means features will get "broadened" by the PSF's lobe's main width, which is approximately $\frac{1}{k_{max}}$. This determines our real spatial resolution. The same effect can be seen in 2D and 3D cartesian sampling schemes as well. For example, in 2D (FOV_x=FOV_y=256 mm, N_x=N_y=64):



When the image is comprised of many point sources, as is the case with real images, each point will get broadened which is perceived as blurring:



To reduce blurring, keep $\Delta x=1/k_{max}$ as small as possible, by going "far out" in k-space.

Of course, how far we can go out in k-space will depend on how long we have to sample and how strong our gradients are.

Another Way to Understand Blurring

Blurring can also be understood as follows: if we could sample all of k-space we could reconstruct our source image perfectly with an inverse continuous Fourier transform:

$$f(x) = ICFT[s(k)]$$

Putting aside for a moment the question of the discreteness of sampling, we look at the effect of the finiteness of our sampling extend. We're basically sampling s(k) only in some interval $\left[-\frac{k_{max}}{2},\frac{k_{max}}{2}\right]$. This is the same as fully sampling s(k) times a windowing function W(k), which is equal to 1 in $k\in\left[-\frac{k_{max}}{2},\frac{k_{max}}{2}\right]$ and 0 outside the interval. What would we get if we were to apply an ICFT to that?

$$ICFT\left[s(k)\cdot W(k)\right]=?$$

Fortunately, a well known theorem from Fourier theory[2] called the Convolution Theorem states that the Fourier of the product equals the convolution of the Fouriers. That is:

$$ICFT[s(k)\cdot W(k)] = ICFT[s(k)]\otimes ICFT[W(k)]$$

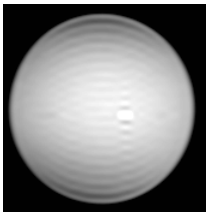
We already know that $f(x)=ICFT[s(k)]$, and we've also shown that the Fourier transform of a boxcar function W(k) of width k_{max} is a sinc of width ~ $1/k_{max}$. Thus,

$$ICFT[s(k)\cdot W(k)] = (\text{image})\otimes\left(\text{sinc of width}\frac{1}{k_{max}}\right)$$

This in essence restates our previous result: features will get "broadened" by the sinc function's main width, which is approximately $\frac{1}{k_{max}}$.

The "Wiggles" in the PSF Lead to "Ringing" in The Reconstructed Image

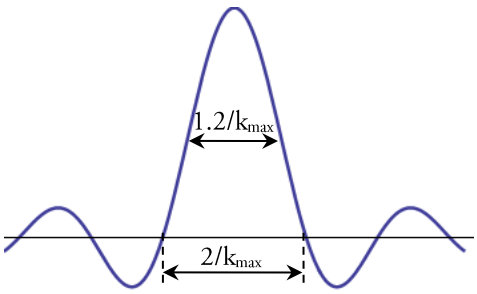
The wiggly edges of the PSF are responsible for an artifact often referred to as Gibbs ringing, observed as "wiggles" upon transitioning from one intensity sharply to another:



Since the wiggles in the PSF become smaller as $N\to\infty$ while keeping the FOV fixed, one way to reduce it (but not completely eliminate it) is to simply take more points. However, some wiggles will remain even for large N, and any sharp edges/transitions in $M_0(\mathbf{r})$ will result in Gibbs ringing even in very high resolution images.

Why The Nominal Resolution Tells Only "Half The Story"

Suppose we image in one dimension with the following parameters: FOV = 10 mm and N=10, meaning our **nominal resolution** is $\Delta x=FOV/N=1$ mm. However, the PSF is not a perfect boxcar with a width of 1 mm. Rather, there is no unique way to define the "resolution" of the PSF, only to say it is approximately $1/k_{max}$. Some parameters of the main lobe are shown below:



So the width at the base is actually **twice** the nominal resolution (Δx), and even the width at half the maximal amplitude is approximately $1.2\cdot\Delta x$. It is best to keep in mind these two figures of merit and not the nominal resolution itself, although almost all published papers do not mention the PSF at all and quote solely the nominal resolution. Still, it is quite easy to visualize the true form of the PSF given the nominal resolution, at least for cartesian sampling.

[2] It's not difficult to prove. You should try it if you feel comfortable with the math (you don't need to prove it to use it, though!).