EEE 475/575 Medical Image Reconstruction & Processing
Homework 3
27 April 2020, Monday at 23:59

GUIDELINES FOR HOMEWORK SUBMISSION

Instructions:

1. NO submission via E-MAIL (all emails will be discarded).
2. NO submission of RAR or ZIP files (rar or zip files will be discarded).
3. You should upload your solutions to Moodle as two separate files. One PDF file (should be named name_surname_id_hw#.pdf), and one MATLAB m file (should be named name_surname_id_hw#.m).
4. Submission system will remain open for 1 day after the deadline. No points will be lost if you submit your assignment within 12 hours of the deadline. There will be a 50% penalty if you submit after 12 hours but within 24 hours past the deadline. No submissions beyond 24 hours past the deadline.

MATLAB File Guidelines

1. It should be a single m file containing the codes for all questions (if you upload many m files, we will not evaluate any of them).
2. There is a template on Moodle to help you organize your solution in one m file, you should use it in all HWs.
3. Read the guidelines in the template file and follow each and every step.
4. If you do not upload your m file, your homework will not be evaluated (i.e., 0 pts). There will not be any exceptions to this rule.
5. If your m file gives runtime errors, your homework will not be evaluated (i.e., 0 pts). There will not be any exceptions to this rule.

PDF Report Guidelines

1. You get your points mainly from the PDF report file. The report should be typeset (no handwriting allowed). The PDF file should contain all results and plots. Unclear presentation of results will be penalized heavily. No partial credits to unjustified answers.
2. Maximum file size that can be uploaded to Turnitin is 40 MB. Pay attention to this before submitting your file.
3. You should properly name your pdf file and upload it to Turnitin upload area. We will discard all improperly named submissions (i.e., 0 pts).
4. After each question or each part you should properly display your MATLAB results, plots and MATLAB codes for that part (zero points for missing outputs).

PREPARATION

From the Moodle page of the class, download the following files:

- **SM_40x40.mat:** Time-domain system matrix calibration measurements for a simulated MPI system with 40×40 grid size. This file contains two matrices (SM_coil1 and SM_coil2) that contain time-domain measurements from two different coils. Each matrix has size 10001×1600, where the first index is for time, and the second index is for position (i.e., 40×40 = 1600). For example, SM_coil1(:,5) contains the calibration measurement signal received by Coil #1 when a point source object was placed at 5th grid location.
- **measurement_phantom.mat:** Time-domain imaging measurements for a simulated phantom. This file contains two vectors (meas_coil1 and meas_coil2) that contain measurements from two different coils, when the object of interest was placed in the FOV and scanned using the same imaging trajectory that was used in the calibration measurement. Each vector has size 10001×1. The phantom itself is also given as a matrix with size 280×280. Using the system matrix, we will try to resolve a 40×40 image of this phantom.
- **su_openmpi.mat:** This file contains *S* and *u* vectors from an actual experimental MPI dataset. This data belongs to a 3D imaging sequence with 37×37×37 grid size, where the signal is acquired by two different coils. The system matrix *S* has size 3056×50653, where the first index is for frequency, and the second index is for position (i.e., 37×37×37 = 50653). The measurement vector *u* has size 3056×1.

PART I – SYSTEM MATRIX AND SVD (45 pts)

1.1) Preparing System Matrix: Load SM_40x40.mat. This file contains system matrix calibration measurements in time-domain. Working in frequency-domain in more advantageous in MPI, as the signal is sparse in frequency domain. For both coils, plot the magnitude spectrum for calibration measurements at three different grid locations. For example, for Coil #1 and 5th grid location,

```
>> plot(abs(fftc(SM_coil1(:,5))))
```

As you will see, the spectrum contains signal at certain discrete frequencies. These are the higher harmonics of the drive field frequency that was used to induce the nanoparticle signal in MPI.

Next, form a system matrix *S*. The nth column of *S* should contain the following:

1) Take the centered FFT of the calibration measurement from Coil #1 and grid location n. The result will be a vector of size 10001×1.
2) Since the calibration measurement is real-valued, the spectrum has conjugate symmetry property. Therefore, only half of the spectrum contains "nonredundant" information. Extract the last 5001 points to reach a vector of size 5001×1.
3) We know that the image in MPI is real-valued. Therefore, it makes things easier if the system matrix is also real-valued. To do this, concatenate the real and imaginary parts of the vector from Step 2 to have a vector of size 10002×1.
4) Repeat Steps 1-3 for Coil #2.

5) Concatenate the resulting vectors from Coil #1 and Coil #2, to reach a vector of size 20004×1. Place this vector in nth column of *S*.
6) Repeat Steps 1-5 for all grid locations.

In the end, the system matrix *S* should have size 20004×1600, and should be purely real-valued. Plot three different columns of the system matrix. Save the system matrix *S* in a .mat file, to be used in the following questions.

1.2) Preparing Measurement Vector: Load measurement_phantom.mat. This file contains time domain imaging measurements for a simulated phantom. First, display the phantom that is given as a matrix with size 280×280. This phantom shows the spatial distribution of nanoparticles in the FOV, where the pixel intensity denotes nanoparticle concentration. We will use this image as the "*reference*" image for the rest of this homework. When comparing the reconstructed images with the reference, **do NOT normalize** any of the images or the reference. System function reconstruction is supposed to give images that are quantitative, i.e., pixel intensity denotes nanoparticle concentration.

Prepare the measurement vector *u* following Steps 1-5 in Question 1.1. In the end, *u* should have size 20004×1. Plot the resulting measurement vector *u*. Save the measurement vector *u* in a .mat file, to be used in the following questions.

1.3) SVD: Compute the compact Singular Value Decomposition (SVD) of S as follows:

```
>> [U,Sigma,V] = svd(S,'econ');
```

Here, Sigma is a diagonal matrix of size 1600×1600 that contains the singular values of *S*. *U* is of size 20004×1600 and *V* is of size 1600×1600.

Plot the singular values (i.e., the diagonal entries of Sigma). Note that the singular values are ordered from largest to smallest. Compute the condition number of *S* using these singular values, i.e., $Cond(S) = \frac{\sigma_{max}}{\sigma_{min}}$.
Hint: Check that you get the same condition number using the built-in *cond* function of MATLAB.

1.4) Reconstruction via SVD: Compute the image *c* using Moore-Pensore pseudo inverse from SVD, i.e., $c = V\Sigma^{-1}U^{H}u$. Here, *c* will be a vector of size 1600×1. Reshape this vector into a 2D image of size 40×40, and then resize it to size 280×280 so that we can compare it to the reference image.

```
>> ima = reshape(c,40,40);
>> ima = imresize(ima, [280 280]);
```

MPI images cannot have negative values, since the pixel intensity represents the nanoparticle concentration. Therefore, set all of the negative pixel values in ima to zero at the last step (do this for **ALL** reconstructions in this homework). Display the resulting image. Display the error image (i.e., the magnitude of the difference between the reconstructed image and the reference image). Compute PSNR and SSIM. Comment on the quality of the reconstruction.

1.5) Truncated SVD: Truncate the singular values such that the condition number is approximately 100. Accordingly, you will keep the largest N singular values, such that $\frac{\sigma_1}{\sigma_N} \approx 100$. You will then keep the first N columns of U and V. In the end, truncated version of Sigma will have size N×N, truncated version of U will have size 20004×N, and truncated version of V will have size 1600×N.

Using these truncated versions, compute the image following the same steps as in Question 1.4. Display the image and error image. Compute PSNR and SSIM. Comment on the improvements on the image.

1.6) Repeat the steps in Question 1.5 for the following condition numbers: [5, 10, 20, 30, 50]. At each condition number, display the image and error image, compute PSNR and SSIM. Next, plot PSNR and SSIM as functions of condition number. Comment on the image quality as a function of condition number. What is the condition number that yields the best image quality?

1.7) Filtered SVD: We will now filter the singular values using the filter values:

$$y_i = \frac{\sigma_i^2}{\sigma_i^2 + \lambda}$$

where λ is a regularization parameter. This is equivalent to replacing the singular value σ_i by:

$$\frac{\sigma_i}{y_i} = \frac{\sigma_i^2 + \lambda}{\sigma_i} = \sigma_i \left(1 + \frac{\lambda}{\sigma_i^2}\right)$$

Here, λ should be such that $\lambda \ll \sigma_i^2$ for the largest singular values [1], so that it does not affect them. On the other hand, λ should be such that $\lambda \gg \sigma_i^2$ for the smallest singular values, so that the inversion of the small singular values does not cause noise amplification.

Based on the singular values that you plotted in Question 1.3, what is a reasonable range for the parameter λ? Explain your answer.

To start with, choose $\lambda = \sigma_1 \sigma_N$ (i.e., the product of the smallest and largest singular values). Plot the filtered singular values. Also plot on the same axis the original singular values, for comparison. Compute the image for filtered SVD using this λ. Display the image and error image. Compute PSNR and SSIM. Comment on the result.

1.8) Try a few different λ values in the reasonable range. Plot PSNR and SSIM as functions of λ. What is the optimal λ value, λ^* (approximately)? Use PSNR and SSIM, as well as visual inspection, to determine λ^*. Show your steps. Then, display the following results for the cases of $\lambda^*/10$, λ^*, and $10\lambda^*$: the image, error image, PSNR and SSIM. Comment on the image quality as a function of λ.

---

[1] Note that there was a typo in both the lecture notes and the Book Chapter posted on Moodle, as both said λ should be such that $\lambda \ll \sigma_i$ for the largest singular values.

1.9) L-curve: In a real-life imaging scenario, we do not know what the “reference” image is, so λ^* cannot be determined using PSNR or SSIM. One method to determine λ^* is to plot the L-curve for residual norm vs. solution norm (i.e., $\|Sc - u\|_2$ vs. $\|c\|_2$). The optimal value for λ is typically the one that yields a result at the corner of this L-curve. Plot this L-curve by trying a few different λ values in the reasonable range. Make sure you capture a decent part of the L-curve (and not just a small section of it). What is λ^* according to the L-curve? For this λ^*, display the image, error image, PSNR and SSIM.

PART II – KACZMARZ METHOD (40 pts)

2.1) Row-norm Thresholding: Kaczmarz inherently normalizes the rows of the system matrix. During that normalization, rows that have close to zero energy can cause noise amplification problems. To avoid this problem, we will first perform thresholding to remove the frequency components (i.e., the rows) of the system matrix with small energy. For this, first compute the norm of each row of the system matrix. Plot the norms (i.e., row number vs. row norm). Then, discard the rows of *S* that have norm smaller than 50. Likewise, discard the corresponding entries of u. What are the sizes of *S* and *u* after this thresholding?

2.2) Standard Kaczmarz Method: First, apply a row-norm threshold of 50 (i.e., use the matrices and vectors from Question 2.1). Implement standard Kaczmarz method[2], without any regularization or relaxation. Make sure that you randomize the order of the rows during sub-iterations. For the outputs of the first 10 iterations (NOT sub-iterations) of standard Kaczmarz: display the image, error image, PSNR and SSIM.

2.3) Reconstruct the image using standard Kaczmarz with row-norm thresholds: [1 10 30 50 100]. For the output of the 10^{th} iteration of each case: display the image, error image, PSNR and SSIM. Which of these thresholds gave the best image quality? Comment on the image quality as a function of row-norm threshold.

2.4) Regularized Kaczmarz Method: First, apply a row-norm threshold of 1. Then, implement the regularized Kaczmarz method [3], incorporating a regularization parameter λ. Make sure that you randomize the order of the rows during sub-iterations. Note that regularized Kaczmarz is an iterative solver for the following problem:

$$(S^H S + \lambda I)c = S^H u$$

Inserting the SVD decomposition of $S = U\Sigma V^H$, this can be shown to be equivalent to:

$$V(\Sigma^2 + \lambda I)V^H c = V\Sigma U^H u$$

---

[2] See Slide#33 on “Linear System of Equations” slides.

[3] See Slide#39 on “Linear System of Equations” slides. Also see Algorithm 2 on Page#145 of the Book Chapter posted on Moodle.

After skipping a few steps, this can be shown to be equivalent to:

$$c = V\tilde{\Sigma}^{-1}U^H u$$

where $\tilde{\Sigma}$ is a diagonal matrix with

$$\tilde{\Sigma}_{i,i} = \frac{\sigma_i^2 + \lambda}{\sigma_i}$$

Note that the diagonal entries of $\tilde{\Sigma}$ contain the filtered singular values, as given in Question 1.7. This proves that solving the regularized least squares problem is equivalent to solving the filtered SVD case. Importantly, these derivations give us an insight into the reasonable range of values for the parameter λ (i.e., determined similarly as in the filtered SVD case).

To start with, choose $\lambda = \sigma_1\sigma_N$. For the outputs of the first 10 iterations of regularized Kaczmarz: display the image, error image, PSNR and SSIM. Comment on the improvements in this image when compared to the result without regularization (but with the same row-norm threshold of 1).

2.5) One approach to choose λ is to define it relative to σ_1^2, i.e., $\lambda = \sigma_1^2\,\lambda_{rel}$. This way, we know that we should choose $0 < \lambda_{rel} < 1$. Reconstruct the image using regularized Kaczmarz for several different λ values, such that $\lambda_{rel} = [10^{-4}\ \ 10^{-3}\ \ 10^{-2}\ 10^{-1}]$. For the output of the 10th iteration of each case: display the image, error image, PSNR and SSIM. Which of these λ values gave the best image quality? Comment on the image quality as a function of λ.

2.6) For the λ value that gave the best image quality in Question 2.5, apply regularized Kaczmarz for row-norm thresholds of [1 10 30 50 100]. For the output of the 10th iteration of each case: display the image, error image, PSNR and SSIM. Which row-norm threshold gave the best image quality? Comment on the image quality as a function of row-norm threshold.

2.7) Effects of Measurement Noise: So far, the measurement vector *u* did not contain any noise. Add white Gaussian noise with zero mean and standard deviation 50 to each entry of the measurement vector *u*. Then, for the λ value that gave the best image quality in Question 2.5, apply regularized Kaczmarz for row-norm thresholds: [1 10 30 50 100]. For the output of the 10th iteration of each case: display the image, error image, PSNR and SSIM. Which row-norm threshold gave the best image quality? Comment on the image quality as a function of row-norm threshold in the presence of noise. What is the effect of noise on the choice of row-norm threshold?

PART III – OPEN MPI DATA (15 pts)

3.1) In the previous parts, we used a simulated dataset. Here, we will work with an actual experimental MPI dataset, downloaded from the Open MPI database [4]. Load su_openmpi.mat. The system matrix and measurement vector have already been preprocessed and prepared in proper format. Plot the singular values of the system matrix S. Note that the size of S is relatively large, so loading the .mat file and computing the SVD may take a while. Compute the condition number of S using these singular values, i.e., $Cond(S) = \frac{\sigma_{max}}{\sigma_{min}}$. Compare the condition number of this experimental system matrix to that of the simulated system matrix in the previous parts. What will be the implications of this?

3.2) Reconstruct the image using regularized Kaczmarz with $\lambda = \sigma_1\sigma_N$. Here, the reconstructed vector c needs to be reshaped into a 3D image of size 37×37×37:

```
>> ima = reshape(c,37,37,37);
```

At the end of the 10th iteration, display all 37 slices of the reconstructed image (assume that the third index of ima is the slice index). To display all 37 slices as a "montage" of images (i.e., concatenated slices), you can use the following command:

```
>> montage(reshape(ima,[37, 37, 1, 37]),'displayRange',[]);
```

3.3) Reconstruct the image using regularized Kaczmarz with $\lambda = 1$. Display all 37 slices of the reconstructed image (assume that the third index is the slice index), at the end of the 10th iteration. Comment on the quality of the image with respect to that in Question 3.1.

3.4) Reconstruct the image using regularized Kaczmarz with $\lambda = 10^{10}$. Display all 37 slices of the reconstructed image (assume that the third index is the slice index), at the end of the 10th iteration. Comment on the quality of the image with respect to that in Questions 3.1 and 3.2.

---

[4] https://magneticparticleimaging.github.io/OpenMPIData.jl/latest/datasets.html