EEE 475/575 Medical Image Reconstruction & Processing
Homework 4
11 May 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 file:

- **multicoil-data.mat:** Contains fully-sampled 2DFT data acquired using an 8-channel coil, together with the estimates for the spatial sensitivity profiles for each coil channel. im(Nx,Ny,Nc) contains the fully-sampled images for each channel, map1(Nx,Ny,Nc) contains the estimated sensitivities for each channel, map2(Nx,Ny,Nc) contains a second set of estimated sensitivities for each channel. Here, Nx = 224 and Ny = 160 are image dimensions, and Nc = 8 is the number of coils.

PART I – SENSE (50 pts)

First, we will implement **SENSE reconstruction** for parallel-imaging with Cartesian data sets. For simplicity, assume that the noise in each channel is independent and identically distributed with unit variance (i.e., $\Psi = I$), so that we can ignore the noise covariance matrix Ψ in our computations and derivations.

1.1) Display the Data: Display the magnitude and phase of the images for all coils. Adjust the display windowing appropriately to see intracranial tissue (i.e., the brain tissue). Display the corresponding k-space spectrums for all coils (see Homework #1 for details on displaying the spectrum). Display the magnitude and phase of the coil sensitivities for all coils (both for **map1** and **map2**). Comment on the results.

Note: Throughout this homework, use the "montage" function (see Homework #3 for details) whenever you need to display images from all coils in a compact form.

1.2) Multi-coil Reconstruction: Implement two different methods for multi-coil reconstruction from full k-space data. The first method should take the square root of the sum-of-squares (SoS) combination across coil channels. The second method should find the optimally weighted linear combination (OLC) across coil channels. In this method, when $\Psi = I$, the weight for each channel is equal to the ratio of the complex conjugate of the respective channel sensitivity to the sum-of-squares combination of the sensitivities across all channels (as covered in class).

Display the magnitude images reconstructed by SoS and OLC. Perform OLC using **map1** and **map2**, separately. For each image, adjust the display windowing appropriately to see intracranial tissue. Ideally the OLC method should improve SNR slightly. We will use the OLC result from **map1** as the *reference image* in Part I. Compute PSNR and SSIM for SoS. Compute PSNR and SSIM for OLC from map2. Comment on the results.

Note 1: The coil sensitivities are zero in the background regions, which may yield NaN pixel values during reconstruction. Set all NaN pixels to zero as the last step of the reconstruction. Do this for all reconstructions that yield NaN values in this homework.

Note 2: Throughout this homework, normalize the images before using PSNR and SSIM.

1.3) Generate Undersampled Images: Write a MATLAB function that takes the fully sampled images "im", and generates undersampled aliased images and the corresponding k-space data, i.e., [imu, Mu] = undersample(im, Rx, Ry). This function will take the image data, take a centered 2D FFT to compute respective k-space data. The k-space data will be **<u>removed</u>** on decimated lines in the frequency domain to yield "Mu". Lastly, the function will take an inverse centered 2D FFT to produce the aliased images, "imu". For example, for Rx=1 and Ry=4, you will **<u>remove</u>** three out of every four columns. For Rx=Ry=2 you will **<u>remove</u>** every other row and column. Hence, both "Mu" and "imu" will be of size (Nx/Rx) × (Ny/Ry) × Nc. For convenience, we will only consider the cases where Nx/Rx and Ny/Ry are even numbers.

Test your routine for (Rx,Ry) = (1,2), (2,1), (2,2), (1,4), (4,1). Display the magnitude of the images and the corresponding k-space spectrums for all coils. Describe what you see in the images.

1.4) SENSE Reconstruction: In class, we covered the unregularized version of SENSE, where the resolved source voxels can be expressed as follows (for $\Psi = I$):

$$\underline{\widehat{m}} = (C^H C)^{-1} C^H \underline{m_s}$$

Here, $\underline{\widehat{m}}$ is the vector of resolved source voxels, C is the matrix of coil sensitivities at the source voxels, and $\underline{m_s}$is the vector of aliased voxel intensities from all coils. A simple improvement of this reconstruction is to incorporate l_2-regularization as follows:

$$\underline{\widehat{m}} = (C^H C + \lambda I)^{-1} C^H \underline{m_s}$$

As we have seen previously in linear system of equations, the addition of an identity matrix improves the conditioning of the problem, and hence regularizes the reconstruction.

Write a function that takes the coil sensitivity maps and the aliased, undersampled images from the previous part, and computes the l_2-regularized SENSE reconstruction of the image, im_sense = l2sense(imu, map, Rx, Ry, lambda). Carefully consider the indices of source voxels when compared to the aliased voxel index.

As a test that your function works, first reconstruct for Rx=Ry=1, λ=0 (<u>use only **map1** for this question</u>). Display the resulting magnitude image, error image (i.e., the magnitude of the difference between the image and the reconstructed image), and compute PSNR and SSIM. Verify that this gives the exact same reconstruction as the OLC method (up to numerical computation error).

1.5) Next, reconstruct for (Rx,Ry) = (1,2), (2,1), (2,2), (1,4), (4,1) for the case of no regularization (i.e., λ = 0). For each case, display the resulting magnitude image and error image, and compute PSNR and SSIM. Comment on the results. (<u>Use only **map1** for this question</u>)

1.6) For (Rx,Ry) = (1,2), (2,1), (2,2), (1,4), (4,1), select a suitable regularization weight λ for each case that achieves improved artifact suppression without creating new artifacts. For each case, display the resulting magnitude image and error image, and compute PSNR and SSIM. Comment on how λ affects the results. Comment on how the image quality changes with increasing acceleration rates. Compare the image qualities for (Rx,Ry) = (1,2) vs. (2,1). Also, compare the image qualities for (Rx, Ry) = (2,2) vs. (1,4) vs. (4,1). (Use only **map1** for this question)

1.7) Importance of Accurate Coil Sensitivities: SENSE reconstruction is very sensitive to the accuracy of coil sensitivity maps. Using the l2sense.m function that you wrote in the previous part, compute SENSE reconstruction based on the second set of sensitivity maps, **map2**. Reconstruct the images for (Rx,Ry) = (1,2), (2,1), (2,2), (1,4), (4,1). For each case, use the same λ that you did in Q1.6. Display the resulting magnitude image and error image, and compute PSNR and SSIM. Compare your results to those from Q1.6. Explain the differences in reconstructions based on the coil sensitivity maps, which you displayed in Q1.1.

1.8) *g*-Factor for *l*$_2$-regularized SENSE: One of the most important characteristics of SENSE reconstruction is the geometry factor *g*, which tells us how well conditioned the reconstruction problem will be. Using the mathematical formulations that we covered in class, we can go one step further to derive the *g*-factor for *l*$_2$-regularized SENSE.

As given in class, if

$$Cov(\underline{m}_s) = \Psi$$

then

$$Cov(A\underline{m}_s) = A\Psi A^H$$

In this homework, we are assuming that $\Psi = I$. For the unregularized version of SENSE,

$$A = (C^H C)^{-1} C^H$$

which then yields

$$Cov(\underline{\widehat{m}}) = AA^H = (C^H C)^{-1}$$

Using this result, we derived the following expression for the *g*-factor:

$$g = \sqrt{[C^H C]_{i,i} [AA^H]_{i,i}} \quad (1)$$

Here, the first term in the square root is the inverse of the covariance from OLC reconstruction of the fully sampled case, and the second term is the covariance from SENSE reconstruction.

Now, for *l*$_2$-regularized SENSE, we have

$$A = (C^H C + \lambda I)^{-1} C^H$$

Derive a closed-form expression for $Cov(\underline{\widehat{m}}) = AA^H$ for this case. Then, insert this expression into Eqn. (1) to derive the *g*-factor for *l*$_2$-regularized SENSE.

1.9) *g*-Factor Map: Using the expression you derived in Q1.8, write a function that calculates the g-factor: g = gfactor(map, Rx, Ry,lambda) where map is the coil sensitivities (use only **map1** for this questions), and Rx and Ry are the acceleration factors in the x and y dimensions.

Compute and display the g-factor maps for (Rx,Ry) = (1,2), (2,1), (2,2), (1,4), (4,1) with the corresponding λ values that you used in Q1.6. Display the results on a scale of [0 5] for each case, so that you can visually compare all cases. Comment on the results. Explain whether they match your expectations based on the results in Q1.6.

Note: If ALL of the original sensitivity maps have a value zero at a voxel, set *g*-factor to 0 for that voxel. You may find complex-valued *g*-factor due to computational round-off errors before the square-root operation. In any case, the imaginary part should be very small. Use only the real part of the computed *g*-factor.

PART II – GRAPPA (50 pts)

Now, we will implement **GRAPPA reconstruction** for parallel-imaging with Cartesian data sets. Same as in Part I, we will assume $\Psi = I$. Note that for GRAPPA, we do not need external coil sensitivities. So, we will only use im(Nx,Ny,Nc).

2.1) Calibration Images: In practice, it is challenging to accurately measure coil sensitivities since the maps can be affected by various factors including patient configuration. Autocalibrating methods such as GRAPPA have been proposed as a solution to this problem. Write a MATLAB function that takes the fully sampled image "im", and generates calibration images and the corresponding k-space data, i.e., [imc, Mc] = imcalib(im, calibx, caliby). This function will take the image data "im", take a centered 2D FFT to compute respective k-space data. The k-space data will be **zeroed out** (i.e., replaced with zeros) everywhere, with the exception of a calibration region of size calibx × caliby at the center of k-space that will be left intact. These operations will yield "Mc". Lastly, the function will take an inverse 2D FFT to produce the calibration images, "imc".

To get a sense of how the calibration region in GRAPPA incorporates coil sensitivity information, test your routine for (calibx, caliby) = (8,8), (16,16), (32,32). Display the magnitude of the images and the corresponding k-space spectrums for all coils. Describe what you see in the images.

2.2) Undersampled Images with Calibration Region: Write a MATLAB function that takes the fully sampled images, and generates undersampled aliased images with calibration region and the corresponding k-space data, i.e., [imu, Mu] = undersamplecalib(im, Rx, Ry, calibx, caliby). This function will take the image data, take a centered 2D FFT to compute respective k-space data. The k-space data will be **zeroed out** (i.e., replaced with zeros) on decimated lines in the frequency domain, with the exception of a calibration region of size calibx × caliby at the center of k-space that will be left intact. These operations will yield "Mu". Lastly, the function will take an inverse 2D FFT to produce the aliased images, "imu". For convenience, assume that the image size is evenly divisible by Rx and Ry.

Test your routine for (Rx, Ry) = (1,2), (2,1), (2,2), (1,4), (4,1). Use (calibx,caliby) = (32,32) for all cases. Display the magnitude of the images and the corresponding k-space spectrums for all coils. Describe what you see in the images.

2.3) GRAPPA Kernel Weights: From this point on, for simplicity, we will only consider cases where Rx = 1 and we will implement a GRAPPA reconstruction that only utilizes the nearest sample points. As discussed in class, GRAPPA reconstruction gets increasingly complicated when both Rx and Ry are greater than 1.

Write a function that takes the central calibration data in k-space to compute GRAPPA kernel weights, i.e., kernel =calibrate(Mc,Ry,lambda). Here, Mc is as computed in Q2.1 (you will only use the non-zero parts of Mc), and λ is the regularization parameter used in kernel estimation stage of GRAPPA. As discussed in class, this process depends on the sampling geometry, so Ry is also an input to this function (remember that we are assuming Rx = 1).

For example, for (Rx, Ry) = (1,2), each unacquired data point has 6 acquired neighboring samples. For that case, the output "kernel" is a 2D matrix that has a column size of (6×Nc), and there are Nc columns. For (Rx, Ry) = (1,4), each unacquired data point can be said to have 10 acquired nearest samples, though the distances to those samples depend on which unacquired row we are considering. Then, "kernel" is a 3D matrix of size (10×Nc) × Nc × (Rx-1). Here, the dimension size of (Rx-1) is because each row "sees" a different geometry, and hence must use different kernel weights.

Compute the kernel for (Rx, Ry) = (1,2), (calibx,caliby) = (32,32), λ=0. Display the magnitude of the kernel as an image (of size (6×Nc) × Nc).

2.4) Compute the kernel for (Rx, Ry) = (1,4), (calibx,caliby) = (32,32), λ=0. For each "geometry", display the magnitude of the kernel as an image (of size (10×Nc) × Nc).

2.5) GRAPPA Reconstruction: Write a MATLAB function that takes undersampled k-space data, and computes the GRAPPA reconstruction of the coil images and the corresponding k-space data, i.e., [imr, Mr] = grappa(Mu, Mc, Ry, lambda), where Mu and Mc are as computed in Q2.2 and Q2.1, respectively. Both imr and Mr are matrices of size Nx × Ny × Nc.

Perform GRAPPA reconstruction for (Rx, Ry) = (1,2), (calibx,caliby) = (32,32), λ=0. Display the magnitude images and the corresponding k-space data from all coils.

Next, perform a sum-of-squares (SoS) combination of coils and display the final image. Also display the error image, and compute PSNR and SSSIM. Here, use the SoS image from Q1.2 as the *reference image*.

Then, repeat GRAPPA reconstruction and the above steps by selecting a suitable λ that yields the minimum amount of artifact in reconstructed images. Comment on the results.

2.6) Next, perform GRAPPA reconstruction for (Rx, Ry) = (1,4), (calibx,caliby) = (32,32), λ=0. Display the magnitude images and the corresponding k-space data from all coils. Perform SoS combination of coils and display the final image. Also display the error image, and compute PSNR and SSSIM. Then, repeat these steps by selecting a suitable λ that yields the minimum amount of artifact in reconstructed images. Comment on the results.

2.7) Final Remarks: Comment on the quality of the images obtained from SENSE to those obtained via GRAPPA at matching acceleration factors. Briefly discuss the relative benefits and disadvantages of the two techniques.