EE369C: Assignment 2 Solutions

Due Thursday, Oct 1

Introduction This assignment introduces the basic operations in a gridding reconstruction algorithm.

The data set for this assignment is a simulated phantom using a small spiral acquisition with 6 interleaves of 1536 samples. This is a typical trajectory for real-time cardiac imaging. With a repetition time T_R of 24 ms, this produces a new image every 144 ms, at a rate of 7 images/s. The k-space trajectory, pre-weighting function, and simulated data are in the file:

`http://www.stanford.edu/class/ee369c/data/rt_spiral_03.mat`

The complex k-space data is in the matlab variable `d`, and the pre-weighting function is in the matlab variable `w`. The k-space trajectory (k_x, k_y) is stored as $k = k_x + i * ky$ in the complex matlab variable `k`. `k` is scaled relative to a $\pm k_{max}$ of ± 0.5. The k-space trajectory doesn't actually reach $\pm k_{max}$, but is scaled to produce the right field-of-view when reconstructed as a 128x128 image.

We will start with a complete gridding routine that does all of the things we talked about in class. Then we'll go through the various parameters and look at what they do. The gridding routine is `gridkb.m` which is available at

`http://www.stanford.edu/class/ee369c/mfiles/gridkb.m`

The call for the function is

```
% function m = gridkb(d,k,w,n,osf,kw,opt)
%
%     d -- k-space data
%     k -- k-trajectory, scaled -0.5 to 0.5
%     w -- k-space weighting
%     n -- image size (m will be osf*n X osf*n)
%     osf -- oversampling factor (usually between 1 and 2)
%     wg -- full kernel width in oversampled grid samples (usually 3 to 7)
%     opt -- 'k-space', 'image', defaults it 'image' if not specified
%
%     m -- gridded k-space data
%     p -- gridding kernel, optional
%
%  Uses optimum Kaiser-Bessel window for a given
%    oversampling factor and kernel size
%  Now uses Phil's numbers
```

You give it the data `d`, the k-space trajectory `k`, and your estimate of the precompensation function `w`. You also specify the reconstruction parameters including the reconstruction size `n`, the oversampling factor `osf` (the returned image will be of size `osf*n`, you have to extract the middle n$\times$ n part yourself), and the kernel size `kw`. It will return the gridded k-space data be default, or the reconstructed, deappodized image if you specify the `'image'` option.

Given your parameter choices for oversampling factor and kernel width, it chooses the optimum kernel based on the Beatty paper.

1. Effects of Reconstruction Parameters. For this problem will take the `gridkb.m` routine, and show what the effects of some of the parameters are. The goal is that you would be able to recognize these artifacts if you saw them in your own work.

a) Simple Reconstruction This example shows the effect of a minimum (1X) field of view, along with a very simple interpolation. Reconstruct an image with a 128×128 size, an oversampling factor `osf` of 1, a kernel width `kw` of 2 (effectively linear interpolation). The call would be

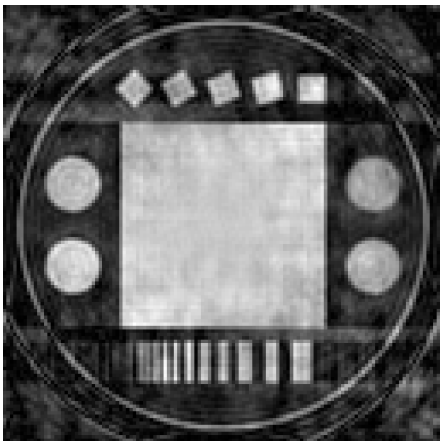
```
>> ima = gridkb(d,k,w,128,1,2,'image');
```

Display your result. Use the `imshow` command,

```
>> imshow(abs(ima),[]);
```

where the `[]` argument tells `imshow` to automatically set the image range. You can set this yourself if you'd like to whatever minimum and maximum values you'd like. For example, to set the range to 0 to 64, use `[0 64]`.

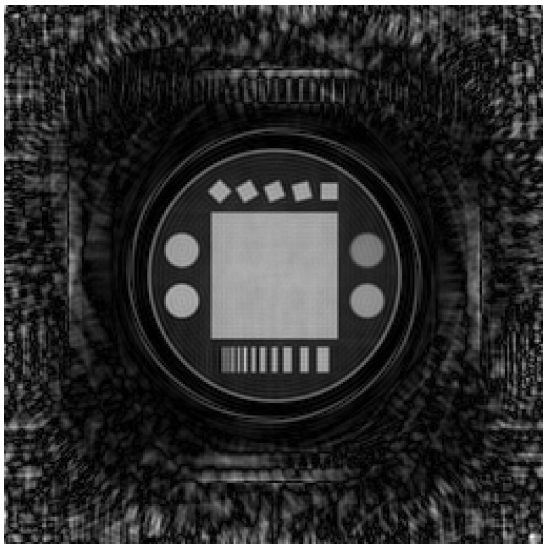
Solution The result is below:

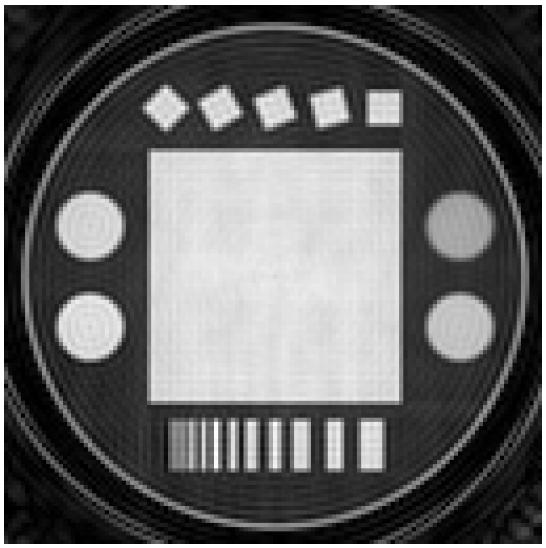


Significant aliasing is apparent.

b) Oversampling Many of the problems in the previous reconstruction can be greatly reduced just by sampling on a finer grid, even with a very simple interpolation kernel. Repeat the above, but use an oversampling factor `osf` of 2. Display your result. This should look better. Note that we only care about the middle 128×128, the rest of the field of view contains aliased energy that was folded back into the image in (a).

Solution The reconstruction for the 2X FOV, and the central 1X FOV shown in below:

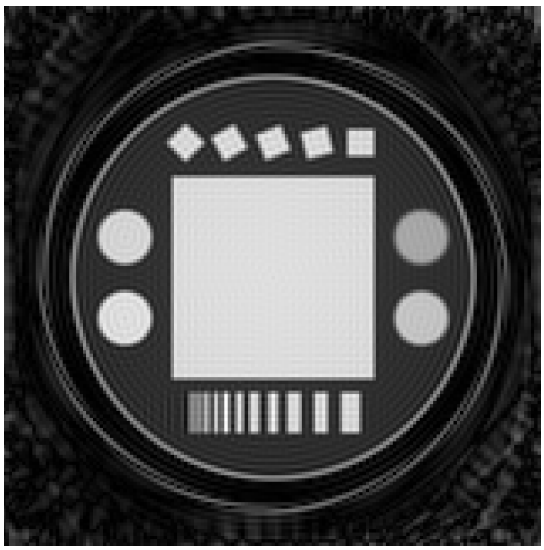


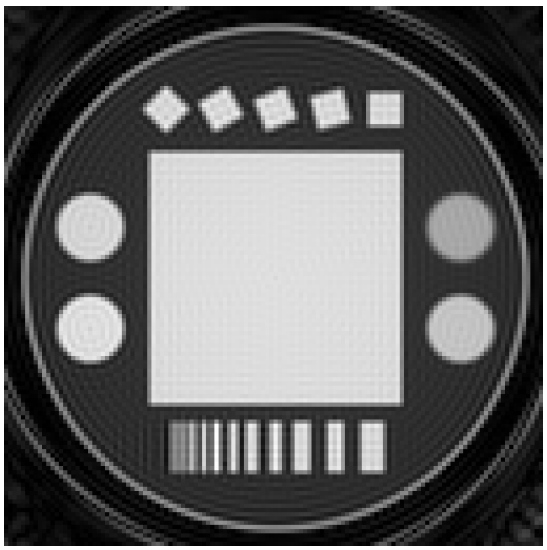


This looks much better!

c) Reduced Oversampling The problem with the 2X oversampling from part (b) is that it takes much more memory, and FFT time. If we increase the kernel size, we can reduce the oversampling ratio greatly, and still produce a high fidelity reconstruction. If we choose an oversampling factor `osf` of 1.25, how big should we choose the kernel to limit the error to 0.1%? Reconstruct the data with this kernel size and oversampling factor, and display the result.

Solution The reconstruction of the full 1.25X FOV and the central 1X FOV are shown in Fig. **??**. This used a kernel size of 5, from the Beatty paper plot.





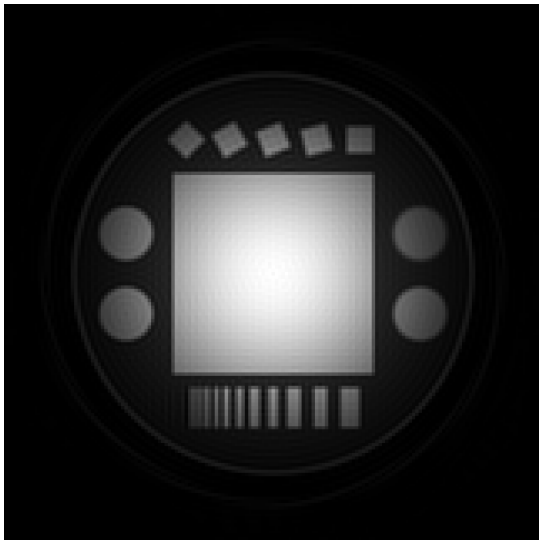
d) Deappodization The `gridkb.m` function does the deappodization when you return an image. This isn't done if you return the k-space data. Reconstruct the k-space data yourself to see what effect the deappodization has for the previous case (c).

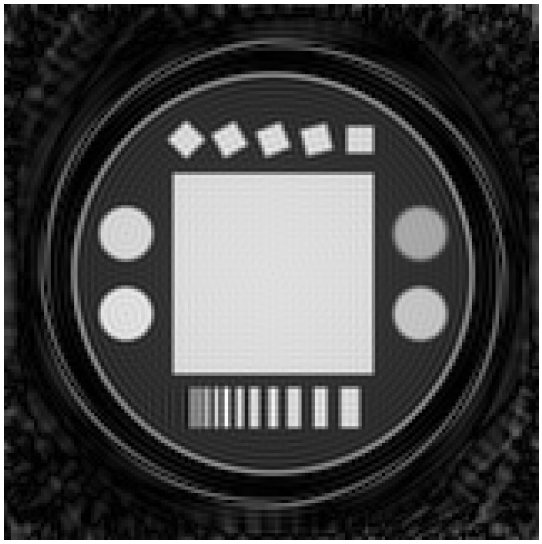
```
>> kd = gridkb(d,k,w,128,1.25,5,'k-space');
```

```
>> imd = fftshift(fft2(fftshift(kd)));
```

where I've arbitrarily set the kernel size to 5. You should use your result from (c). Display this reconstruction, and compare it to the previous one.

Solution Reconstruction without (left) and with (right) deapodization. There is a lot of apodization to fix with a 1.25X oversampling factor.





2. Density Correction Estimation with Voronoi Diagrams For this problem, we will use a variable density spiral data set of a simulated phantom. This has twice the sampling density near the origin. The data is in `var_dens.mat`, also in the `data` subdirectory. This contains the k-space trajectory `k`, and the complex data `d`. The acquisition has 2048 points per spiral interleave, and 6 interleaves. The image matrix size is 128 by 128. Use a 2X grid for the reconstruction, and the kernel width of 5 samples.

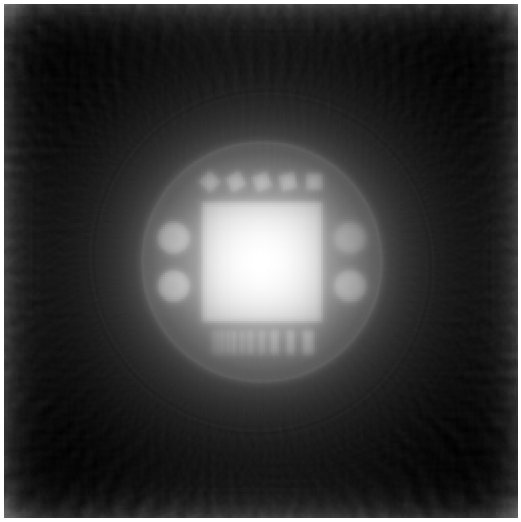
a) No Density Correction First, reconstruct the data with unity weighting (set all the values of `w` to one). Display the reconstructed image with no density compensation.

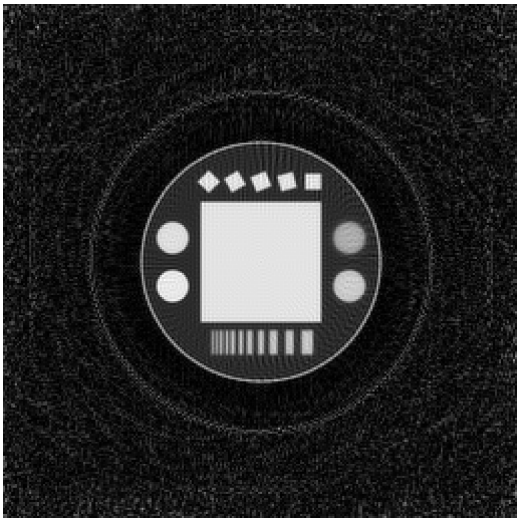
b) Voronoi Density Correction Next, use the `voronoidens.m` routine from the `mfiles` subdirectory to estimate the density on the data points using the Voronoi diagram. This routine returns the area associated with each sample, which is the weighting function (*i.e.* 1/density). You also need to devise a method for assigning reasonable values to the k-space samples at the edge of the sampled region, since the `voronoidens.m` routine returns areas of `NaN` for these samples.

Display the resulting density compensated image. Note that the backround inside the circular ring of the phantom has a DC level of 20% of the peak, and is not zero (i.e., this is not an artifact!).

Note: There are a few streaks in the reconstruction of this data set that may not go away. These are subtle, but still noticeable.

Solution: The reconstruction with and without the Voronoi compensation are shown below:





The density at the end was held at the last reasonable value before it blew up, which was 1e-4.