2.161 Signal Processing: Continuous and Discrete

Fall 2008

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
DEPARTMENT OF MECHANICAL ENGINEERING

2.161 Signal Processing - Continuous and Discrete
Fall Term 2008

Solution of Problem Set 9

Assigned: November 20, 2008 **Due:** December 4, 2008

Problem 1:

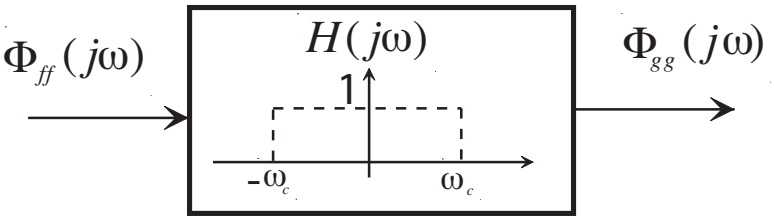
$$\mu_x = \int_{-\infty}^{+\infty} xp(x)dx = \int_{0}^{+\infty} x(\lambda e^{-\lambda x})dx = e^{-\lambda x}(-x - \frac{1}{\lambda})\,\big|_0^{+\infty} = \frac{1}{\lambda}$$

$$\sigma_x^2 = \mu_{x^2} - \mu_x^2 = \int_{-\infty}^{+\infty} x^2 p(x)dx - \mu_x^2 = \int_{0}^{+\infty} x^2(\lambda e^{-\lambda x})dx - \frac{1}{\lambda^2}$$

$$\sigma_x^2 = e^{-\lambda x}(-x^2 + 2\frac{1}{\lambda}(-x - \frac{1}{\lambda}))\,\big|_0^{+\infty} - \frac{1}{\lambda^2} = \frac{2}{\lambda^2} - \frac{1}{\lambda^2} = \frac{1}{\lambda^2}$$

$$\sigma_x = \frac{1}{\lambda}$$

Problem 2:

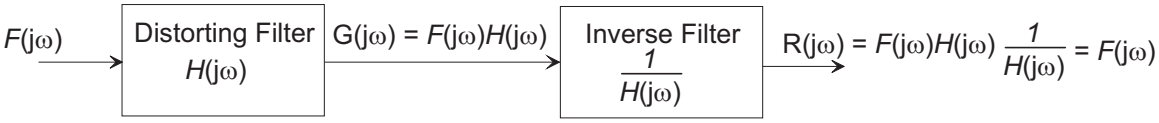


Input f is a white noise with flat spectrum and hence we assume that $\Phi_{ff}(j\omega) = 1$. The output can be computed from:

$$\Phi_{gg}(j\omega) = |H(j\omega)|^2 \Phi_{ff}(j\omega) = |H(j\omega)|^2 .1 = |H(j\omega)|^2$$

$$\Phi_{gg}(\tau) = \mathcal{F}^{-1}(\Phi_{gg}(j\omega)) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} |H(j\omega)|^2 e^{jw\tau}d\omega = \frac{1}{2\pi}\int_{-\omega_c}^{\omega_c} 1.e^{j\omega\tau}d\omega = \frac{\omega_c}{\pi}\text{sinc}(\omega_c\tau) = \frac{\sin(\omega_c\tau)}{\pi\tau}$$

Problem 3:

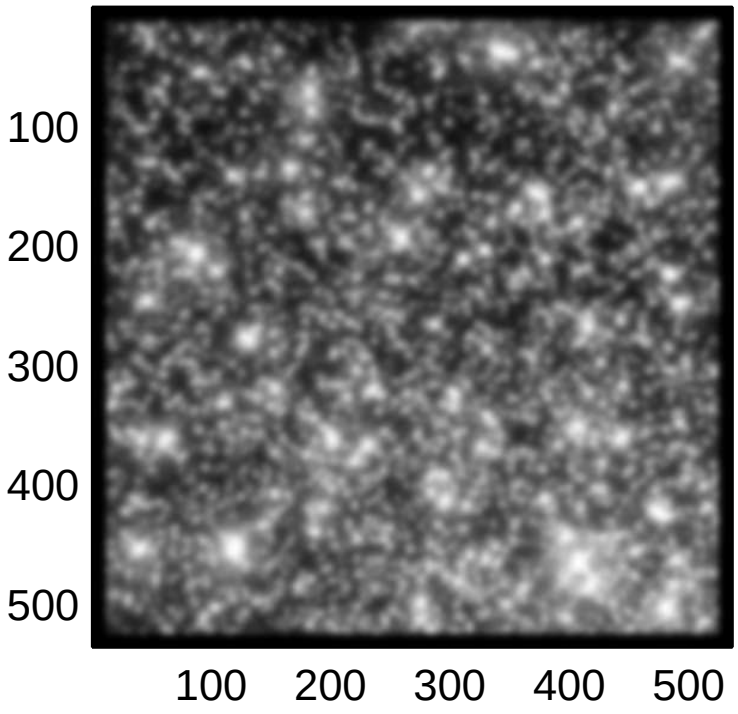


The inverse filter for this *high-precision* problem can be implemented as the *exact* inverse of the input filter. We will revisit this problem in the 2nd quiz where additive noise of quantization imposes some modification to the inverse filter. Here is the code that we used.

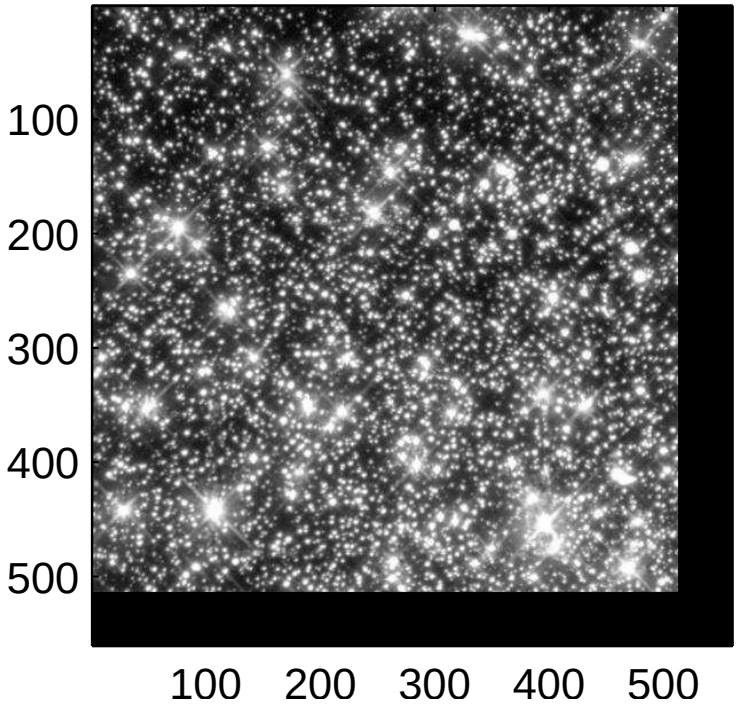
- Note that "h" kernel needs to be zero padded such that that image and h DFT have the same size. This ensures that their corresponding DFT matrix elements, correspond to the same (u, v) value and we can multiply them element-wise.
- Note the color scaling at the end. All the color channels are scaled with the same scale, such that relative RGB pattern is not altered.

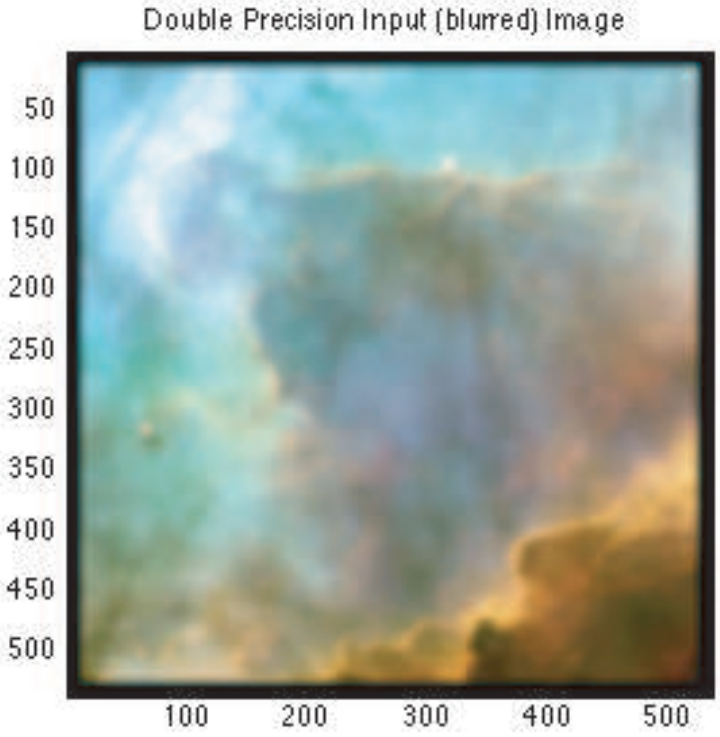
```
clear all,close all,clc
%%   Read in the image
load('blurredimage2.mat','blurred')
figure,subplot(2,1,1),image(blurred), axis image
title('Double Precision Input (blurred) Image')
%% Determine the size of the image supplied
pad = 24;
imsize =  size(blurred);
xsize = imsize(1) + pad;ysize = imsize(2) + pad;
% Take fft of image
fftblur = fft2(blurred,xsize,ysize);
%---------------------------------
%% Design the inverse filter
%  Use a gaussian kernel from the image processing toolbox
hsize=25;sigma = 3;
h = fspecial('gaussian', hsize, sigma);
ffth = fft2(h,xsize,ysize);
% Create the inverse filter - this is just the simple form:
fftinvh = 1./ffth;
% figure
% mesh(fftshift(abs(fftinvh)))
% title('Inverse Filter Magnitude')
%---------------------------------
%% Filter the image
fftdeblur = zeros(xsize,ysize,3);
for i=1:3
    fftdeblur(:,:,i) = fftinvh.*fftblur(:,:,i);
end
%% Inverse fft and Color Intensity Scaling
deblurred = abs(ifft2(fftdeblur));
%  Normalize the pixel intensity for the whole image
%  to set all values in the range 0 -- 1 (already all are positive)
maxpix = max(max(max(max(deblurred))),1);
deblurred = deblurred/maxpix;
% Show the image
subplot(2,1,2),image(deblurred), axis image
title('Double Precision Output (deblurred) Image')
```

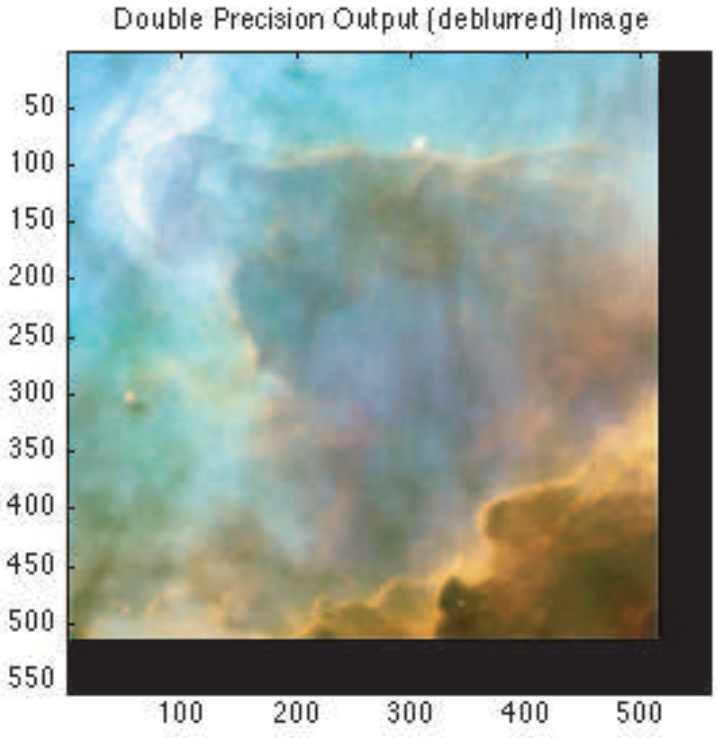
Double Precision Input (blurred) Image



Double Precision Output (deblurred) Image




Double Precision Input (blurred) Image
50
100
150
200
250
300
350
400
450
500
100
200
300
400
500


Double Precision Output (deblurred) Image
50
100
150
200
250
300
350
400
450
500
550
100
200
300
400
500

Problem 4: (This is from the take-home Quiz 2 from last year.)

(a) The autocorrelation of the contaminated signal is

$$\begin{aligned}\phi_{gg}(\tau) &= E\{g(t)g(t+\tau)\} \\ &= E\{(f(t)+\alpha f(t-\Delta))(f(t+\tau)+\alpha f(t+\tau-\Delta))\} \\ &= E\{(f(t)f(t+\tau)\} + E\left\{\alpha^2 f(t-\Delta)f(t+\tau-\Delta)\right\} \\ &\quad + E\{\alpha f(t)f(t+\tau-\Delta)\} + E\{\alpha f(t+\tau)f(t-\Delta)\}\end{aligned}$$

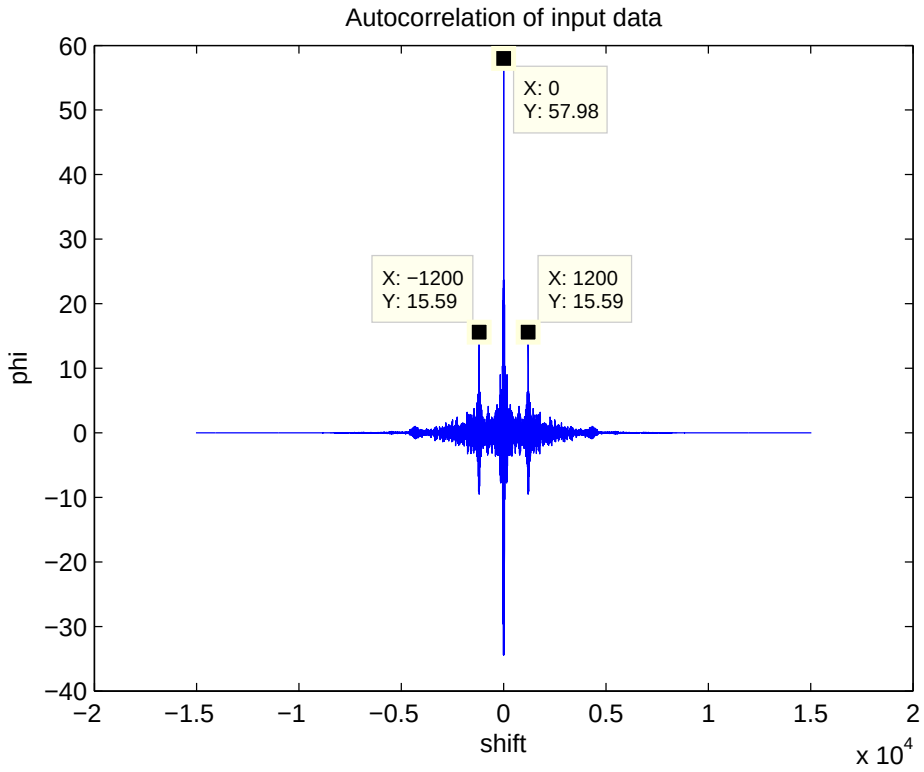
and recognizing that $\phi_{ff}(\tau)$ does not depend on the time origin,

$$\phi_{gg}(\tau) = (1+\alpha^2)\phi_{ff}(\tau) + \alpha\phi_{ff}(\tau-\Delta) + \alpha\phi_{ff}(\tau+\Delta)$$

(b) With the MATLAB code

```
[f,fs,Nbits] = wavread('PS9Prob4.wav');
L = length(f);
figure(1), plot(-(L-1):L-1, xcorr(f,f))
```

the following plot was annotated using the cursor tip.



We assume that $\phi_{ff}(0) >> \phi_{ff}(2\tau), \phi_{ff}(\tau)$ and then from (a) peaks on the graph correspond to:

$$(1+\alpha^2)\phi_{ff}(0) \approx 57.98 \qquad \alpha\phi_{ff}(0) \approx 15.59$$

or

$$\alpha^2 - 3.72\alpha + 1 = 0$$

so that $\alpha \approx 0.29$. Actually $\alpha = 0.3$ and we will use $\alpha = 0.3$ value. Similarly, from the autocorrelation function $m = 1200$.

(c) Recognize that Eq. (2) is a filter with $H(z) = 1 + \alpha z^{-\Delta}$. Then the inverse filter $H'(z) = 1/H(z)$ to remove the echo is

$$H'(z) = \frac{1}{1+\alpha z^{-\Delta}}$$

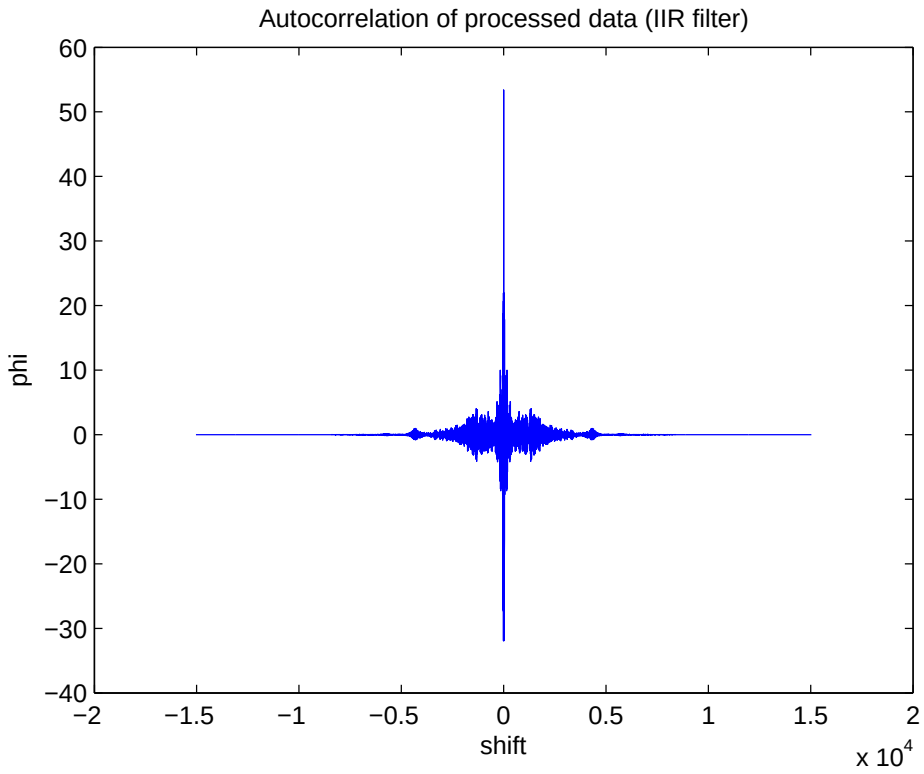
which is an IIR filter with a difference equation

$$f_n = -\alpha f_{n-m} + g_n.$$

(d) Below is the complete MATLAB code to implement the filter and filter the data:

```
[f,fs,Nbits] = wavread('PS9Prob4.wav');
L = length(f);
figure(1), plot(-(L-1):L-1, xcorr(f,f))
title('Autocorrelation of input data')
xlabel('shift')
ylabel('phi')
% Look at the autocorrelation function
wavplay(f,fs)
% Determine that the delay is 1200 and alpha=0.3
delay = 1200; alpha = 0.3;
% Recursive filter:
% Set up the denominator:
a1 = zeros(1,1201); a1(1) = 1; a1(1201) = alpha;
y = filter(1, a1, f);
figure(2), plot(-(L-1):L-1, xcorr(y,y))
title('Autocorrelation of processed data (IIR filter)')
xlabel('shift')
ylabel('phi')
%
wavplay(f1,fs)
```

which produced the following output autocorrelation plot



Note that the echo components have been eliminated.