CSCI5254

Homework Sets #5-6

[Boyd&Vandenberghe, Convex Optimization]:

Exercises 6.1, 6.9, 7.3, 7.4 [(a) is required; (b) is optional with extra credit], 7.8, 7.9

[Additional Exercises,
http://spot.colorado.edu/~lich1539/cvx/extra_exercises.pdf **]**

Exercises 3.9, 4.1, 5.2, 5.6, 5.13, 5.15, 6.4, 6.6, 12.4, 15.3

Matlab file for data for exercise 5.6: [the original image can be found at
http://spot.colorado.edu/~lich1539/cvx/tv_img_interp.png]

```
% tv_img_interp.m
% Total variation image interpolation.
% Defines m, n, Uorig, Known.

% Load original image.
Uorig = double(imread('tv_img_interp.png'));

[m, n] = size(Uorig);

% Create 50% mask of known pixels.
rand('state', 1029);
Known = rand(m,n) > 0.5;

%%%%% Put your solution code here

% Calculate and define Ul2 and Utv.

% Placeholder:
Ul2 = ones(m, n);
Utv = ones(m, n);

%%%%%

% Graph everything.
figure(1); cla;
colormap gray;

subplot(221);
```

```
imagesc(Uorig)
title('Original image');
axis image;

subplot(222);
imagesc(Known.*Uorig + 256-150*Known);
title('Obscured image');
axis image;

subplot(223);
imagesc(Ul2);
title('l_2 reconstructed image');
axis image;

subplot(224);
imagesc(Utv);
title('Total variation reconstructed image');
axis image;
```

Matlab file for data for exercise 5.13:

```
% data for censored fitting problem.
randn('state',0);

n = 20;  % dimension of x's
M = 25;  % number of non-censored data points
K = 100; % total number of points
c_true = randn(n,1);
X = randn(n,K);
y = X'*c_true + 0.1*(sqrt(n))*randn(K,1);

% Reorder measurements, then censor
[y, sort_ind] = sort(y);
X = X(:,sort_ind);
D = (y(M)+y(M+1))/2;
y = y(1:M);
```

Matlab file for date for exercise 5.15:

```
%% data for learning a quadratic metric
% provides X, Y, d, X_test, Y_test, d_test

rand('seed',0);
randn('seed',0);
```

```
n = 5;    % dimension
N = 100;  % number of distance samples
N_test = 10;

X = randn(n,N);
Y = randn(n,N);
X_test = randn(n,N_test);
Y_test = randn(n,N_test);

P =randn(n,n);
P = P*P'+eye(n);
sqrtP = sqrtm(P);

d = norms(sqrtP*(X-Y)); % exact distances
d = pos(d+randn(1,N)); % add noise and make nonnegative
d_test = norms(sqrtP*(X_test-Y_test));
d_test = pos(d_test+randn(1,N_test));

clear P sqrtP;
```

Matlab file for data for exercise 6.4:

http://spot.colorado.edu/~lich1539/cvx/team_data.m

File for data for exercise 6.6:

```
clear all; close all;

% create problem data
randn('state',0);
N = 100;
% create an increasing input signal
xtrue = zeros(N,1);
xtrue(1:40) = 0.1;
xtrue(50) = 2;
xtrue(70:80) = 0.15;
xtrue(80) = 1;
xtrue = cumsum(xtrue);

% pass the increasing input through a moving-average
filter
% and add Gaussian noise
```

```
h = [1 -0.85 0.7 -0.3]; k = length(h);
yhat = conv(h,xtrue);
y = yhat(1:end-3) + randn(N,1);
```

File for data for exercise 15.3:
http://spot.colorado.edu/~lich1539/cvx/net_util_data.m