EL 7133 (DSP II)
Spring 2016
Homework Assignment – Week 07(HW 06)

Name: **Amitesh Kumar Sah**
NYU ID: **N19714360**

QUESTION 1

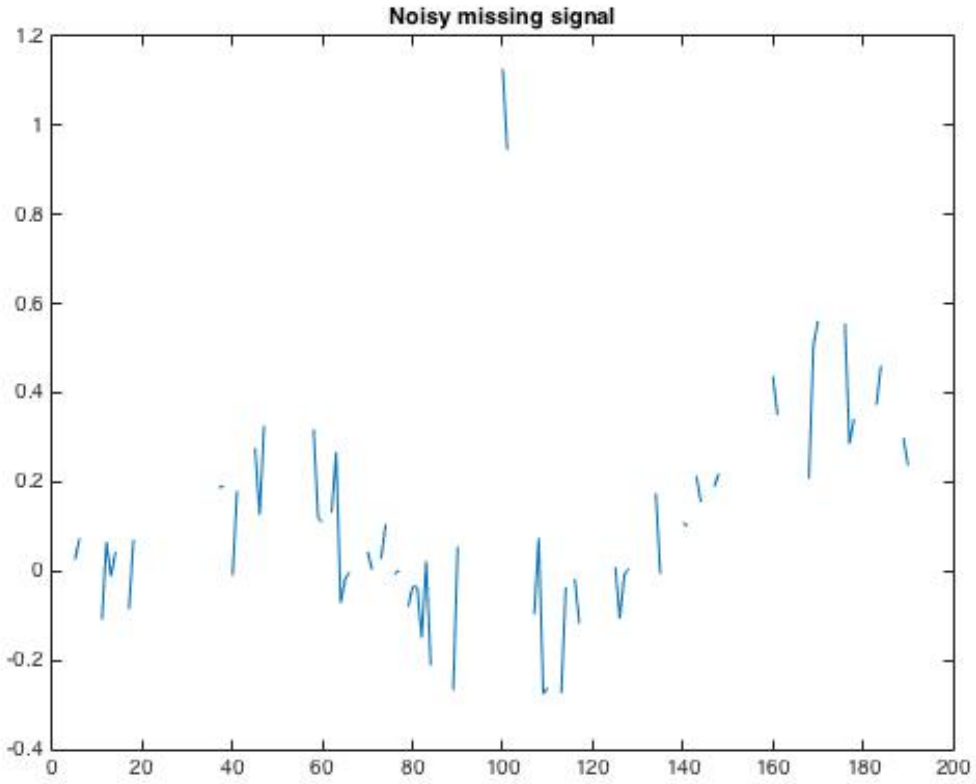
Suppose the available data is noisy and that some samples are missing. Formulate a suitable least squares optimization problem to simultaneously smooth the data and recover the missing samples. Illustrate the effectiveness by a numerical demonstration (e.g. using Matlab).

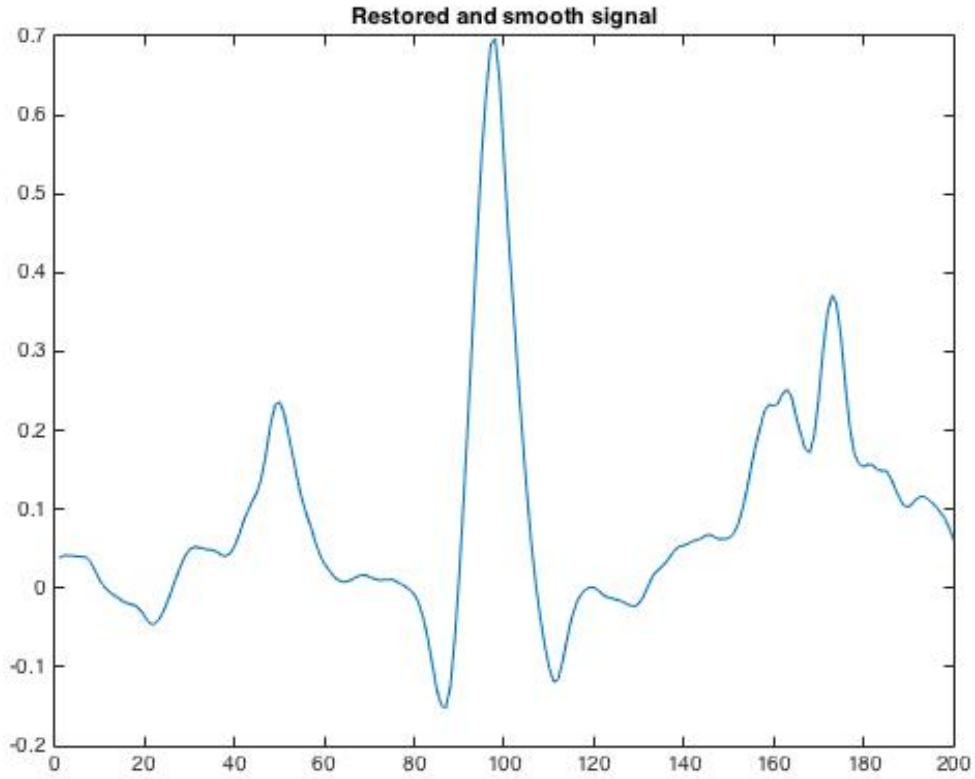
Matlab code

```
clc;clear all;close all
x=textread('ECG_noisy_incomplete.txt');
figure,plot(x),title('Noisy missing signal');
N=size(x,1);
% Calculating matrix S
miss=isnan(x);
k=sum(miss(:)==0);
s=zeros(k,N);
j=1;
for i=1:N
   if miss(i)==0;
      s(j,i)=1;
      j=j+1;
   end
end
x(isnan(x)) = 0 ;
y=s*x;
I=eye(N);
% ?complement? of S
sc=zeros(N-k,N);
j=1;
for i=1:N
   if miss(i)==1;
      sc(j,i)=1;
      j=j+1;
   end
end
% second order difference matrix
D1=[1 -2 1];
D=zeros(N,N);
for i=1:N-2
   D(i,i:i+2)=D1(1,:);
end
lambda1=15;
lambda2=10;

C=(sc*sc'+lambda2.*sc*D'*D*sc');
A=(lambda1*D'*D+I);
B=sc';
```

```
E=(sc*s'*y+lambda2.*sc*D'*D*s'*y);
x=-A\(B*(C\E));
figure,plot(x),title('Restored and smooth signal');
```

Output





Observation:
Here , the objective function was taken jointly for missing signal and the smoothing signal. Here is the objective function

$$J(x) = min_x \left\| S^T y + Sc^T v - x \right\|_2^2 + \lambda_1 \left\| Dx \right\|_2^2 + \lambda_2 \left\| D(S^T y + Sc^T v)_2^2 \right\|$$

X is obtained from here. By changing the value of lambda 1 and lambda2 , smoothness changes. It also leads to change in the magnitude. Above graph is obtained for lambda 1=15 and lambda2=10. We get a approximately close signal with still some noise on it.

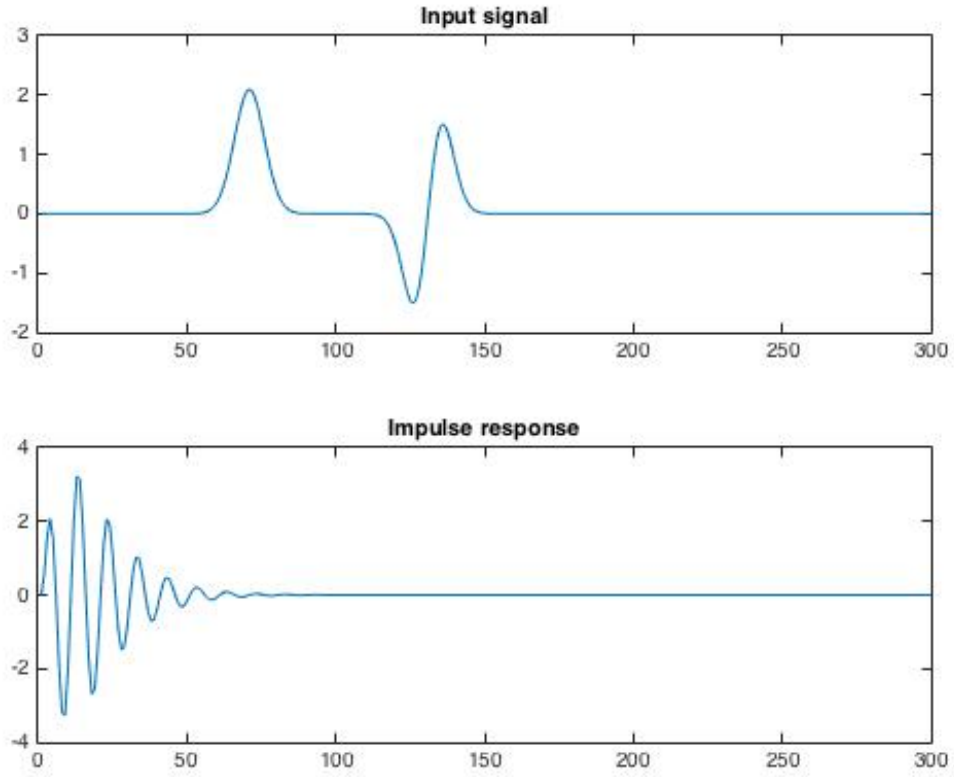
Question 2
Perform deconvolution by solving the problem using the Landweber Algorithm

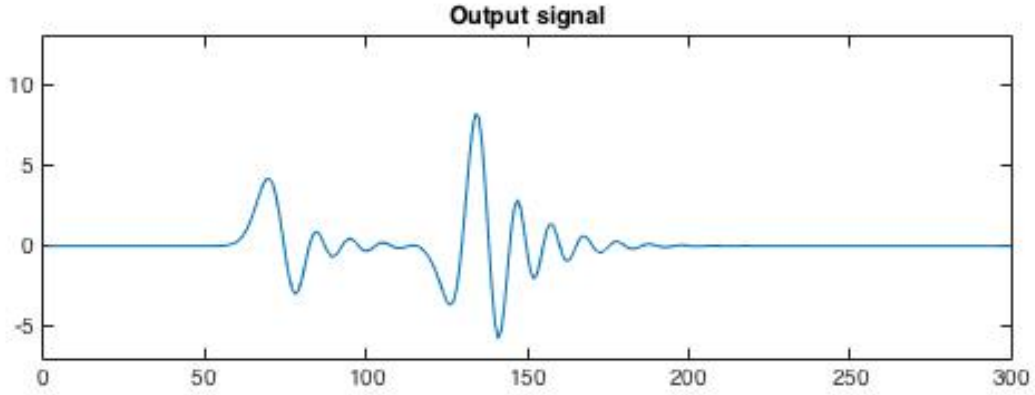
$$\min_{\mathbf{x}} \|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 + \lambda \|\mathbf{D}\mathbf{x}\|_2^2$$

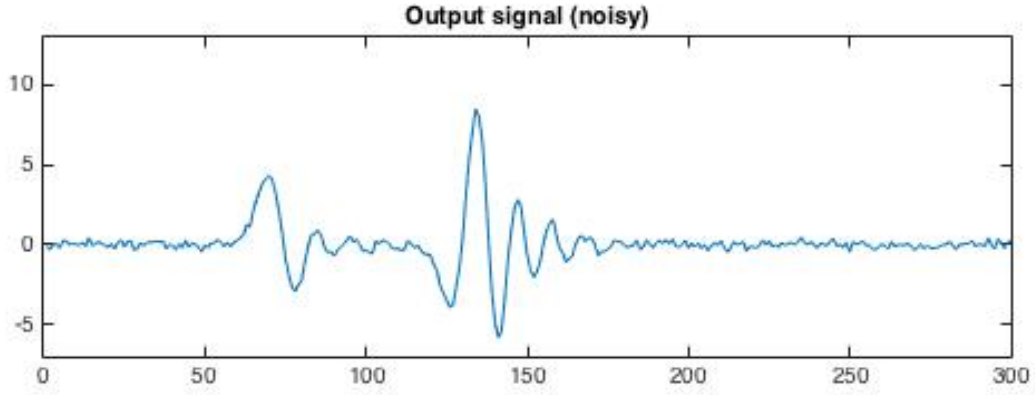
```
clc;clear all;close all
N = 300;
n = (0:N-1)';                    % n : discrete-time index

w = 5;
n1 = 70;
n2 = 130;
x = 2.1 * exp(-0.5*((n-n1)/w).^2) - 0.5*exp(-0.5*((n-n2)/w).^2).*(n2 - n);   % x : input signal

h = n .* (0.9 .^n) .* sin(0.2*pi*n);      % h : impulse response

figure(1)
clf
subplot(2, 1, 1)
plot(x)
title('Input signal');
YL1 = [-2 3];
ylim(YL1);

subplot(2, 1, 2)
plot(h)
title('Impulse response');
randn('state', 0);                % Set state for reproducibility

y = conv(h, x);
y = y(1:N);                    % y : output signal (noise-free)
yn = y + 0.2 * randn(N, 1);          % yn : output signal (noisy)
figure(2)
clf
subplot(2, 1, 1)
plot(y);
YL2 = [-7 13];
ylim(YL2);
title('Output signal');

subplot(2, 1, 2)
```

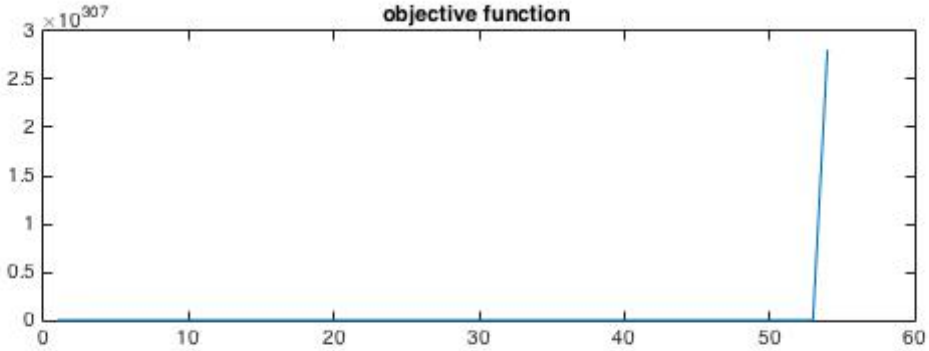
```
plot(yn);
title('Output signal (noisy)');
ylim(YL2);
H = convmtx(h, N);
H = H(1:N, :);                    % H : convolution matrix
e = ones(N, 1);
D = spdiags([e -2*e e], 0:2, N-2, N);
N = 100;
lambda = 0.1;
alpha = 1;
beta=max(eig(D'*D));
Nit = 700;

J = zeros(1, Nit); % Objective function
x = 0*H'*y; % Initialize x
T = lambda/(2*(alpha+lambda*beta));
for k = 1:Nit
Hx = H*x;
Dx= D*x;
J(k) = sum(abs(Hx(:)-y(:)).^2) + lambda*sum(abs(Dx(:)).^2);
x = wthresh(x*alpha + (H'*(y - Hx))/(alpha+lambda*beta),'s',T);
end
figure,subplot(2,1,1),plot(J),title('objective function');
subplot(2,1,2),plot(x),title('Restored signal');
```

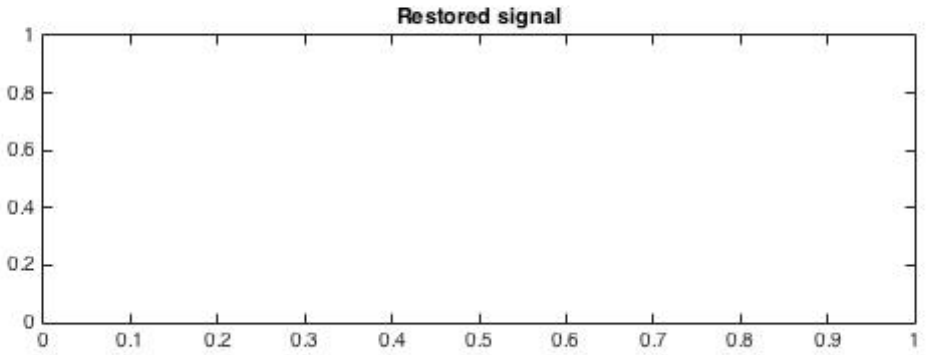
Output:











Observation

I am getting very high value for objective function and also the signal x is showing all NaN values. I am not able to figure out my error.

Question 3
Describe the transpose of 2D convolution and write a Matlab function to implement it

Matlab Code

```
clc;clear all;close all
x=imread('Lenna.png');
x=rgb2gray(x);
h = [1 1 1;1 -4 1;1 1 1]/4;
y = conv2(h,x);
y1= conv2(h',x);
% H = @(x) conv2(h,x);
Ht = @(y) convt2(h,x);
% Y=H(x);
Yt=Ht(x);
Ydiff=sum(sum(abs(y1-Yt)))
figure,
subplot(2,2,1),imshow(x);title('Input image');
subplot(2,2,2),imshow(y,[]);title('Output conv2(h,x) without function handle');
subplot(2,2,3),imshow(y1,[]);title('y1 Output conv2(h^t,x) without function handle');
subplot(2,2,4),imshow(Yt,[]);title('Yt Output Ht(x) @convt2(h,x)');


function f = convt2(h,g);
% f = convt(h,g);
% Transpose convolution: f = H? g
[Nhh,Nhw] = size(h);
[Ngh,Ngw] = size(g);
f = conv2(h(Nhh:-1:1,Nhw:-1:1), g);
% f = f(Nhh:Ngh,Nhw:Ngw);
end
```

Output

Ydiff = 0

Input image

Output conv2(h,x) without function handle

y1 Output conv2(h^t,x) without function handle

Yt Output Ht(x) @convt2(h,x)

Observation

Here, I used a smoothing filter. It is observed that the matrix obtained H^t from function handle using convt2 () function is same as the matrix obtained from convolution. We see that the difference between the two is zero.

Question 4

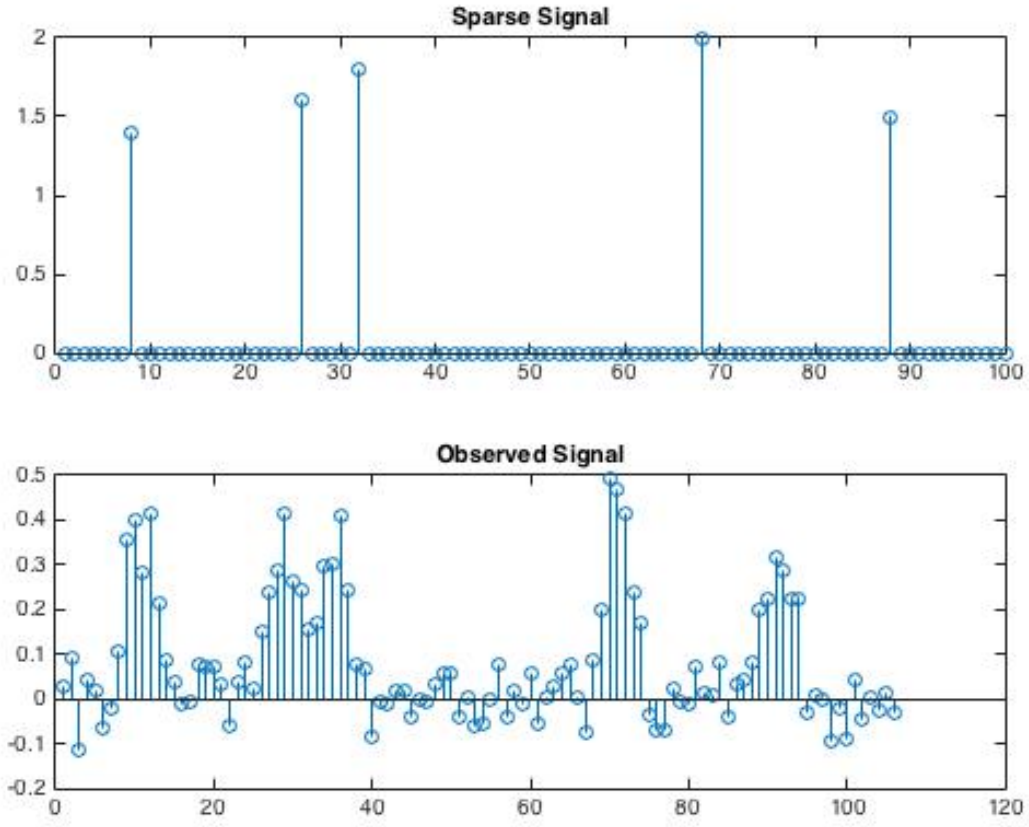
Implement and demonstrate sparse deconvolution like the example in Section 6 in the notes Sparse Signal Restoration.

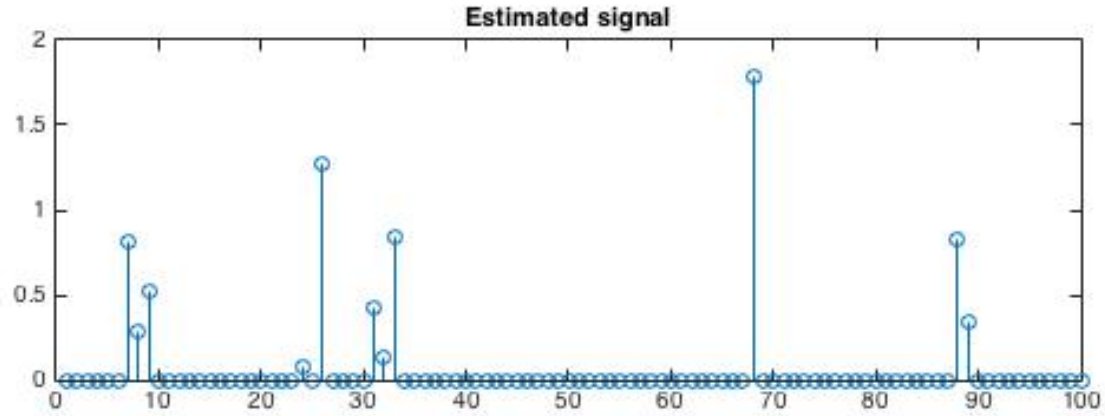
Matlab Code

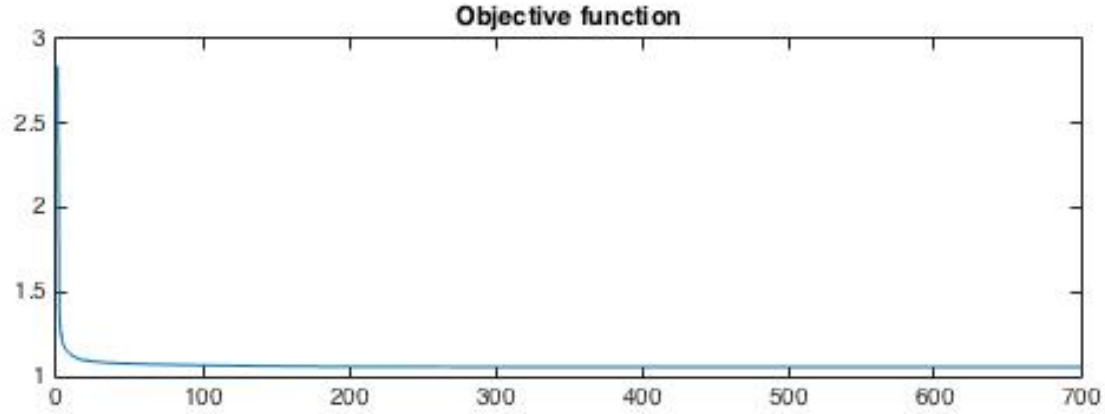
```
clc;clear all;close all
X=zeros(100,1);
X(8)=1.4;X(26)=1.6;X(32)=1.8;X(68)=2;X(88)=1.5;
h = [1 2 3 4 3 2 1]/16;
y=conv(X,h);
% adding white Gaussian noise with standard deviation 0.05
sigma = 0.05; N=length(y);
noise = sigma * randn(N,1);
y=y+noise;
N = 100;
H = convmtx(h',N);
lambda = 0.1;
```

```
alpha = 1;
Nit = 700;
[x, J] = ista(y, H, lambda, alpha, Nit);
figure,
subplot(2,1,1),stem(X),title('Sparse Signal');
subplot(2,1,2),stem(y),title('Observed Signal');
figure,
subplot(2,1,1),stem(x),title('Estimated signal');
subplot(2,1,2),plot(J),title('Objective function');
```

Output







Observation

The estimated output is approximately equal to the sparse signal. We can also see how the objective function decays with number of iteration. However ,if you increase the number of iteration, we get more closer result.

Question 5

Find a couple of the papers listed in the reference list of the notes Sparse Signal Restoration. Comment on the problem the papers solve and how they solve it.

Paper 1

T. Goldstein and S. Osher. The split Bregman method for L1-regularized problems. SIAM J. Imag. Sci., 2(2):323–343, 2009

Problem they want to solve:

The class of l1-regularized optimization problems has received much attention recently because of the introduction of "compressed sensing," which allows images and signals to be reconstructed from small amounts of data. Despite this recent attention, many l1-regularized problems still remain difficult to solve, or require techniques that are very problem-specific.

The general form for such problems is min u $|\Phi(u)| + H(u)$

Solution

In this paper, we show that Bregman iteration can be used to solve a wide variety of constrained optimization problems. Using this technique, theye propose a "Split Bregman" method, which can solve a very broad class of l1-regularized problems. We apply this technique to the ROF functional for image denoising, and to a compressed sensing problem that arises in Magnetic Resonance Imaging.

Besides its speed, our algorithm has several advantages: Becuase the Split Bregman algorithm makes extensive use of Gauss-Seidel and Fourier transform methods, it is easily parallelizable. Also, it has a relatively small memory footprint compared to second order methods that require explicit representations of the Hessian matrix. Both of these characteristics make Split Bregman a practical algorithm for large scale problems. Finally, the method is easy to code.

Paper 2

E. T. Hale, W. Yin, and Y. Zhang. Fixed-point continuation for `1-minimization: Methodology and convergence. SIAM J. on Optimization, 19(3):1107–1130, 2008.

Problem they want to solve

This paper presents an algorithm for solving the ℓ_1-regularized minimization problem:

$$\text{minimize}_x ||x||_1 + \mu f(x),$$

where $f(x)$ is a differentiable convex function.

Solution

Our approach is based on two powerful algorithmic ideas: operator-splitting and continuation. Operator-splitting results in a fixed-point algorithm for any given scalar μ; continuation refers to approximately following the path traced by the optimal value of x as μ increases. In this paper, we study the structure of optimal solution sets, prove finite convergence for important quantities, and establish q-linear convergence rates for the fixed-point algorithm applied to problems with f(x) convex, but not necessarily strictly convex. The continuation framework, motivated by our convergence results, is demonstrated to facilitate the construction of practical algorithms.