Homework 9

Submission instructions.

- Submissions are due on Thursday 11/12 at 10.00pm ET
- Please upload scans of your solution in GradeScope (via Canvas)

Instructions

- Please solve all non-MATL t resorting to a computer.

- Please show all necessary steps to get the final answer. However, there is no need to be overly elaborate. Crisp and complete answers.

- For all MATLAB problems, include all code written to generate sol
- Please post all questions on the discussion board on the Piazza cours

- If you feel some inform ions and proceed. Someti ble assumptions will be ac

1. *(Sub-differentials)* Let $\mathbf{x} \in \mathbb{R}^N$ and let $f(x) = \|$

   (Part a) Is $f(x)$ differentiable? Justify.

   (Part b) Derive the sub-differential of $f(x)$.

   (Part c) Derive the solution to the optimization problem

   $$\min_{\mathbf{x}} \beta\|\mathbf{x}\|_2 + \frac{1}{2}\|\mathbf{y} - \mathbf{x}\|_2^2.$$

   Hint: If you wait till Tuesday, Aswin will solve it in class.

2. *(Orthogonality in OMP)* Consider the OMP algorithm that we derived in class, and also given below as pseudo-code

   function $x = \text{OMP}(y, A, K, \text{tol})$
   $\Omega \leftarrow \phi$
   $r \leftarrow y$
   $k \leftarrow 0$
   while $(k < K)$ AND $(\|r\|/\|y\| \geq \text{tol})$
   $k \leftarrow k + 1$
   $j \leftarrow \arg\max_i |\langle r, a_j \rangle|$
   $\Omega \leftarrow \Omega \cup \{j\}$
   $x_\Omega \leftarrow \arg\min_\alpha \|y - A_\Omega \alpha\|^2, \qquad x_{\Omega^c} \leftarrow 0$
   $r \leftarrow y - Ax$

Show that, at the end of each iteration, the residue r is orthogonal to the columns of A in the index set Ω.

Implications: A column once selected wont be selected again since the residue will be orthogonal to it.

3. This problem uses OMP to solve the inpainting problem.

(Part a) Implement OMP s
implementation you are p ator


```
function x = OMP(y, A, K, tol)
%%
%% your code goes here
%%
end
```

You can use the pseudocode in problem 2 as a guide.

Deliverable: Y

(Part b) Test your C


```
clear all
close all

N = 1024;
M = 256;
K = 50;

A = randn(M,N)/sqrt(M);
x0 = zeros(N, 1);
Supp = randperm(N, K);
x0(Supp) = randn(K, 1);

y = A*x0;

xhat = OMP(y, A, K, 0);

stem(x0,  'kx');
hold on
stem(xhat, 'ro');
```

Deliverable: The plots. You can expect the black and red plot to match very closely.

(Part c) In `hw9.mat` you have the classic inpainting problem again. As is the case in inpainting problem, the unknown image `x` is related to the measurement `y` as follows: `y = x.*m;` You can find `y` and `m` in the mat file.

We are now going to use OMP to solve this. Let x_p be a 16×16 patch of the image `x`. Let m_p and y_p be patches at the same location for `y` and `m`, respectively. We will vectorize all three of them to create 256-dimensional vectors. Now the input-output relationship can be written as

You also have a matrix or s_p defined as

$$x_p = Ds_p$$

is K-sparse, where K is typically a small number, say smaller than 16.

With this, you can get the following equation

$$y_p = \text{diag}(m_p)Ds_p, \quad \|s$$

By iterating over patc construct the image `x`.

Deliverables. (a) Code, (b) Recovered image.

Remark. We have obviously left out a lot of details, includ K to use, and whether or not to overlap the patches in the image. Sto something you can consider. Basically, fill in these details as needed.