DFT Applications – Final Exam

Tamoghna Chattopadhyay

Section 1: Sampling and Reconstruction with DFT filter.

Source Code:

```
% -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
-- -- -- -- -- -- -- -- -
% TITLE: Sampling and Reconstruction with DFT filter.
%
% Purpose: To  applying the Discrete Fourier Transform (DFT) to a
% variety of signal processing problems.
%
% Date created: 07/30/2016 Author: Tamoghna Chattopadhyay
% Date modified: rev1 - 08/02/2016
% -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
-- -- -- -- -- -- -- --

% Generate 8192 samples of x(t) at a sampling rate of 100K (10 u seconds).
n = [0:8191];
x = cos( 2*pi*(160/100)*n ) + 0.5 * sin( 2*pi*(180/100)*n );

% Compute the DFT and plot its magnitude for the sampled signal
X = fft(x);
figure(1);
subplot(211),plot( abs(X) );
title( 'Fourier Transform of Original' );
xlabel( 'Frequency (Hertz)');
ylabel( 'Magnitude (unknown) ');

% Insert 3 zeros between each sample
x1 = ins_zeros( x, 4 );

% The spectrum of the up sampled signal
X1 = fft(x1);
subplot(212),plot( abs(X1) );
title( 'Fourier Transform of Zero Inserted' );
xlabel( 'Frequency (Hertz)');
ylabel( 'Magnitude (unknown) ');

% Construct a normalized frequency array ( 0 to 1 matches 0 to fs/2 )
f = [0:length(X1)-1]/length(X1);

% Apply Highpass Filter to the transform
```

```
HP_FM  = X1 .* HighPass_dft( length( X1 ) )';

% Display the filtered transform
figure(2);
plot( f(1:end/2),  abs( HP_FM(1:end/2)));
title('Fourier Transform of the filtered version');
xlabel( 'Frequency (KHz)');
ylabel( 'Magnitude (unknown) ');

% Generate 8192 samples of x(t) at a sampling rate of 400K (2.5 u
seconds).
x2 = cos( 2*pi*(160/400)*n ) + 0.5 * sin( 2*pi*(180/400)*n );

% Reconstruct the Original Signal
Xr = 4*ifft(HP_FM);

% Compare the reconstructed signal and the original signal sampled at 400K
figure(3);
plot( x2, 'r:');
hold on;
plot( Xr, 'b-');
title('Comparison of the reconstructed signal and the original signal
sampled at 400K');
xlabel( 'Frequency (Hertz)');
ylabel( 'Magnitude (unknown) ');
```

Function for Inserting Zeroes:

```
function xe = ins_zeros( x, I )
%
% xe = ins_zeros( x, I );
%
% This function will create an extended version of the signal x
% by simply inserting (I-1) zeros between each sample.

% Create extended set.
xe = zeros( length(x)*I, 1 );
xe(1:I:end) = x;
return;
```

Function for HighPass DFT Filter:

```
function H = HighPass_dft( N );
%
% H = HighPass_dft( N );
%
% This program generates a High Pass filter for application to the DFT of
a signal.
% The output is an array that is point by point multiplied
% with the dft of the signal.
% The input parameter N is then length of the signal.

n1 = floor( 0.35 * N );
n2 = floor( 0.3875 * N );
n3 = floor( 0.5 * N + 1 );

H = zeros( 1, N );
H( n1:n2 ) = [0:1/(n2-n1):1]; % Up to one
H( n2+1:n3 ) = H( n2+1:n3 ) + 1; % Set At one
H( (N/2+2):N ) = H(N/2:-1:2); % Conjugate Symmetry
return;
```

Plots:

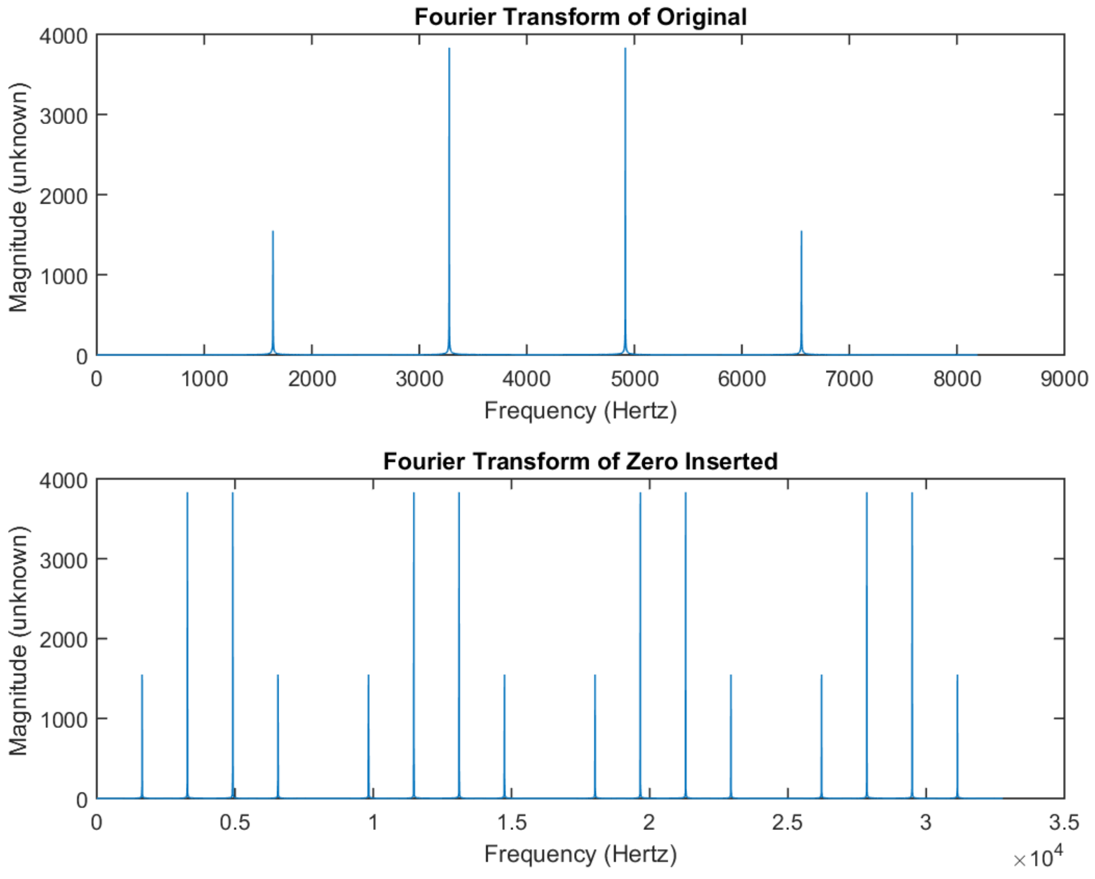

Figure 1

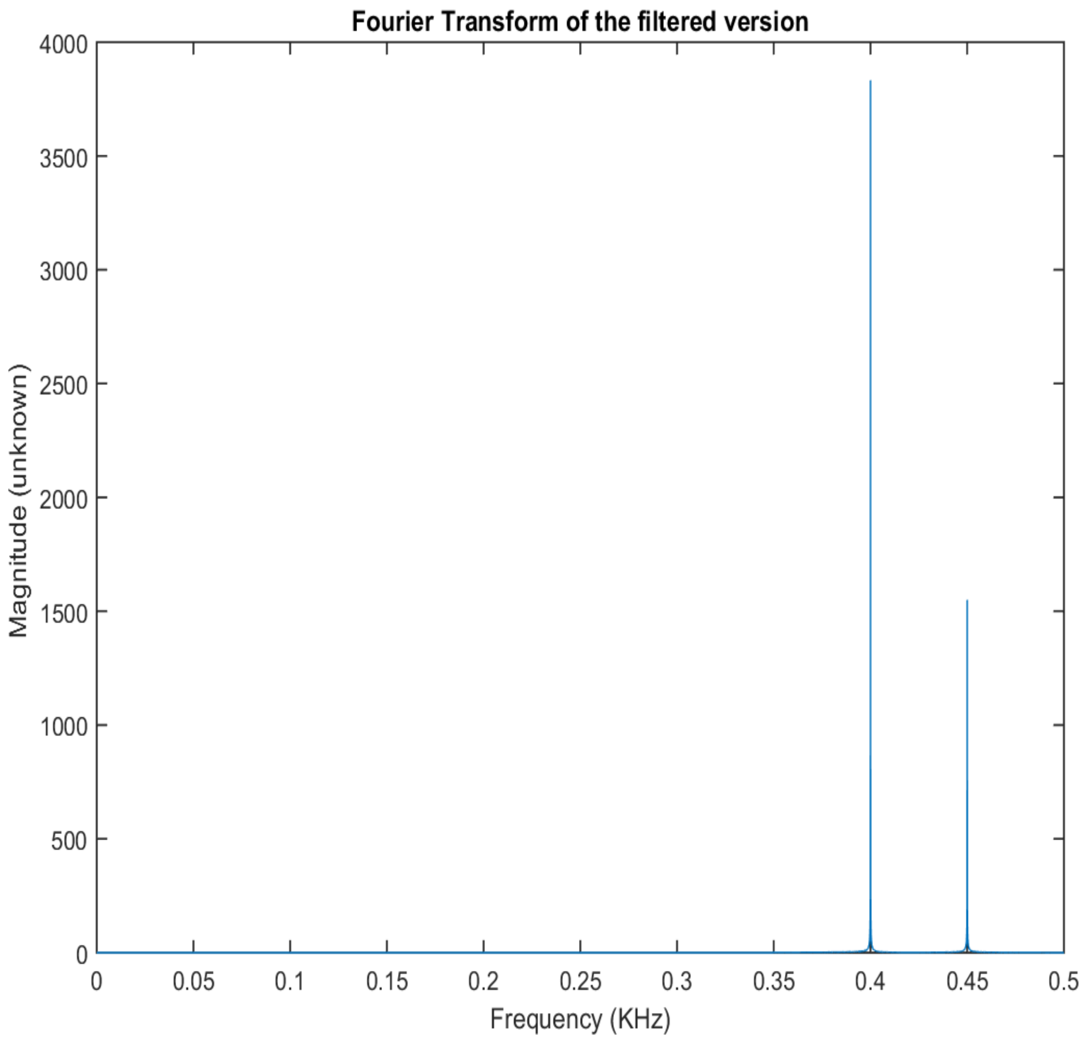
Fourier Transform of the filtered version
Magnitude (unknown)
4000
3500
3000
2500
2000
1500
1000
500
0
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
Frequency (KHz)

Figure 2

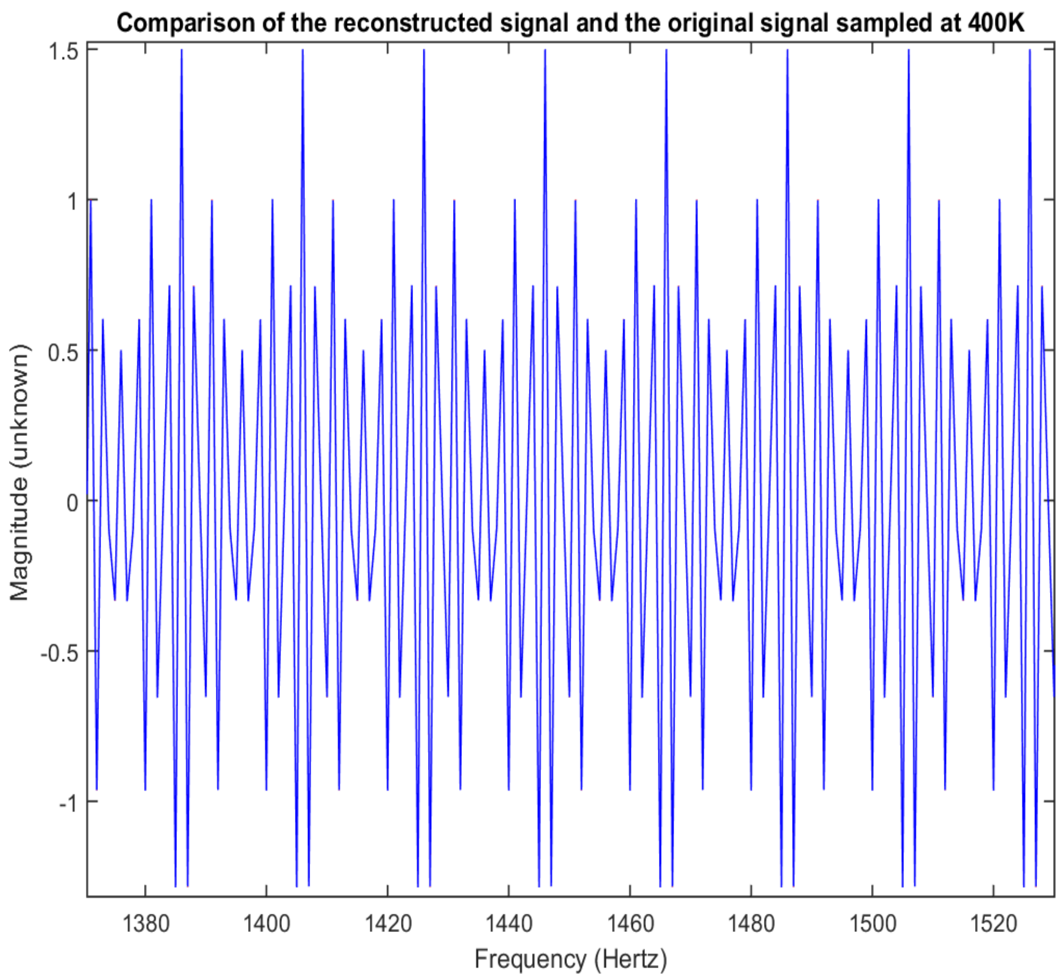
Comparison of the reconstructed signal and the original signal sampled at 400K
Magnitude (unknown)
Frequency (Hertz)
1.5
1
0.5
0
-0.5
-1
1380
1400
1420
1440
1460
1480
1500
1520

Figure 3

Section 2: Overlap Add.

Source Code:

```
% -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
-- -- -- -- -- -- -- -- -
% TITLE: Overlap Add
%
% Purpose: To employ Overlap Add in order to process a long signal. A FIR
% bandpass filter with cosine times a hamming window is utilised.
%
%
% Date created: 08/01/2016 Author: Tamoghna Chattopadhyay
% Date modified: rev1 - 08/02/2016
% -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
-- -- -- -- -- -- -- --

% Load the given file
load('DFT_Final.mat');

% Plot the original signal
figure(1);
plot( signal ); % Plot the spectrum
title('Original signal in given MatLab file');
xlabel (' Frequency in Hertz ' );
ylabel (' Magnitude (unknown) ');

% Define the lengths of filter and transform, i.e. 1024 point data
H_Length = length(h);
F_Length = 1024

% Find the fourier transform
H = fft( h, 1024 );

% Filter the data using Overlap-Add
n = 0;
m = 1;
figure(2);
for k = 1:4
    SIG = fft( signal( m:m+F_Length-H_Length-1 ),F_Length );
    y = ifft( SIG.*H );
    if( k==1 )
        z = y;
        n = F_Length-H_Length;
    else
        z = [z(1:n)  z(n+1:n+H_Length)+y(1:H_Length)
y(H_Length+1:F_Length)];
        n = n + F_Length-H_Length;
        end;
    m = m + F_Length-H_Length;
end;

% Plot the result of filtering
```

```
figure(2);
plot( real( z ) );
title('Overlap-Add Method of Filter Data');
xlabel( 'Frequency (Hertz)');
ylabel( 'Magnitude (unknown) ');
```

Plots:

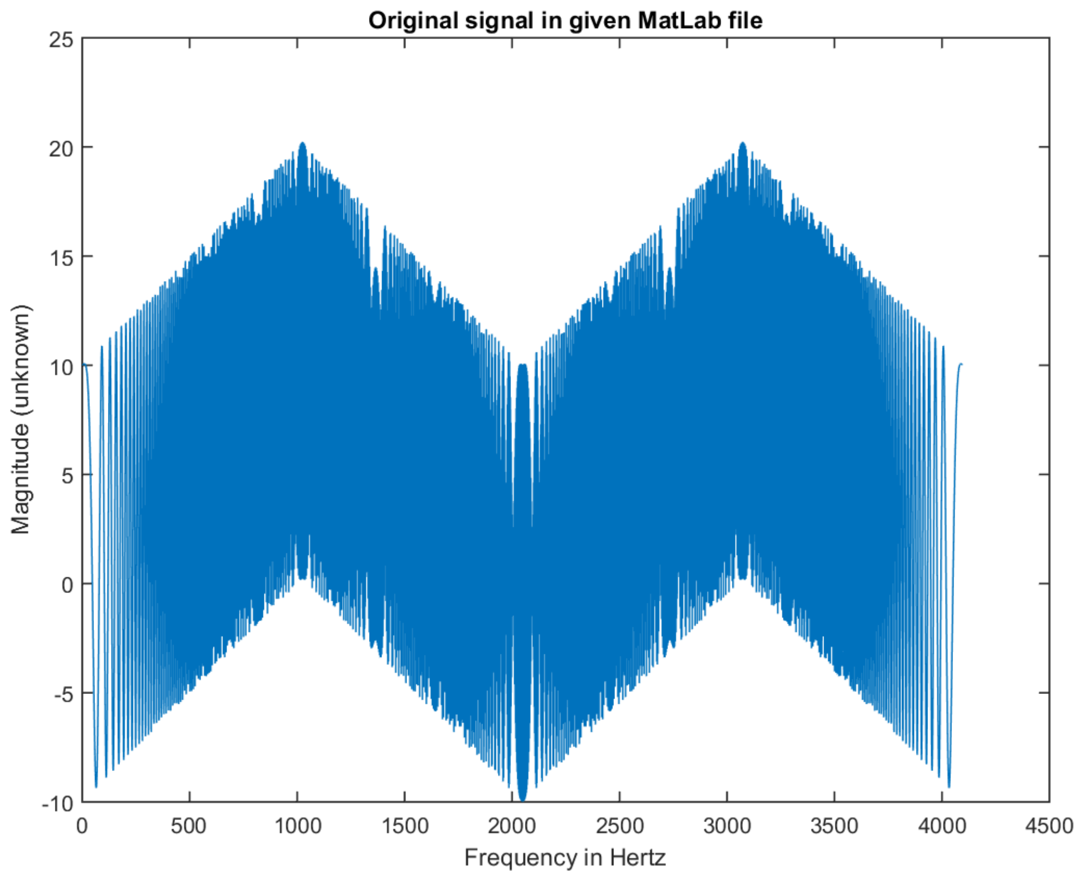

Figure 1

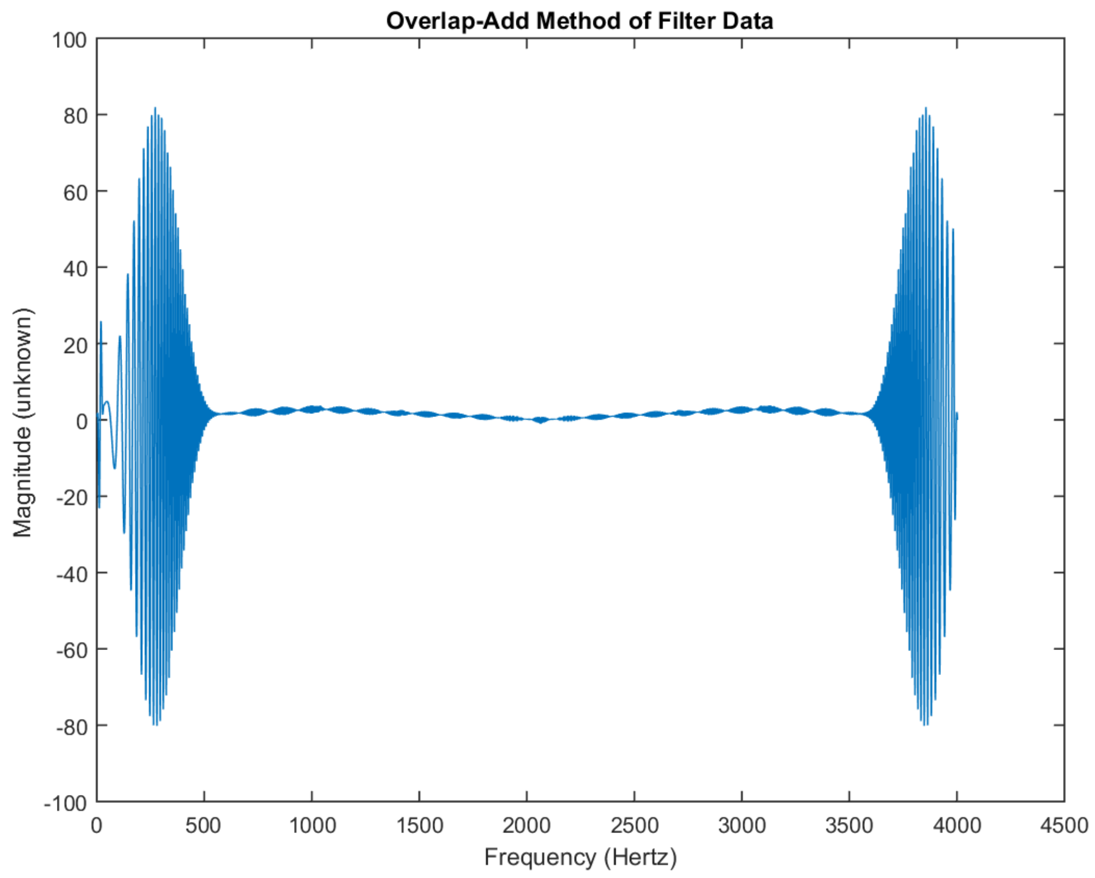
Overlap-Add Method of Filter Data
Magnitude (unknown)
Frequency (Hertz)
100
80
60
40
20
0
-20
-40
-60
-80
-100
0
500
1000
1500
2000
2500
3000
3500
4000
4500

Figure 2

Section 3: Spectral Analysis.

Source Code:

```
% -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
-- -- -- -- -- -- -- -- -
% TITLE: Spectral Analysis
%
% Purpose: To do spectral analysis on the signal test given in the MatLab
% file provided.
%
%
% Date created: 08/01/2016 Author: Tamoghna Chattopadhyay
% Date modified: rev1 - 08/02/2016
% -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
-- -- -- -- -- -- -- --

% Load the given file
load('DFT_Final.mat');

% Find the frequency of the large main component by iteration
N = 256;
TEST = fft(test,N);
[TMax,NMax] = max(abs(TEST(1:end/2)));
freq = 100e3*(NMax-1)/N;

N = 2*N;
TEST = fft(test,N);
[TMax,NMax] = max(abs(TEST(1:end/2)));
freq = 100e3*(NMax-1)/N;

N = 2*N;
TEST = fft(test,N);
[TMax,NMax] = max(abs(TEST(1:end/2)));
freq = 100e3*(NMax-1)/N;

N = 2*N;
TEST = fft(test,N);
[TMax,NMax] = max(abs(TEST(1:end/2)));
freq = 100e3*(NMax-1)/N;

N = 2*N;
TEST = fft(test,N);
[TMax,NMax] = max(abs(TEST(1:end/2)));
freq = 100e3*(NMax-1)/N

% Plot the fourier transform of the signal
figure(1);
plot( abs(TEST));
title('Fourier Transform of original Signal');
xlabel( 'Frequency (Hertz)');
ylabel( 'Magnitude (unknown) ');
```

```
% Subtract out the maximum component
n=0:length(test)-1;
Sub = test - real((1/length(test))*(TMax*exp(i*2*pi*(NMax-1)/N*n))) + ...
conj(TMax)*exp(-i*2*pi*(NMax-1)/N*n);

% Compute its fourier transform
SUB = fft(Sub,N);

% Plot the output of the fourier transform of the subtraction
figure(2);
plot( abs(SUB));
title('Fourier Transform of Signal with subtracted component ');
xlabel( 'Frequency (Hertz)');
ylabel( 'Magnitude (unknown) ');

% Construct a normalized frequency array ( 0 to 1 matches 0 to fs/2 )
f = [0:length(TEST)-1]/length(TEST);

% Apply Highpass Filter to the transform
BP_FM  = TEST .* BandPass_dft( length( TEST ) );

% Display the filtered transform
figure(3);
plot( f(1:end/2),  abs( BP_FM(1:end/2)));
title('Fourier Transform of the filtered version');
xlabel( 'Frequency (KHz)');
ylabel( 'Magnitude (unknown) ');
```

Function for BandPass DFT Filter:

```
function H = BandPass_dft( N );
%
% H = BandPass_dft( N );
%
% This program generates a Band Pass filter for application to the DFT of
a signal.
% The output is an array that is point by point multiplied
% with the dft of the signal.

% The input parameter N is then length of the signal.

n1 = floor( 0.1 * N );
n2 = floor( 0.15 * N );
n3 = floor( 0.25 * N );
n4 = floor( 0.3 * N );
H = zeros( 1, N );
H( n1:n2 ) = [0:1/(n2-n1):1]; % Up to one
H( n2+1:n3-1 ) = H( n2+1:n3-1 ) + 1; % Set At one
H( n3:n4 ) = [1:-1/(n4-n3):0]; % Down to zero
H( (N/2+2):N ) = H(N/2:-1:2); % Conjugate Symmetry
return;
```

Plots:

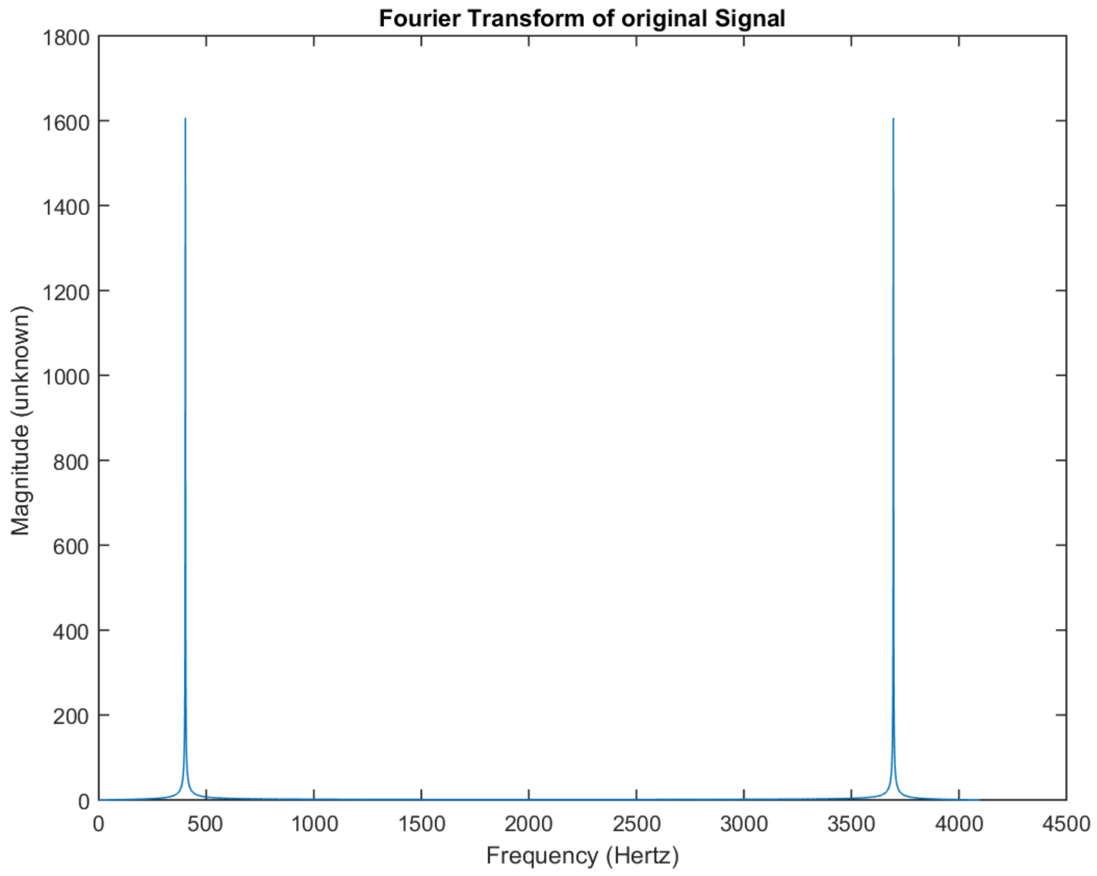

Figure 1

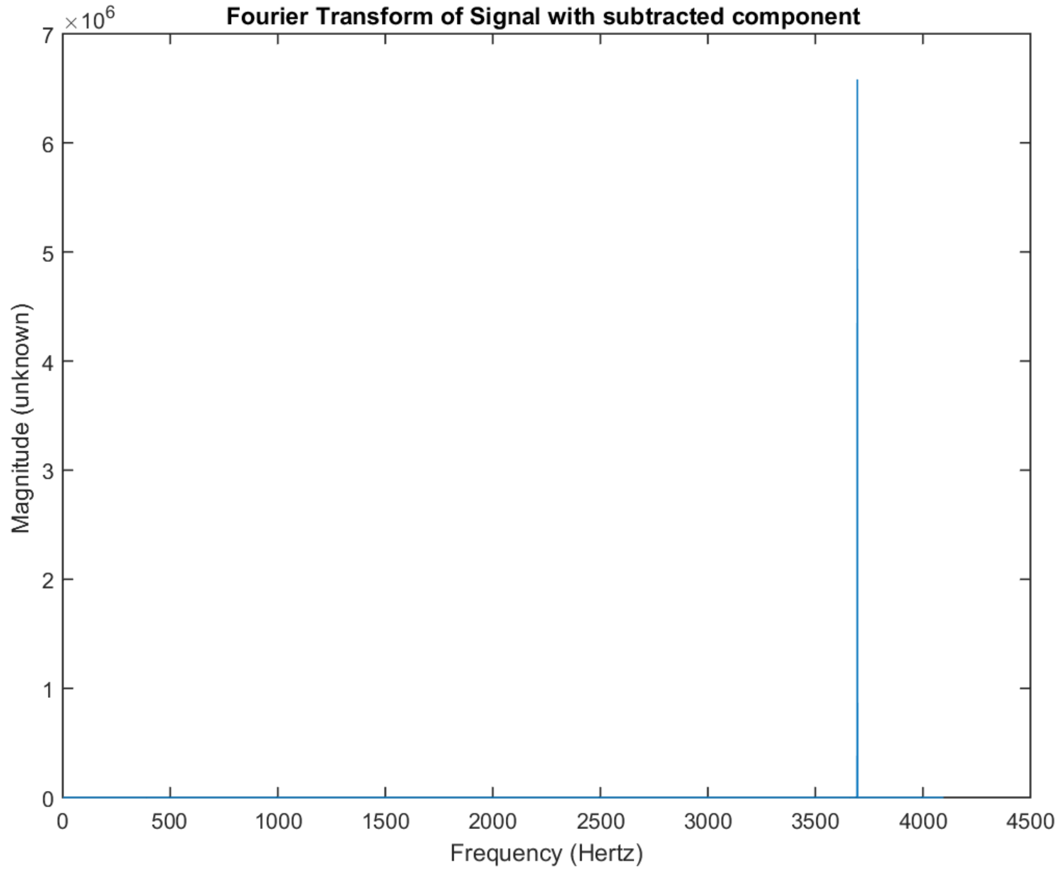
Fourier Transform of Signal with subtracted component
$\times 10^6$
Magnitude (unknown)
Frequency (Hertz)
0
1
2
3
4
5
6
7
0
500
1000
1500
2000
2500
3000
3500
4000
4500

Figure 2

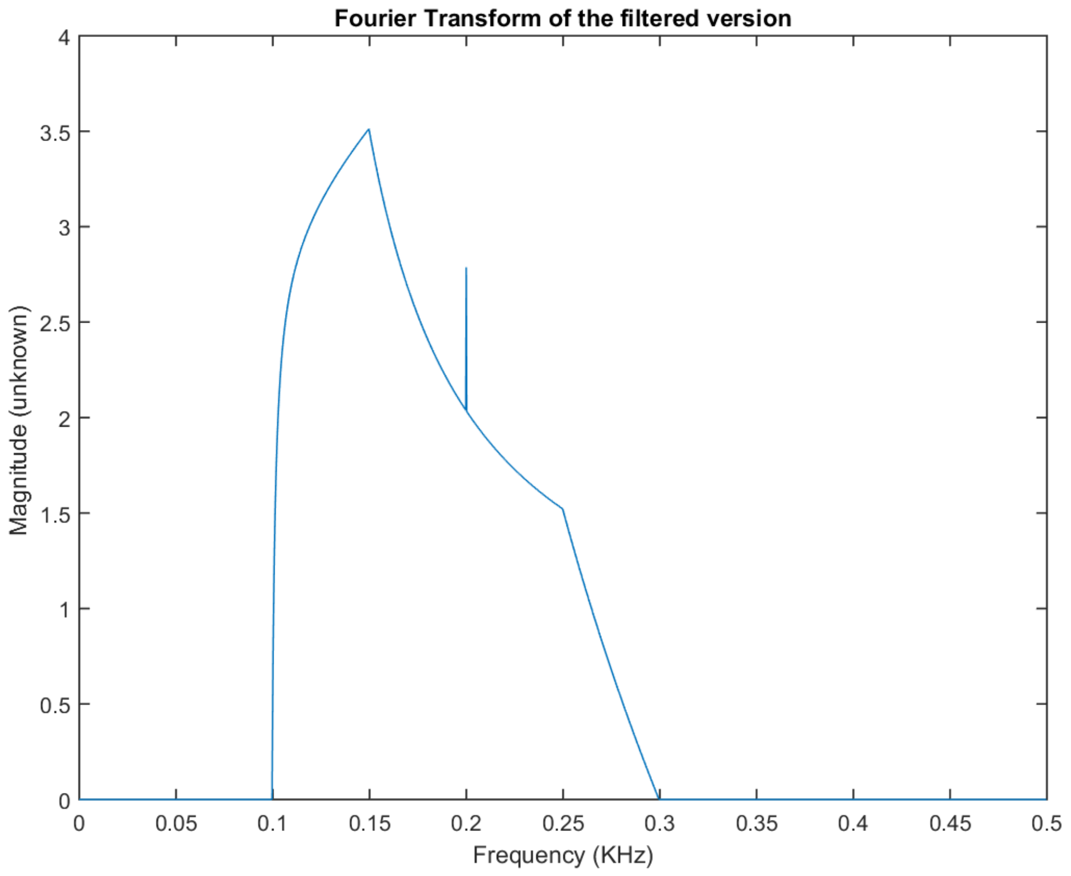
Fourier Transform of the filtered version
Magnitude (unknown)
Frequency (KHz)
0
0.5
1
1.5
2
2.5
3
3.5
4
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5

Figure 3

4) ADC's and DSP Processors. (15 points)

Your company has won a contract from the U.S. Customs and Border Patrol to supply a handheld instrument for detecting dangerous chemicals. The instrument is to detect and classify many different organic compounds – in particular, Sarin, VX, and Mustard Gas.

Your team is tasked with designing and developing the handheld instrument, which will be the size of a portable drill. The instrument has a gas/air sample handling system (pump, valves, and tubing), a Surface Acoustic Wave (SAW) detector as the transducer, a 32-bit microcontroller for handling the sampling, passing data, running the display, and sounding an alarm if needed.

The instrument will be the size and shape of a portable drill. It will have a battery pack in the bottom of the handle (see Figure 3). The display will be a small LCD screen on the back.

You are the DSP expert and you will design the DSP classifier for the instrument. In essence the microcontroller controls everything (so don't worry about it) and then passes the sample values to the DSP classifier; the classifier then correlates the sample signature with a library of up to 80,000 different compound signatures. The sample signatures are histograms of 1024 different values. The DSP classifier takes each signature and cross-correlates it with each of the 40,000 signatures in the library. The cross-correlation result closest to 1 identifies as the compound.

Your task in this problem is to select a type of DSP chip and the number of chips needed by calculating how many operations will be needed to classify a sample within 1 second. You are to load each DSP chip to 50% of full operational capacity or less. A cross-correlation is very similar to a FFT in operation, so use that as your guide to calculate the number of operations needed.

You have three different DSP chips within the same family from which to select a type of chip. You also have to determine the number of DSP chips to run in parallel. All three types of chips have the same instruction set and run four (4) instructions in parallel within a single clock cycle. But each type of chip runs at a different clock speed from the others, with a different power consumption and physical size. Table 1 lists the chip types and their characteristics.

Figure 3. A portable drill, which the instrument will look like.

Table 1. DSP Specifications.

	Chip A	Chip B	Chip C
Clock speed (MHz)	80	150	300
Max. power consumption (W)	1.3	2	4
Average power at 50% load (W)	0.7	1.2	2.6
Size (in mm)	10 x 10 x 2	12 x 12 x 2	20 x 20 x 2
Cost/chip ($)	$2.00	$3.50	$9.00

Answer these questions:

How many instructions can each DSP chip execute per second?
(Remember that each chip executes four instructions in parallel)

Chip A = 320×10^6 instructions/sec

Chip B = 600×10^6 instructions/sec

Chip C = 1200×10^6 instructions/sec

What is n (the number of bins in the histogram or signature)? n = 1024

What is the general scaling factor for calculating the size of a FFT?

(Hint: this is a formula) $N \log_2 N$

Assuming that a cross-correlation is identical to a FFT, use the formula for scaling factor to derive the number of instructions in each cross-correlation. Assume that each DSP chip uses four instructions for each step in the scaling factor formula. Show your work: $(1024 \log_2 1024) \times 4 = 40960$

Number of instructions/cross-correlation = 40960

Number of clock cycles/cross-correlation = 10240

10240

How many cross-correlations does each DSP chip execute per second at 100% loading?

Chip A = $\frac{320 \times 10^6}{40960} = 7812.5$ cross-correlations/sec

Chip B = $\frac{600 \times 10^6}{40960} = 14648.43$ cross-correlations/sec

Chip C = $\frac{1200 \times 10^6}{40960} = 29296.875$ cross-correlations/sec

How many cross-correlations does the instrument need to complete per second?

40000 instructions × 1 crosscorrelation / ins / sec.

Number of cross-correlations/sec for the instrument = 4000 crosscorrelations / sec

How many chips from each type, running at or below 50% loading, are needed to complete all cross-correlations within 1 second?

Number of Chip A = $\frac{40000}{7812.5} \times 2 = 10.24 \approx 11$

Number of Chip B = $\frac{40000}{14648.43} \times 2 = 5.461 \approx 6$

Number of Chip C = $\frac{40000}{29296.875} \times 2 = 2.730 \approx 3$

Considering all the specifications in Table 1, which chip type you would pick and why – be thorough. It's more than raw processing speed that counts.

	CHIP A	CHIP B ✓	CHIP C
No of chips reqd	11	6	3
Clock speed (MHz)	80	150	300
Avg power at 50% load (W)	0.7	1.2	2.6
Price	\$ 2.00	\$ 3.50	\$ 9.00
Total power (W)	7.7	7.2	7.8
Total price	\$ 22.00	\$ 21.00	\$ 27.00

Chip B is used as it has least total power consumption and also least price.

Answer this question about ADC:

The transducer in the sensor has a SNR of 85 dB. The instrument must sample the transducer up to 10 times at 1000 Hz and then go quiet while the DSP array performs the cross-correlations. Calculate ENOB. Knowing that the instrument must be light in mass and operate at low power, select an appropriate ADC type for this instrument from Table 2.

Sampling Rate = 10 samples × 1000 Hz
= 10 K samples /sec.

Explain the reasons for your selection.

Table 2. Some specifications for different ADCs that you may pick.

	Resolution (bits)		Sampling rate		Power Demand (W)		Physical size (mm)
ADC type	max.	min.	max.	min.	max.	average	wide x long x high
Flash	10	4	1 Gs/s	10 Ms/s	4	2	20 x 40 x 4
Pipeline	16	8	500 Ms/s	1 Ms/s	1	0.7	20 x 20 x 2
SAR	16	10	1 Ms/s	10 Ms/s	0.2	0.01	4 x 4 x 1
Σ-Δ	20	16	10 Ks/s	0.01 s/s	0.01	0.0001	4 x 4 x 1

$\frac{85 - 1.76}{6.02} = 13.82 \approx 14$

Calculate ENOB from the required SNR: ENOB = ____________________

nave Two options : i) SAR ADC – resolution lies in between and has least price and less power demand also size is less.
ii) ΣΔ ADC
└ resolution is less than minimum of chip but po demand, price and size are very low.

Extra credit: (5 points)

Assuming the following power demands, complete the table and calculate the minimum capacity of the battery in Amp-hour (assume the battery is 18 VDC).

so ΣΔ chip will be best chip, otherwise, SAR is also an admitable option.

	Chip A			Chip B			Chip C		
Average Power/subsystem	Qty.	Single part	Sub-total	Qty.	Single part	Sub-total	Qty.	Single part	Sub-total
gas sampler, pump, power (W)	1	1.5	1.5	1	1.5	1.5	1	1.5	1.5
microntroller (W)	1	0.8	0.8	1	0.8	0.8	1	0.8	0.8
transducer, ADC (W)	1	0.5	0.5	1	0.5	0.5	1	0.5	0.5
DSP chips (W)	11	0.7	7.7	6	1.2	7.2	3	2.6	7.8
		Total (W) =	10.5		Total (W) =	10		Total (W) =	10.6
power capacity for 8 hours (W-hr)			84			80			84.8
battery capacity @18 V (Amp-hr)			4.67			4.44			4.71