

EE369C: Assignment 7

Due: Optional

Reconstruct the parallel projection data set in `ct_data_5.mat` in the directory `http://ee369c.stanford.edu/restricted/ct_data`. This is actually data that has been reprojected from a reconstructed fan beam data set. It only has 200 projections, so it doesn't take too long to reconstruct. The data was provided by Prof. Joergen Arendt Jensen of the Technical University of Denmark. The data set has 512 samples for each of 200 full projections acquired over 180 degrees.

1. Parallel Beam Filtered Backprojection

a) Display the sinogram of the data set.

b) Compute the filtered projection data by zero padding in projection direction by 2-to-1, transforming, multiplying by the rho filter, and inverse transforming. Display the sinogram of the filtered projection data.

c) Backproject the filtered projection data onto a 256x256 matrix. The 512 sample projections have already been interpolated up from the 128 samples that this number of projections supports, so you don't need additional interpolation to avoid fractional pixel artifacts. Assume that the 512 projection samples fill the 256 sample FOV.

Backprojection can be performed efficiently in matlab using subscripting. There are many ways to do this. Let the matrix `fpm` contain all the filtered projections as columns. If `fp` is one filtered projection, the basic idea is to compute an `n` by `n` matrix `ndx` whose value is the subscript into the projection for each sample at that angle. With this, the backprojection becomes

```
>> im = zeros(n,n);
>> for jj=1:number_projections,
>>     proj_angle = (jj-1)*pi/number_projections;
>>     ndx = compute_indices(proj_angle,n);
>>     fp = fpm(:,jj);
>>     fp(1) = 0;
>>     fp(length(fp)) = 0;
```

```
>> im = im+fp(ndx);
>> end;
```

One option for `compute_indices` is to first define a matrix `xy`

```
>> y = [=n/2:n/2-1]/(n/2);
>> x = y';
>> xy = x*ones(1,n)+i*ones(n,1)*y;
```

where `n` is the size of the reconstruction matrix. The real part of this matrix is the `x` offset of each sample in the matrix. If we multiply the matrix by `exp(i*a)` and then take the real part, we get the offset of each sample at the angle `a`. All we need to do is scale it to the size of the projection data. Indexes off either the end of the projection are set to ends of the projection, and these values of the projection are set to zero for this purpose.

2. Frequency Domain Interpolation Do the same reconstruction using your gridding reconstruction. Generate a `k-space` and weighting file based on the class notes.

3. Fan Beam Filtered Backprojection Most CT data is actually collected using a fan beam geometry. Modify your parallel beam backprojection routine to reconstruct fan beam data.

A fan beam data set from a commercial GE scanner is in `fan_beam_1.mat` in http://ee369c.stanford.edu/restricted/ct_data. This has 984 views of 888 samples over an angle range of 2π . The fan beam angle is ± 0.4787175 radians, with a beam spacing of 0.0010788 radians. This data was provided by Sarah Patch of the Applied Sciences Laboratory at GE Medical Systems.

There are two common ways of doing fan beam reconstruction. One is a modification of filtered backprojection. The fan beam data is first weighted by $\cos(n\Delta\gamma)$, where $n\Delta\gamma$ is the angle of the n^{th} beam in the fan. The projection data is then filtered. To a good approximation, this is the same as the rho filter in the parallel beam algorithm. The filtered projection data is backprojected along the fan beams weighted by $(D/d(x,y))^2$ where D is the distance from the source to the middle of the image, and $d(x,y)$ is the distance along the beam from the source to each pixel. Backprojecting along the fan beams can be performed very similarly to the parallel beam case, where `compute_index()` is now based on `angle(D-xy*exp(i*a))`. The weighting function is `D./(abs(D-xy*exp(i*a)).^2)`.

This was covered in class, and is described in detail in Chapter 3 Section 4 of the Kak and Slaney book.