Problem 3:

3. (30%) Based on the lecture demo on compressive imaging, use MATLAB to simulate the computations of a single pixel camera, which uses DMD to generate the random sensing matrix ( ∅𝑖,𝑗∈{0,1}).

Ans:

Here we used the code provided by Prof Liang and Romberg’s L1 optimization. We also use a coherence function that calculates the mutual incoherence of the matrix. While the RIP (Restricted Isometry Property) and the Null Space Property (NSP) provide guarantees for the recovery of a k-sparse signal, they are not easy to compute. The mutual coherence of a matrix, with a lower number meaning that the matrices, sensing, and representation matrices are more incoherent and gives a good recovery of k sparse vectors.

%---------------------------------------------------------------------

% file name : hmwk\_4\_prob\_3\_dmd\_simulation.m

% Student: Ray Duran

% Date: 10/16/21

% Class : EECS 590 Professor Liang, Fall Semester

% University of North Dakota

% Descr:

% Model a Digital Micro-Mirror Device(DMD) with a Bernouli Random Matrix

%

%

% Code Borrowed from :

% compressive\_sensing\_demo

% Use "L1 magic" by Justin Romberg.

% http://www.acm.caltech.edu/l1magic

% Download the L1 magic toolbox and add to the matlab dir

%---------------------------------------------------------------------

A = imread('cameraman.tif');

M = 50; %Downsample the image to 50x50

N = 50;

A = imresize(A,[M N]);

x = double(A(:));

n = length(x);

%\_\_\_MEASUREMENT MATRIX\_\_\_

m = 1000; % Number of samples

%Phi = randn(m,n); % Generate Gaussian distribute random numbers

p = 0.2;

Phi = binornd(1,p,m,n);

%\_\_\_COMPRESSION\_\_\_

y = Phi\*x;

%\_\_\_THETA\_\_\_

% NOTE: Avoid calculating Psi (nxn) directly to avoid memory issues.

Theta = zeros(m,n);

for ii = 1:n

ek = zeros(1,n);

ek(ii) = 1;

psi = idct(ek)';

Theta(:,ii) = Phi\*psi;

end

% Calculate coherence

u = mutual\_coherence(Theta);

lower\_bound\_on\_mutual\_coherence = ((N-M)/(M\*(N-1)))^0.5;

%\_\_\_l2 NORM SOLUTION\_\_\_ s2 = Theta\y; %s2 = pinv(Theta)\*y

s2 = pinv(Theta)\*y;

%\_\_\_BP SOLUTION\_\_\_

s1 = l1eq\_pd(s2,Theta,Theta',y,5e-3,20); % L1-magic toolbox

%\_\_\_IMAGE RECONSTRUCTIONS\_\_\_

x1 = zeros(n,1);

for ii = 1:n

ek = zeros(1,n);

ek(ii) = 1;

psi = idct(ek)';

x1 = x1+psi\*s1(ii);

end

%\_\_\_\_\_ Plots

coherence\_str = (['Coherence =' num2str(u)]);

figure('name','Compressive sensing image reconstructions')

subplot(1,2,1), imagesc(reshape(x,M,N)), xlabel('original'), axis image

subplot(1,2,2), imagesc(reshape(x1,M,N)), xlabel('basis pursuit'), axis image

str = sprintf( 'Digital Micro-Mirror Device Single Pixel Camera Simulation \n mututal coherence w/ Bernouli random matrix = %4.2f ',u)

sgtitle(str)

%subtitle(coherence\_str)

colormap gray

