BME 493/593, Spring 2019: Computational Methods for Imaging Science Final Project (due May 3, 2019)

Submission instructions. Final projects are due on **Friday, May 3rd at 11:59pm**. Submission will be accepted on Canvas, **one single submission for each team**. Each submission should include:

- The final report in **PDF format**.
- **MATLAB or Python codes** to reproduce the results presented in the report. Code should be **runnable** and documented.
- A **readme file** that explains how to run the code and reproduce all the results in the report.

Do not wait till the last minute for this project. The image reconstruction times can be long, especially if you use your laptops to reconstruct.

Project goals. The goal of this project is to reconstruct X-ray computed tomography (CT) images from an experimental data set. The imaging model corresponds to a discrete two-dimensional Radon transform, see e.g. Section 7.4.1 of A. C. Kak and M. Slaney, *Principles of Computerized Tomographic Imaging*, SIAM, 2001.

The region of interest is a square slice of dimensions $61.44mm$ by $61.44mm$. The measured full view sinogram corresponds to 540 projections, taken in increments of $(5/12)^o$ from 0^o (included) to 225^o (excluded). Each projection consists of 512 rays over a distance equal to $61.32mm$. Note that the forward model does not exactly match the true measurement conditions, due to both model and measurement errors. This is inevitable, at least to some degree. Keep this sources of error in mind when designing your reconstruction algorithms.

The following studies should be performed:

1. Reconstruction by use of all available tomographic views (540):
   (a) Reconstruct the image using a stationary iterative method and iterative regularization.

(b) Design an optimization problem that includes a regularization strategy that is both reasonable and non-smooth. Justify your design choices.

(c) Design an algorithm to solve the optimization problem in 1(b).

   i. Provide pseudocode for the algorithm.

   ii. Describe how all the parameters (the regularization parameter(s) and any other possible parameters you have introduced for the algorithm) will be determined.

   iii. Describe the stopping rule for the iterations. Note that you can't use the reference image when determining your stopping rule.

(d) Implement the algorithm. (You MUST submit your code)

(e) Display the reconstructed image. Extract the central row of the image and plot the values. Do the same for the reference image and display the two plots on the same figure.

2. Reconstruction by use of 270 uniformly spaced tomographic views, and repeat all steps of problem 1.

3. Reconstruction by use of 90 uniformly spaced tomographic views, and repeat all steps of problem 1.

Report. Prepare a report that summarizes all of the above studies. Compare and discuss differences between reconstructed image MSE and rate of convergence observed in the three studies, as well as anything else of interest. The report should follow a laboratory report format. It should contain (but not restricted to) the following sections:

1. Introduction/background (about 1 page)

2. Methods (between 2 and 4 pages, should include pseudocode for the algorithms)

3. Results (about 1 page of text, as many figures as needed)

4. Discussion (about 2 pages)

5. Conclusions (two or three paragraphs)

6. Bibliographical references (as needed)

The report will be graded based on design of regularization term, robustness of reconstruction algorithm, implementation correctness, quality of reconstructed image, clarity of exposition, proper use of the English language, and depth of discussion.

Understanding the data and setting up the imaging operator. The provided Matlab file `project_data.mat` contains two variables

1. `sinogram`: a number of rays (512) by number of views (540) matrix containing the measured X-ray CT data;
2. `imgref`: a 256 by 256 reference image[1].

To construct the imaging operator use the `paralleltomo.m` code in Assignment 2. The Matlab code below shows how to create the imaging operator $\mathbf{A}$ in the full view case.

```
% Size of the region of interest (unit: mm)
L = 0.06144;
% Number of pixels in each direction
npixels = 256;
% Pixel size
pixel_size = L/npixels;

% Numer of views
nviews = 540;
% Angle increment between views (unit: degree)
dtheta = 5/12;
% Views
views = [0:nviews-1]*dtheta;

% Numer of rays for each view
nrays  = 512;
% Distance between first and last ray (unit pixels)
d      = npixels*(nrays-1)/nrays;

% Construct imaging operator (unit: pixels)
A = paralleltomo(npixels, views, nrays, d);
% Rescale A to physical units (unit: mm)
A = A*pixel_size;
```

If you wish to use Python for your implementation, save $\mathbf{A}$ to file and read it back in Python using `scipy.io.loadmat`.

[1] If you are feeling ambitious, you can try to reconstruct a 512 by 512 image.