Computational MR imaging
Laboratory 8: Compressed Sensing

Report is due on Wednesday before the next lab session at 23:50. Please upload your report on StudOn.

Learning objectives

- Refresh your linear algebra skills
- Apply compression transforms (e.g., wavelets) to obtain sparse representations of MR images
- Reconstruct randomly undersampled k-space data using compressed sensing approach

Sparsity/compressibility of brain images using the wavelet transform:

Medical images are generally not sparse, but they usually have a sparse representation after applying an appropriate transform. An example is the wavelet transform, which is the core transform used in the JPEG2000 standard. Wavelet coefficients are sub-band filters that hold both spatial (pixels) and frequency (k-space) information and thus they are able to represent an image with fewer non-zero coefficients.

- In this exercise we will use the Python wavelet functions: dwt2 and idwt2. These work much like fft2 and ifft2. The following shows example usage:

```
>> coeff_arr, coeff_slice = dwt2(X_recon);
>> X_recon = idwt2(coeff_arr, coeff_slice);
```

- The wavelet transform decomposes an image into low-pass (approximation coefficients) and high-pass (detail coefficients) features. Coeff_arr is approximate coefficients of wavelet transform in complex numbers. Coeff_slice stores approximation detailes. This particular code applies the Daubechies D4 wavelet transform (always a good one to start with – another good one is Haar) with periodic boundary conditions.

- Load the file data_lab8.mat and apply the wavelet transform to the fully-sampled data (kfull). The dwt2 and idwt2 functions only take real input, so you will have to split the image into its real and imaginary parts and then recombine them. Plot the magnitude of the brain image and its wavelet representation. Use a window from 0-1 for both images. Compute the l1-norm for both images. Which one is sparser?

- Compress the brain image by factors 5, 10 and 20 using the wavelet transform (hint: sort the wavelet coefficients in descending order using the sort function, compute the threshold T by finding the coefficient corresponding to $\frac{N}{Compression\ factor}$-ratio ($N$: total sample points) and hard-threshold the wavelet transform using T). Plot the compressed image (scale: 0-1) and error image (scale: 0-0.1) and compute the RMSE for each compression ratios. Which compression ratio would you choose? What would be the maximum compressed sensing acceleration?

1. Compressed sensing reconstruction using iterative soft thresholding:

- Implement the iterative soft-thresholding approach discussed in class using the 4-tap Daubechies-type wavelet transform from exercise 1.

- The variable kacc is an undersampled dataset using variable-density random undersampling. Plot the undersampling pattern and compute the acceleration factor.

- Reconstruct kacc using your iterative soft-thresholding algorithm. The algorithm requires choosing a value for lambda. A rule of thumb is to choose a threshold close to 1% of the maximum absolute value of the starting solution. Try your reconstructions with l=5%, 1% and 0.5%. Plot the initial solution, final reconstruction and corresponding error images with respect to the fully-sampled one. Compute the value of the cost function for each iteration.