Assignment 4: CS 754, Advanced Image Processing

Due: 26th March before 11:55 pm

Remember the honor code while submitting this (and every other) assignment. All members of the group should work on and understand all parts of the assignment. We will adopt a zero-tolerance policy against any violation.

Submission instructions: You should ideally type out all the answers in Word (with the equation editor) or using Latex. In either case, prepare a pdf file. Create a single zip or rar file containing the report, code and sample outputs and name it as follows: A4-IdNumberOfFirstStudent-IdNumberOfSecondStudent.zip. (If you are doing the assignment alone, the name of the zip file is A4-IdNumber.zip). Upload the file on moodle BEFORE 11:55 pm on the due date. Late assignments will be assessed a penalty of 50% per day late. Note that only one student per group should upload their work on moodle. Please preserve a copy of all your work until the end of the semester. If you have difficulties, please do not hesitate to seek help from me.

1. In this task, you will implement the compressive KSVD algorithm that we did (or will soon do) in class. The algorithm is based on the paper 'Compressive K-SVD' by Anaraki and Hughes. You should test the algorithm on the following two datasets and include the required results in your report.

   (a) Generate a dictionary $\boldsymbol{D}$ with $K = 20$ columns for signals of $p = 100$ elements each. The dictionary entries should be drawn i.i.d. from $\mathcal{N}(0, 1)$, and the columns should subsequently be unit-normalized. Generate $N = 200$ signals where the i^{th} signal $\boldsymbol{x_i}$ is a random linear combination of any 4 randomly chosen dictionary columns. Different signals should be chosen to have different supports, and the coefficients of the linear combination should be drawn from Uniform$[0, 10]$. For each $\boldsymbol{x_i}$, generate a compressive measurement vector $\boldsymbol{y_i} = \boldsymbol{\Phi_i x_i} + \boldsymbol{\eta_i}$ where each $\boldsymbol{\Phi_i}$ has size $m \times p$. Each entry of $\boldsymbol{\Phi_i}$ should be drawn i.i.d. (with 50% probability) from $\{-1/\sqrt{m}, +1/\sqrt{m}\}$. Also $\boldsymbol{\eta_i}$ represents noise from a zero-mean Gaussian distribution with standard deviation $\sigma = f \times \dfrac{1}{mN} \sum_{i=1}^{N} \|\boldsymbol{\Phi_i x_i}\|_1$ and f is a fractional value. For each $m \in \{30, 40, 50, 70, 90\}$ and each $f \in \{0.001, 0.01, 0.05\}$, compute the average relative error for reconstruction of the N signals from their compressive measurements using the compressive KSVD algorithm. The average relative error is given as $\dfrac{1}{N} \sum_{i=1}^{N} \dfrac{\|\boldsymbol{\hat{x}_i} - \boldsymbol{x_i}\|_2}{\|\boldsymbol{x_i}\|_2}$ where $\boldsymbol{\hat{x}_i}$ is the estimate for $\boldsymbol{x_i}$. Plot these error values as a chart on a scale from 0 to 1. Include a legend in the plot as well. State your observations or inferences from the chart. For the sparse coding part of the algorithm, you should run OMP either based on the Gaussian noise tail bound, or else with a fixed value of $T_0 = 5$ as mentioned in the paper, but you should be consistent (i.e. use the same method in all iterations of the algorithm).

   (b) In the second experiment, you should download the MNIST database from `http://yann.lecun.com/exdb/mnist/`.You may use ready-made code from the web to parse the file formats. From the training file, take any 600 images per digit (each image has size 28×28 and downsample each image using bilinear interpolation to size 16×16). For each such image $\boldsymbol{x_i}$, generate compressive measurements $\boldsymbol{y_i}$ using the sensing matrix as described in the previous part but with $f = 0.01$ and $m \in \{160, 180, 200, 220\}$. Now learn a dictionary $\boldsymbol{D}$ with $K = 128$ columns from these compressive measurements. In your report, display the dictionary columns reshaped as images. Create a test set of noisy compressive measurements of any 10 images per digit, from the training set (again $f = 0.01$). Reconstruct the images in the test set from their compressive measurements using the OMP algorithm with the learned dictionary $\boldsymbol{D}$. If you use the method with T_0 in OMP, then set $T_0 = 40$. Compare the performance to that of 2D-DCT for the OMP algorithm. For the comparison, report average relative errors (as defined earlier) for the reconstruction. Display a sample reconstruction for each of the 10 digits, in your report. [20+20 = 40 points]

2. Consider a signal $\boldsymbol{f} = \boldsymbol{f_1} + \boldsymbol{f_2} + \boldsymbol{\eta}$ where $\boldsymbol{f_1}$ is a sparse linear combination of cosine waves with integer frequencies (i.e. sparse in DCT basis) and $\boldsymbol{f_2}$ is a signal consisting of a small number of spikes. $\boldsymbol{\eta}$ represents noise from $\mathcal{N}(0, \sigma^2)$ where $\sigma = 0.01\times$ average value of $\boldsymbol{f_1} + \boldsymbol{f_2}$. Consider that $\boldsymbol{f}$ is a 1D discrete signal with 256 elements. Your job is to implement any technique of your choice to separate $\boldsymbol{f}$ into $\boldsymbol{f_1}$ and $\boldsymbol{f_2}$. That is, you are given only $\boldsymbol{f}$ (which is noisy) and you want to estimate $\boldsymbol{f_1}$ and $\boldsymbol{f_2}$. Experimentally study the quality of the estimation of both components (in terms of relative reconstruction error for both $\boldsymbol{f_1}$ and $\boldsymbol{f_2}$) with (a) varying σ and fixed s, and (b) varying sparsity level s with fixed σ. You may assume for simplicity that s is same for both. Include all relevant plots in your report, and mention which technique you used. Comment on these results.
Now suppose that the magnitude of $\boldsymbol{f_2}$ was k times that of $\boldsymbol{f_1}$. Study the effect of varying k on the RMSE of both signals, on the same algorithm. Again, include the relevant plot and comments in your report. You may use any ready-made CS solver - examples are your own implementation of ISTA, OMP or solvers such as L1_LS, SPGL1, YALL1, L1-MAGIC (MATLAB codes for all are freely downloadable from the web). In all cases, state how you picked the relevant parameters for the solver or algorithm. [20 points]

3. We have studied two greedy algorithms for compressive recovery in class - MP and OMP. Your task is to do a google search and find out a research paper that proposes a greedy algorithm for CS recovery that is different from OMP and MP. Write down the algorithm in your report in the form of a simple pseudo-code. State the key theorem from the paper which presents performance bounds for the algorithm, and explain the meaning of the terms involved. If there are multiple theorems, pick the one that states the strongest result. [20 points]

4. Under the usual notation, consider a vector of m compressive measurements of the form $\boldsymbol{y} = \boldsymbol{\Phi\Psi\theta} + \boldsymbol{\eta}$, where every element of $\boldsymbol{\eta}$ is drawn from $\mathcal{N}(0, \sigma^2)$, and where $\boldsymbol{x}$ is sparse. Most sensors, however, have a fixed dynamic range, i.e. they cannot store arbitrarily large values (in $\boldsymbol{y}$). In other words, the sensor elements get saturated if the value of y_i (for any i) exceeds a 'saturation threshold' L, and the stored value is equal to L. So the forward model for the compressive measurements is represented as $\boldsymbol{y} = \mathcal{S}(\boldsymbol{\Phi\Psi\theta} + \boldsymbol{\eta})$ where $\mathcal{S}(z) = z$ if $z \leq L$ and $\mathcal{S}(z) = L$ if $z \geq L$. (Here of course, $\mathcal{S}$ is acting separately on each element from the entire measurement vector, so $\mathcal{S}(\boldsymbol{v}) = [\mathcal{S}(v_1), \mathcal{S}(v_2), ..., \mathcal{S}(v_n)]$ for n-dimensional vector $\boldsymbol{v}$.) Your job is to implement a method that will estimate $\boldsymbol{x}$ from $\boldsymbol{y}, \boldsymbol{\Phi}$ if you know L and σ. You can use any existing solver for CS. If you need to formulate a more complex optimization algorithm, feel free to use the CVX library, which is a well-known and easy to use convex optimization library. Note that due to saturation, this has effectively become a non-linear sensing model.

   Suppose that f is the number of saturated measurements divided by m. Plot a graph of the relative error for your algorithm with respect to (a) number of measurements m keeping f fixed, and (b) the fraction of saturated measurements f, keeping m fixed. State any conclusions you may wish to draw from these plots. You should use 1D signals of 256 elements. Of course, your aim is to get as good an estimate of $\boldsymbol{x}$ as possible. Tips: How do you know your estimate is good? If f is not too large, you should expect to get nearly the same RMSE as you would have got if there was no saturation. [20 points]