Non Linear Programming: Homework 8

vishvAs vAsuki

April 6, 2010

1 Total variation image interpolation

1.1 Code

```
% tv_img_interp.m
% Total variation image interpolation.
% EE364a
% Defines m, n, Uorig, Known.

% Load original image.
Uorig = double(imread('flowgray.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
cvx_begin
variable Ul2(m, n);
variable T(m-1, n-1);
minimize sum(sum((Ul2(2:end, 2:end) - Ul2(1:end-1,2:end))
    .^2 + (Ul2(2:end, 2:end) - Ul2(2:end,1:end-1)).^2))
subject to
Ul2 .* Known == Uorig .* Known;
cvx_end

cvx_begin
variable Utv(m, n);
variable T(m-1, n-1);
minimize sum(sum(abs(Utv(2:end, 2:end) - Utv(1:end-1,2:
    end)) + abs(Utv(2:end, 2:end) - Utv(2:end,1:end-1))))
subject to
```

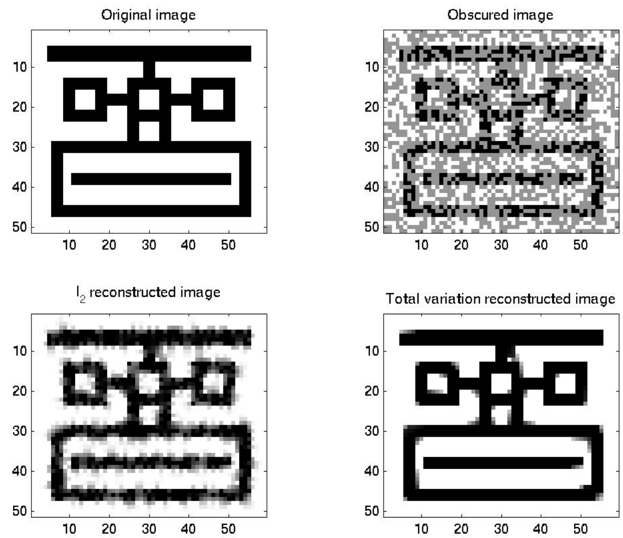
```
Utv .* Known == Uorig .* Known;
cvx_end

% Calculate and define Ul2 and Utv.

%%%%

% Graph everything.
figureHandle = 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;

saveas(figureHandle, ['/u/vvasuki/vishvas/work/
    optimization/hw/hw8/code/imageInterpolation.jpg'], '
    jpg');
% close all;
```

1.2 Results

2 Sparse linear separation

2.1 Code

```
function sparseLinearSeparationExperiment
threshold = 10^-2;
hw8_sparse;
```




```
[thicknessMax, sparsityMin] = getSlabThicknessSparsity(X,
     Y, 0, threshold);
fprintf(1, 'Max thickness: %d min sparsity: %d \n',
    thicknessMax, sparsityMin);
%   return

thicknesses = [];
sparsities = [];
a_10ftr = [];
a = [];
for l = 0:5:100
     %   for lPow = -50:50:100
     %   l = 10^lPow;
     l
     [thicknesses(end+1), sparsities(end+1), a] =
         getSlabThicknessSparsity(X, Y, l, threshold);
     if(sparsities(end) > 39/50 && numel(a_10ftr) == 0)
          a_10ftr = a;
     end
     a'
%        keyboard
end

plotFigure(thicknesses, sparsities);
```

```
importantFeatureIndices = find(abs(a_10ftr) >= max(abs(
    a_10ftr))*threshold);
importantFeatureIndices
[thickness_10ftr, sparsities_10ftr, a] =
    getSlabThicknessSparsity(X(importantFeatureIndices, :)
    , Y(importantFeatureIndices, :), 0, threshold);

display('All done, ready for inspection');
keyboard
end

function plotFigure(thicknesses, sparsities)
figureHandle = figure();
figureHandle = plot(sparsities, thicknesses);
xlabel('sparsity');
ylabel('slab thickness');
% close all;
saveas(figureHandle, ['/u/vvasuki/vishvas/work/
    optimization/hw/hw8/code/sparseSeparation.jpg'], 'jpg'
    );
end

function [thickness, sparsity, a] =
    getSlabThicknessSparsity(X, Y, l, threshold)
[n, N] = size(X);
M = size(Y, 2);
cvx_begin
variable a(n);
variable b;
minimize norm(a, 2) + l*norm(a, 1)
subject to
X'*a + b*ones(N,1) >= ones(N,1);
Y'*a + b*ones(M,1) <= -ones(M,1);
cvx_end
thickness = 2/norm(a,2);
sparsity = sum(abs(a) < max(abs(a))*threshold)/n;
end
```

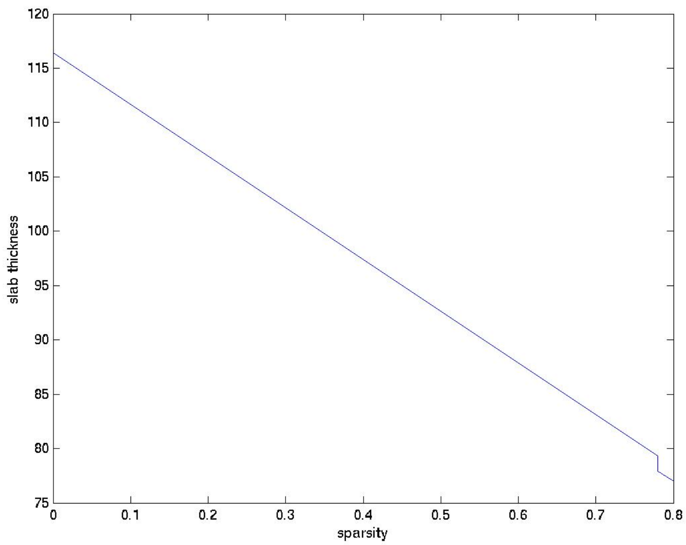
2.2 Max slab thickness

Max thickness: 1.164244e+02 min sparsity: 0

Maximizing slab width ($\frac{2}{\|a\|}$) is equivalent to minimizing $\|a\|$.

2.3 Tradeoff curve

2.4 Solution after identifying important features




```
importantFeatureIndices =
     1.0000e+000
     7.0000e+000
     8.0000e+000
    18.0000e+000
    19.0000e+000
    21.0000e+000
    23.0000e+000
    26.0000e+000
    27.0000e+000
    46.0000e+000


thickness_10ftr =

    78.4697e+000
```