Homework 10

Submission instructions.

- Submissions are due on Thursday 11/19 at 10.00pm ET
- Please upload scans of your s

Instructions

- Please solve all non-MATLAB problems using **only** paper and pen, without 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 questio
- If you feel some inform ions and proceed. Someti ble assumptions will be accepted.

1. *(Deblurring and boundary conditions)*

   Consider the code snippet below.

```
clear all
close all
sharpimg = imread('cameraman.tif'); %%load sharp image
sharpimg = double(sharpimg)/255;

%blur kernel -- lets create a random one
k0 = rand(11,11);
k0(6, :) = 1;
k0(:, 6) = 1;

k0 = k0/sum(sum(k0));

%Linear convolution
blurimage = conv2(sharpimg, k0, 'valid');
%%this is typically what you get when you take a photo
%%the 'valid' boundary condition suggests the following
```

```
%% ------ world is not periodic or zero padded, but instead some light from the region
%%

clear sharpimg %%avoid temptation to access the answer

%%%Your code goes here
%%you are welcome to use the variables blurimage, and k0
```

(Part a) Use Fourier doma
FFT of blurimage and k0, di
sharp image. The size of the FFT is something that is left to you. Also left to you is a strategy on how to handle nulls in the the Fourier transform of k0.

Deliverable: Code, and deblurred sharp image. You are also expected to explain why your deblurred image looks as follows.

(Part b) Now lets attempt to deblur again but this time using algebrai
the following code to reconstruct the sharp image.

```
vec  = @(x) x(:);
fA = @(x) vec(conv2                                         alid'));
fAadj = @(x) vec(con                                        nd:-1:1), 'full'));

rhsterm = fAadj(blurimage);
fCGS = @(x) fAadj(fA(x));
xhat = cgs( fCGS, rhsterm, 1e-6, 1000);

xhat= reshape(xhat, size(blurimage)+size(k0)-1);
imshow(xhat);
```

Deliverable: Just the reconstructed image. no code.

2. *(Iterative Soft Thresholding)*

   Implement iterative soft thresholding in MATLAB under sparsity prior in some basis.

   Your function for iterative soft thresholding that solves the problem

$$\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{y} - A(\mathbf{x})\|_2^2 + \lambda\|\Psi(\mathbf{x})\|_1,$$

   might look like this

```
function xstar = IST(y, lambda, Psi, PsiAdj, A, AAdj, MaxIter, Tol,
                         xinit)
```

Deliverable 1. Matlab function for IST. Please label the code for readability.

Deliverable 2. Use the code below to test your code. You should include the completed code and the figure it generates.

```
clear all; close all

N = 1024; %Signal dimensio
M = 256; %%number of measure
K = 64; %sparsity level

x0 = zeros(N, 1);
Supp0 = randperm(N, K); %support of sparse signal
x0(Supp0) = randn(K, 1); %%sparse signal

A = randn(M, N)/sqrt(M); %%measurement matrix

y = A*x0; %%measurements

%%%%[Your code goe
%%setup function h
%%call IST and get output in the variable xstar

%%Visualize results
stem(x0); hold on
stem(xstar, 'rx');
```

3. *(Deblurring with Iterative Soft Thresholding)*

   We will test the code from the problem above on the deblurring problem.

   In `hw10.mat`, you are given a blurred image in the variable `imblur` and a blur kernel `k0`. Deblur with IST.

   Deliverable 1. For the deblurring problem , provide MATLAB code for implementing the operator A and its adjoint A^*.

   Deliverable 2. Deblur the image using your IST code. Do sweep through values of λ for best results. In your submission you are expected to show the restored image. You will be evaluated on the quality of the reconstructed image.