FIR FILTERS (FINITE IMPULSE RESPONSE FILTERS)

The General FIR Filter

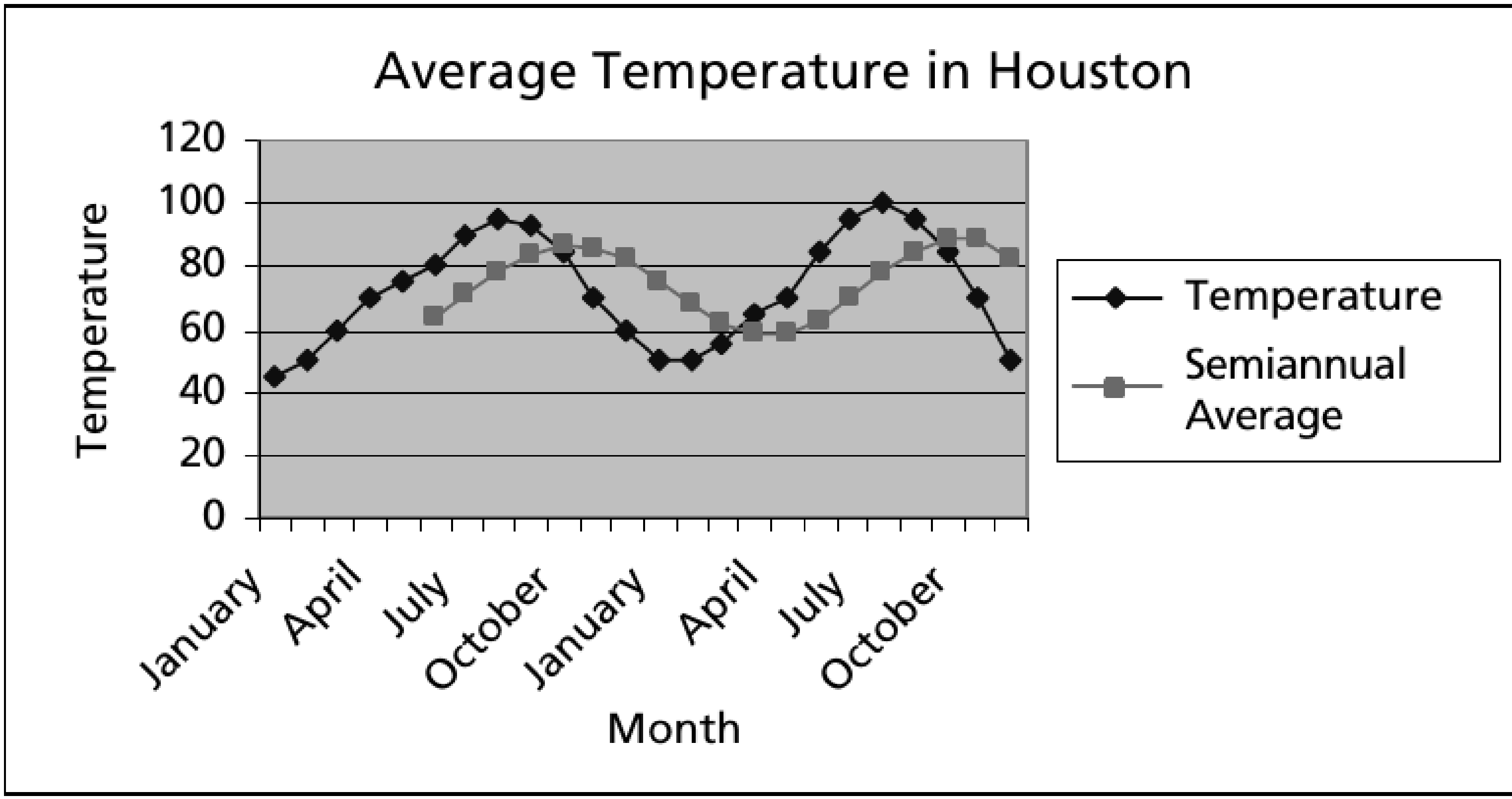
- The general form for the FIR filter is:

$$y(n) = \sum_{i=0}^{K} b_i x(n-i)$$

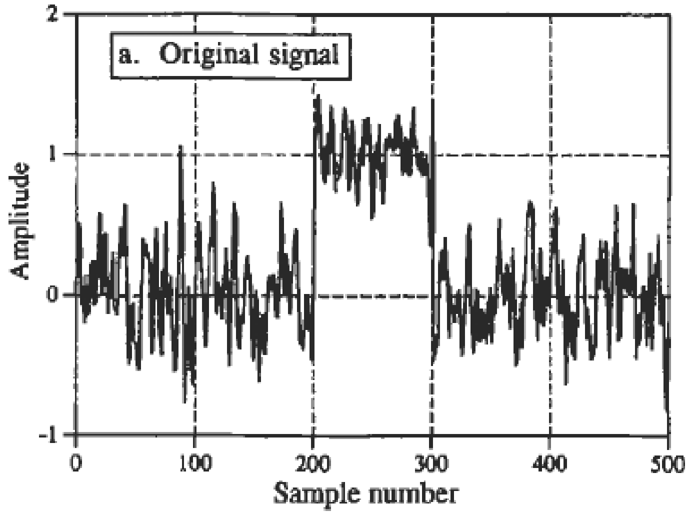
$$= b_0\, x(n) + b_1 x(n-1) + b_2 x(n-2) + \cdots + b_K x(n-K)$$

MOVING AVERAGE FILTER

Moving average filter



Noise reductions vs. step response



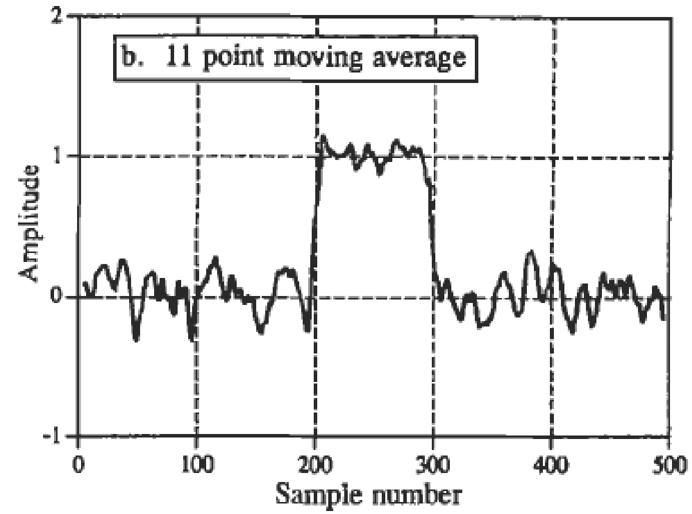
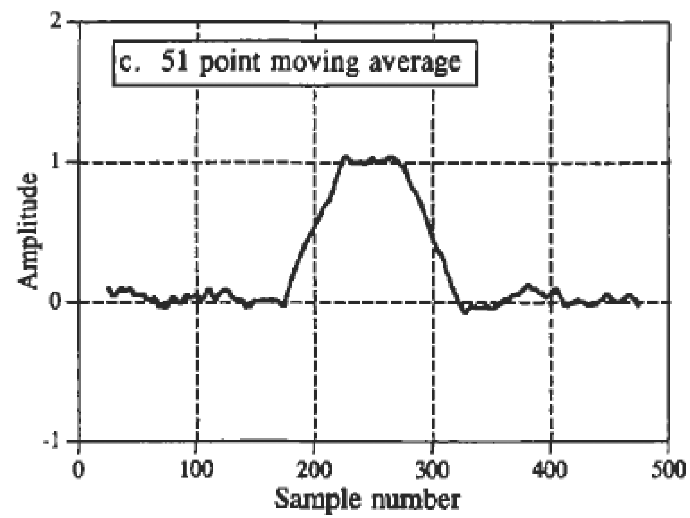
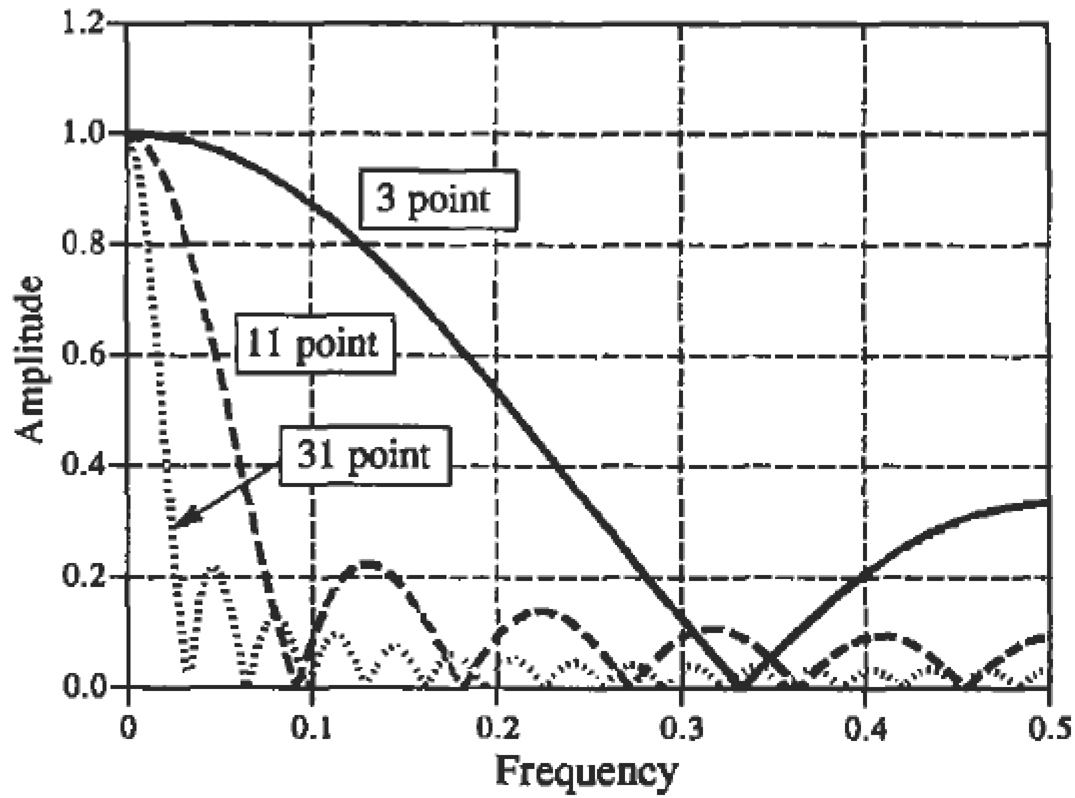


FIGURE 15-1
Example of a moving average filter. In (a), a rectangular pulse is buried in random noise. In (b) and (c), this signal is filtered with 11 and 51 point moving average filters, respectively. As the number of points in the filter increases, the noise becomes lower; however, the edges becoming less sharp. The moving average filter is the *optimal* solution for this problem, providing the lowest noise possible for a given edge sharpness.



Frequency response



Moving average is an exceptionally good smoothingf ilter (the action in the time domain), but an exceptionally bad low-pass filter (the action in the frequency domain).

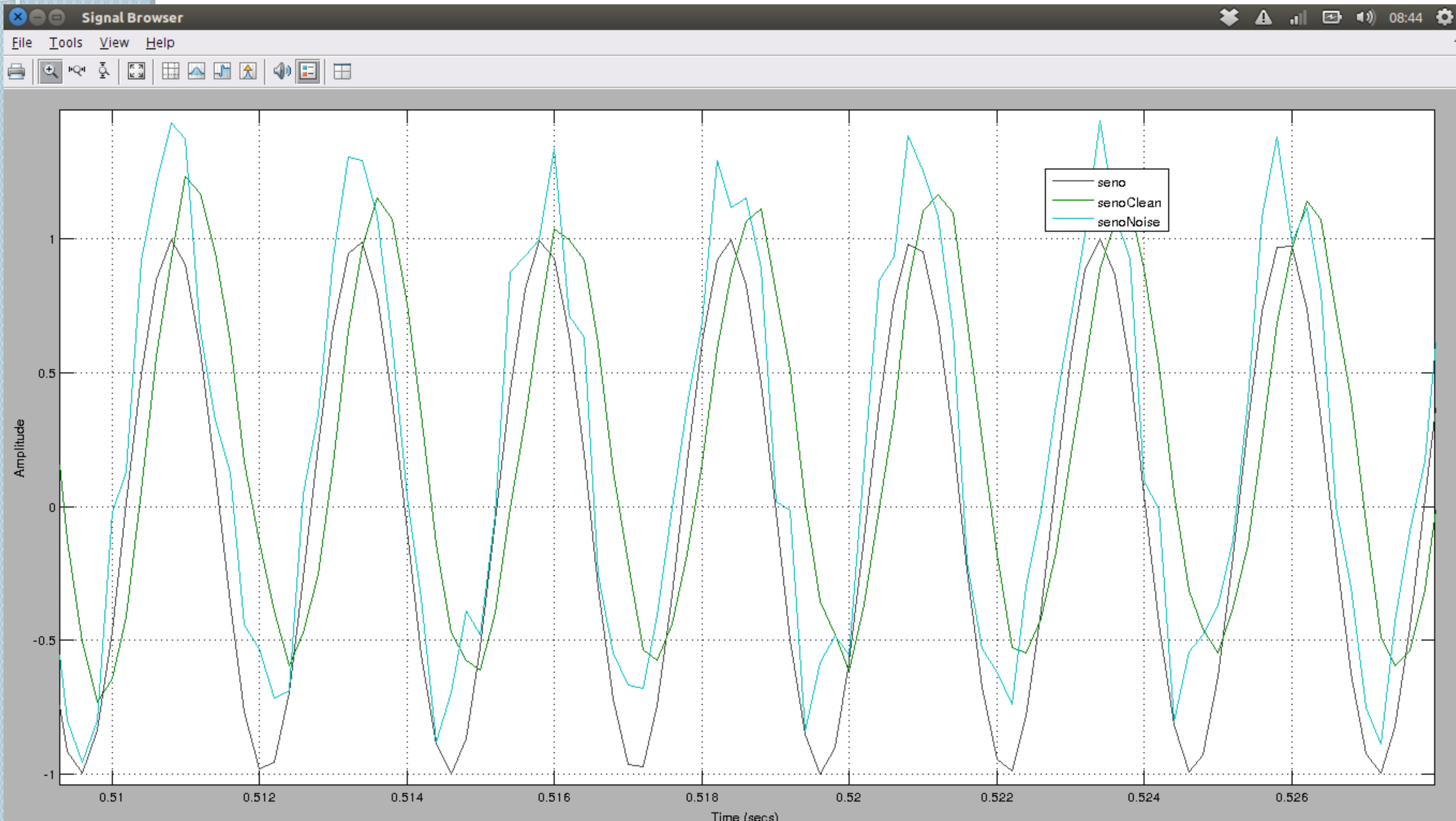
Matlab

- The filter …

$$a(1)y(n) = b(1)x(n) + b(2)x(n-1) + \ldots + b(N_b)x(n-N_b+1)$$
$$- a(2)y(n-1) - \ldots - a(N_a)y(n-N_a+1)$$

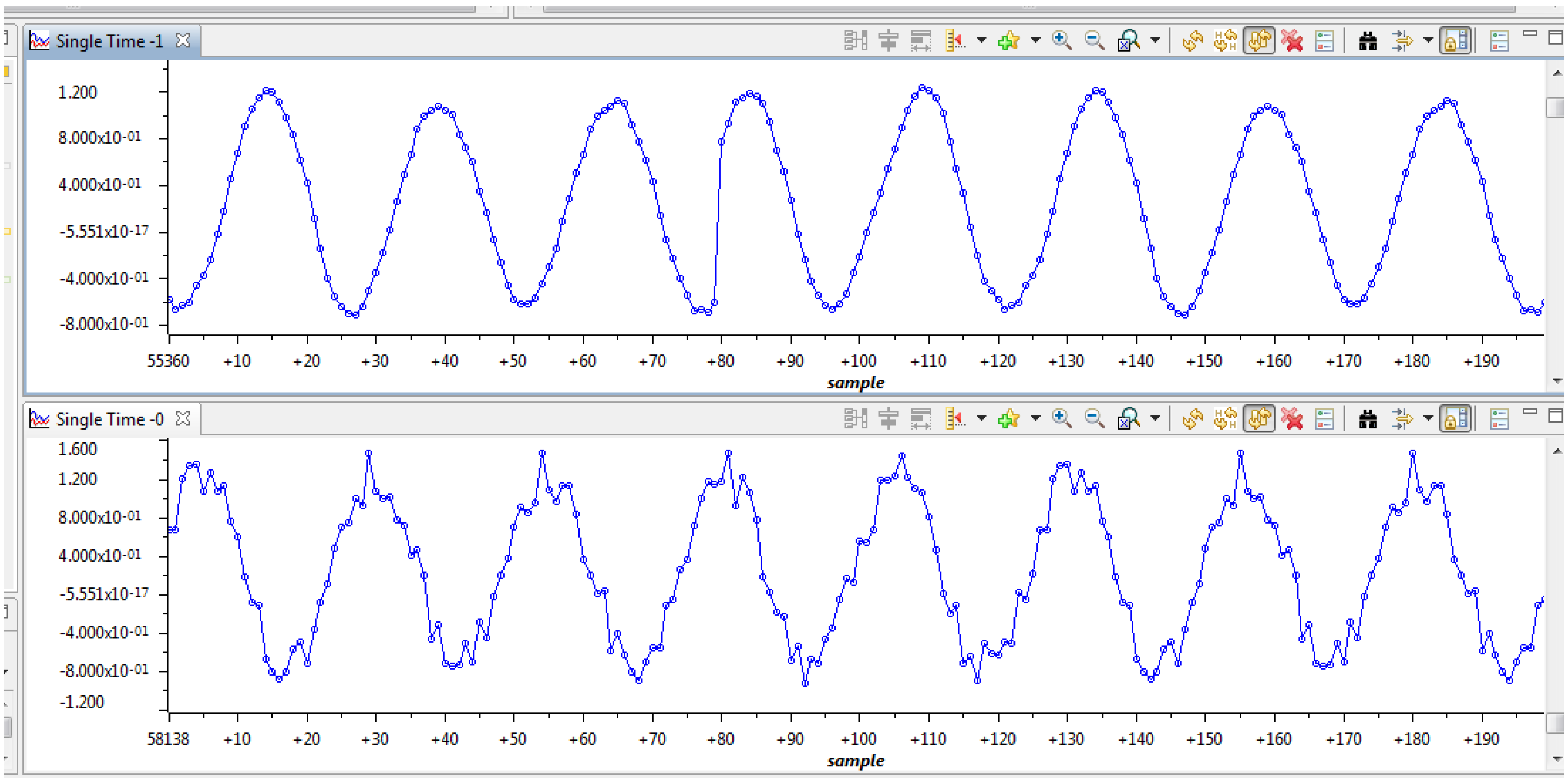
New to MATLAB? Watch this Video, see Examples, or read Getting

```
>> t=0:0.01:50*pi;
>> seno=sin(50*t);
>> senoNoise=sin(50*t)+0.5*rand(1,length(t));
>> a=1;
>> b = [1/4 1/4 1/4 1/4];
>> senoClean=filter(b,a,senoNoise);
>> sptool
>>
```



Signal Browser
08:44
File
Tools
View
Help
seno
senoClean
senoNoise
1
0.5
0
-0.5
-1
Amplitude
0.51
0.512
0.514
0.516
0.518
0.52
0.522
0.524
0.526
Time (secs)

Práctica



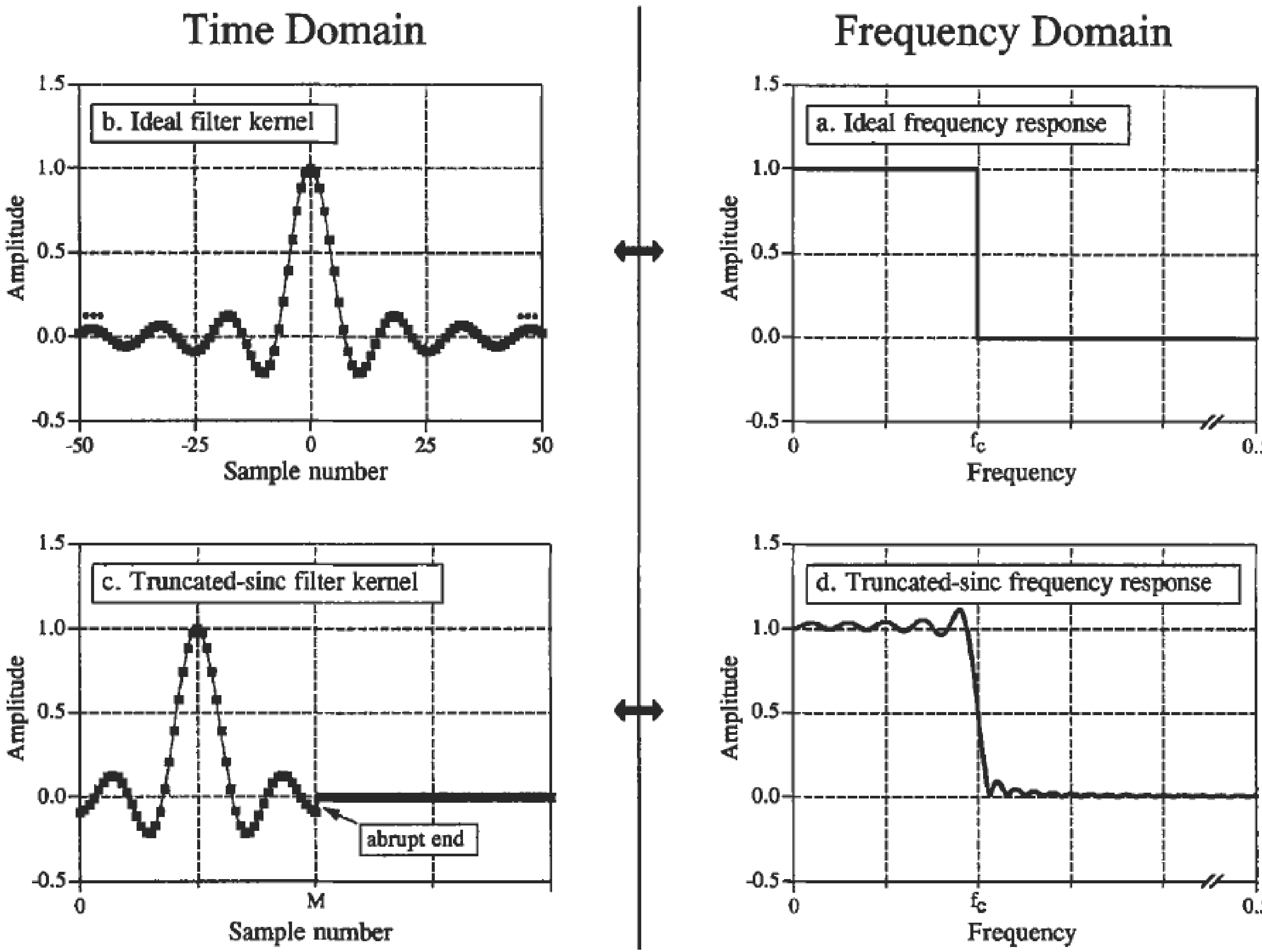
Single Time -1
1.200
8.000x10-01
4.000x10-01
-5.551x10-17
-4.000x10-01
-8.000x10-01
55360 +10 +20 +30 +40 +50 +60 +70 +80 +90 +100 +110 +120 +130 +140 +150 +160 +170 +180 +190
sample
Single Time -0
1.600
1.200
8.000x10-01
4.000x10-01
-5.551x10-17
-4.000x10-01
-8.000x10-01
-1.200
58138 +10 +20 +30 +40 +50 +60 +70 +80 +90 +100 +110 +120 +130 +140 +150 +160 +170 +180 +190
sample

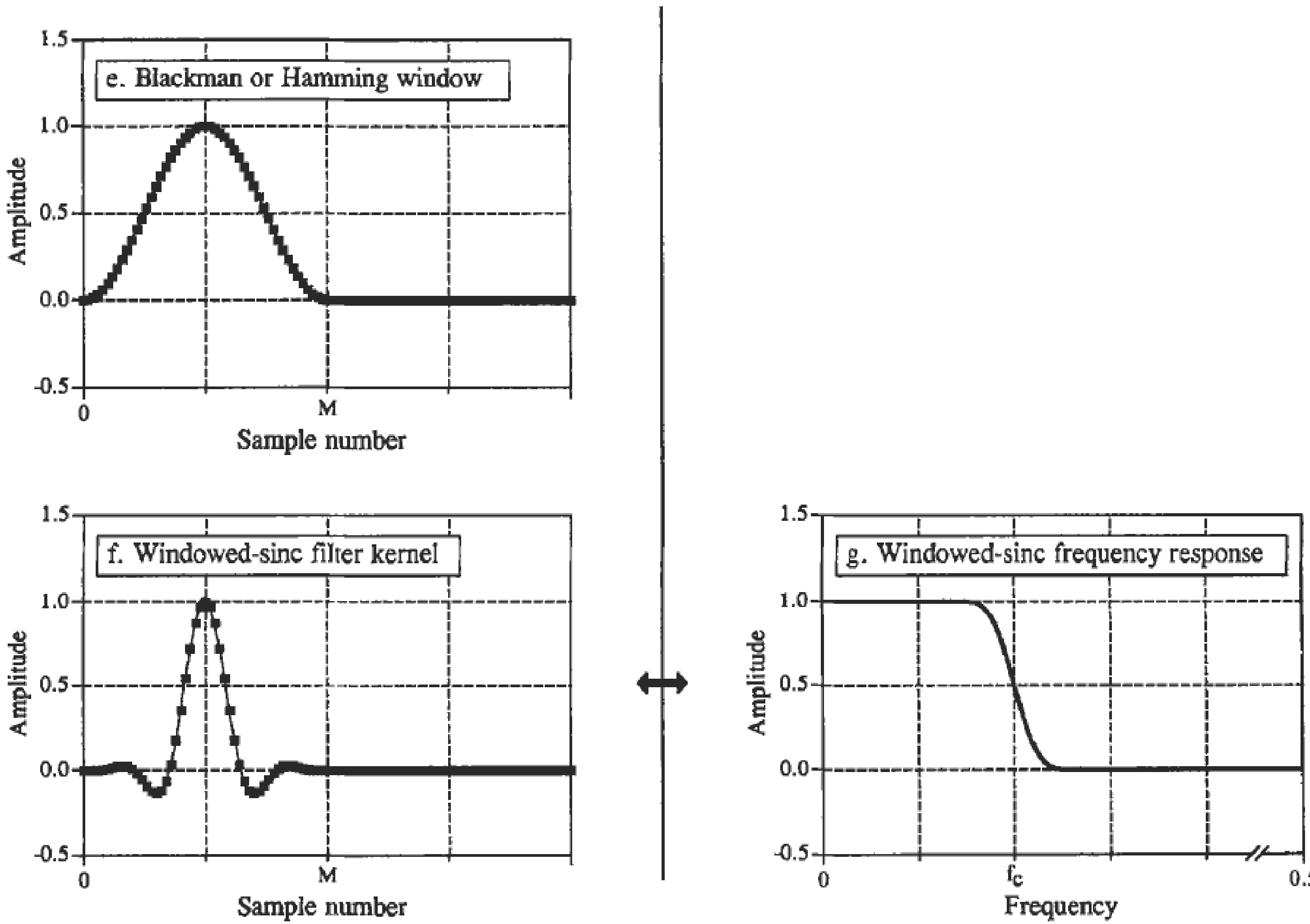
WINDOWED-SINC FILTERS

- Windowed-sinc filters are used to separate one band of frequencies from another. They are very stable and have exceptional frequency domain characteristics are obtained at the expense of poor performance in the time domain, including excessive ripple and overshoot in the step response. When carried out by standard convolution, windowed-sinc filters are easy to program, but slow to execute.

sinc function

$$h[i] = \frac{\sin(2\pi f_C i)}{i\pi}$$


Time Domain
Frequency Domain
b. Ideal filter kernel
a. Ideal frequency response
c. Truncated-sinc filter kernel
d. Truncated-sinc frequency response
abrupt end
Amplitude
Sample number
Frequency
1.5
1.0
0.5
0.0
-0.5
-50
-25
0
25
50
M
fc
0.5



e. Blackman or Hamming window
1.5
1.0
0.5
0.0
-0.5
Amplitude
0
M
Sample number
f. Windowed-sinc filter kernel
1.5
1.0
0.5
0.0
-0.5
Amplitude
0
M
Sample number
g. Windowed-sinc frequency response
1.5
1.0
0.5
0.0
-0.5
Amplitude
0
fc
0.5
Frequency

Hamming and Blackman window

$$w[i] = 0.54 - 0.46 \cos(2\pi i / M)$$

$$w[i] = 0.42 - 0.5 \cos(2\pi i / M) + 0.08 \cos(4\pi i / M)$$

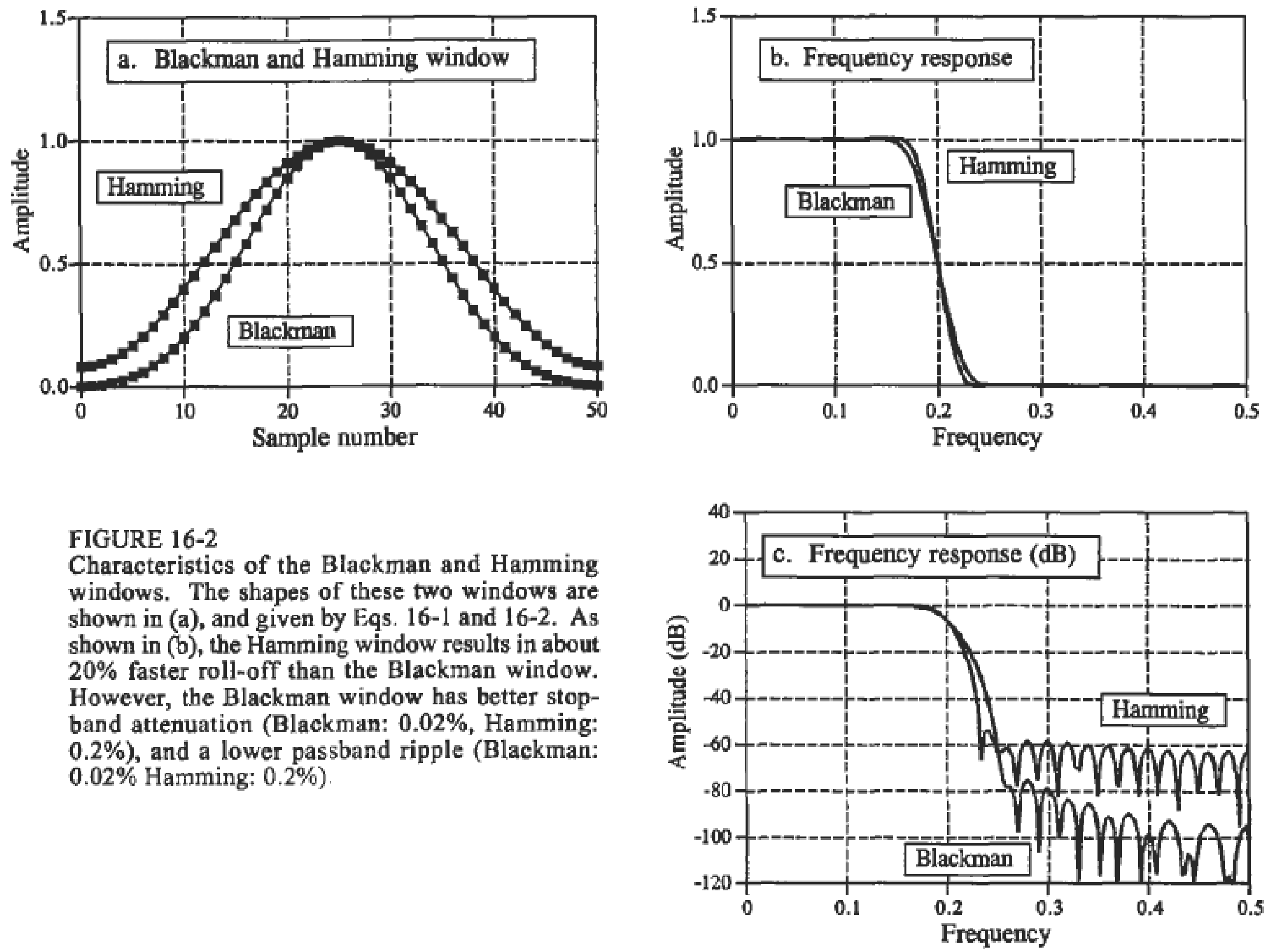
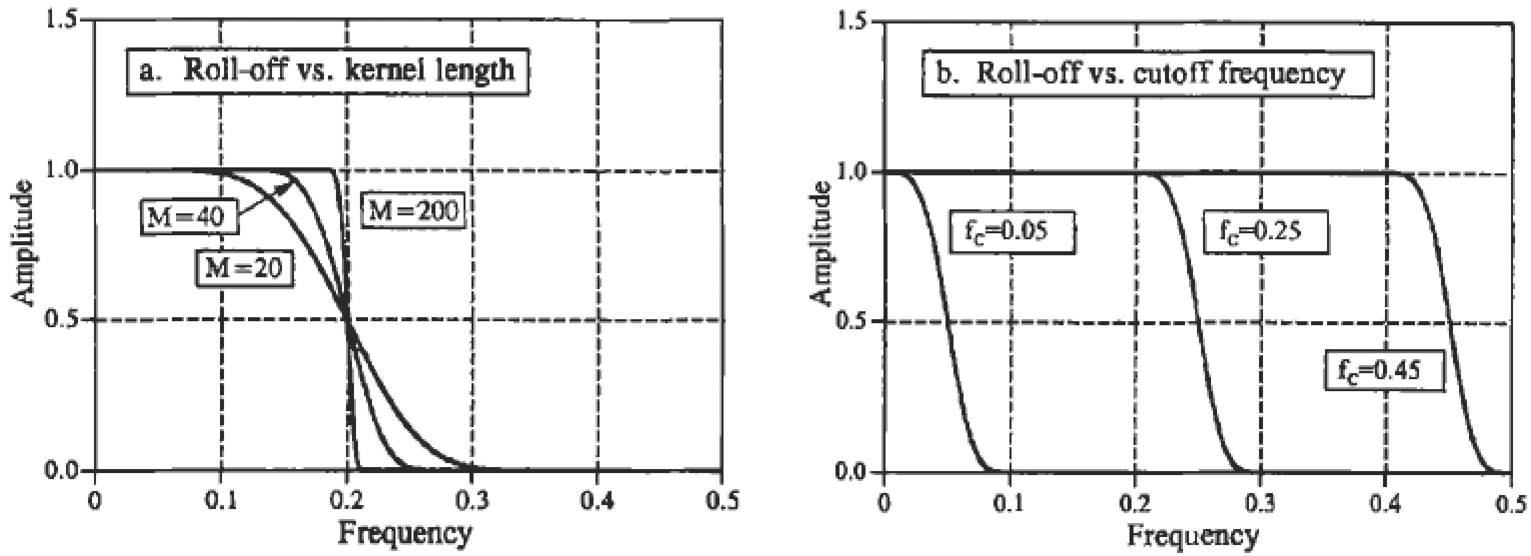


FIGURE 16-2
Characteristics of the Blackman and Hamming windows. The shapes of these two windows are shown in (a), and given by Eqs. 16-1 and 16-2. As shown in (b), the Hamming window results in about 20% faster roll-off than the Blackman window. However, the Blackman window has better stop-band attenuation (Blackman: 0.02%, Hamming: 0.2%), and a lower passband ripple (Blackman: 0.02% Hamming: 0.2%).

Designing the Filter (f_c and M)



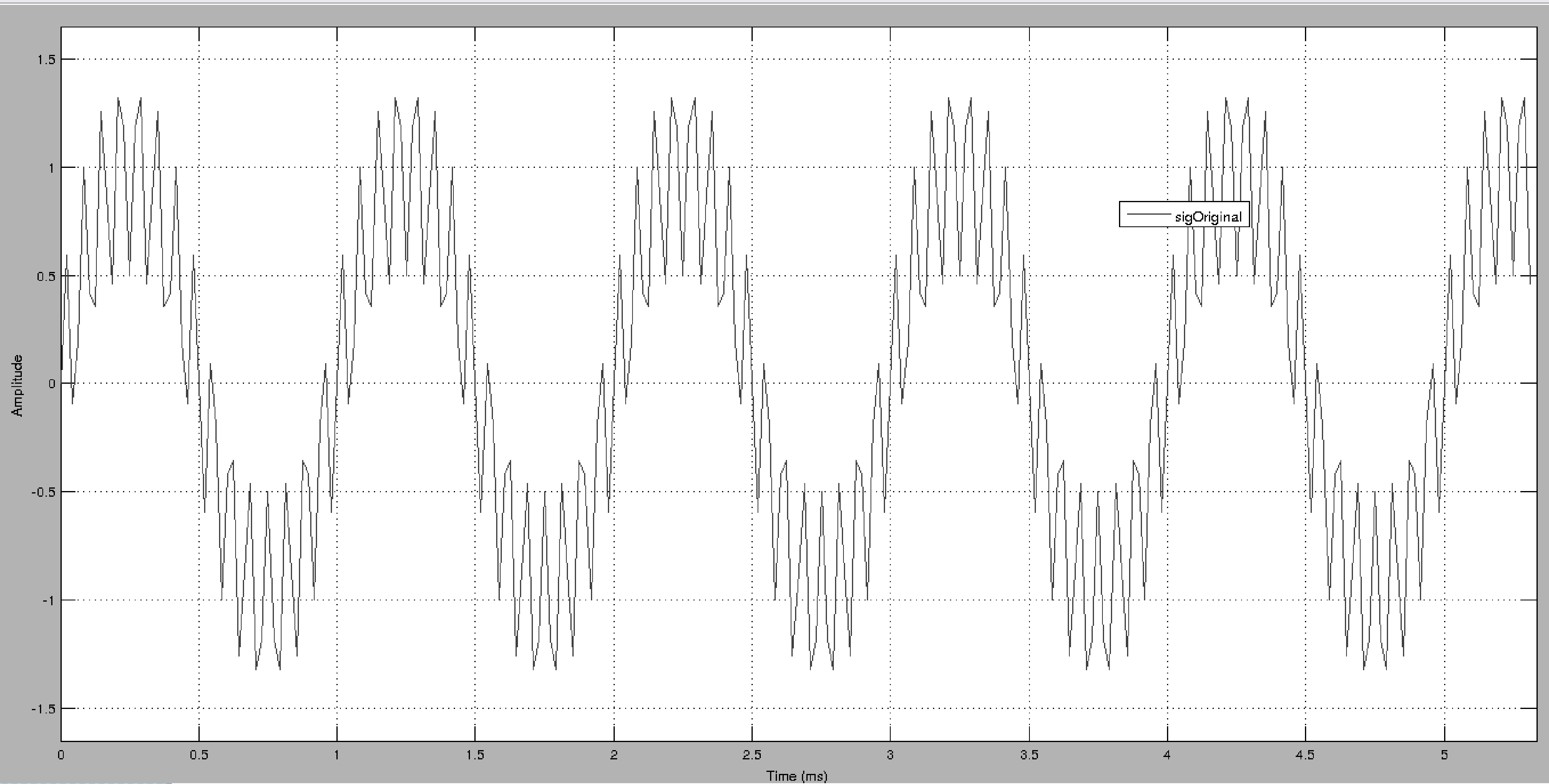
$$M \approx \frac{4}{BW}$$

$$h[i] = K \frac{\sin(2\pi f_c (i - M/2))}{i - M/2} \left[0.42 - 0.5\cos\left(\frac{2\pi i}{M}\right) + 0.08\cos\left(\frac{4\pi i}{M}\right) \right]$$

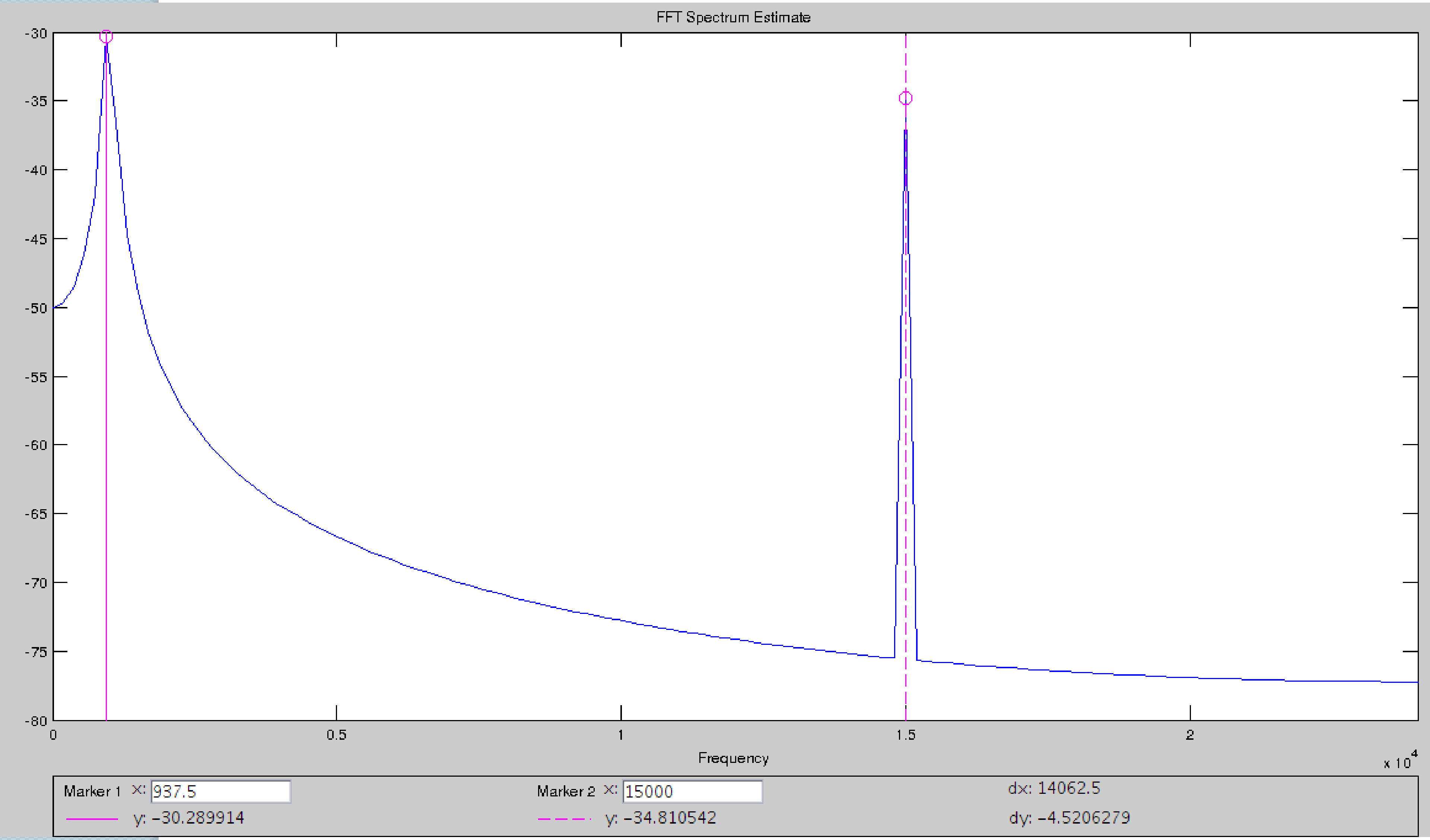
Exercise

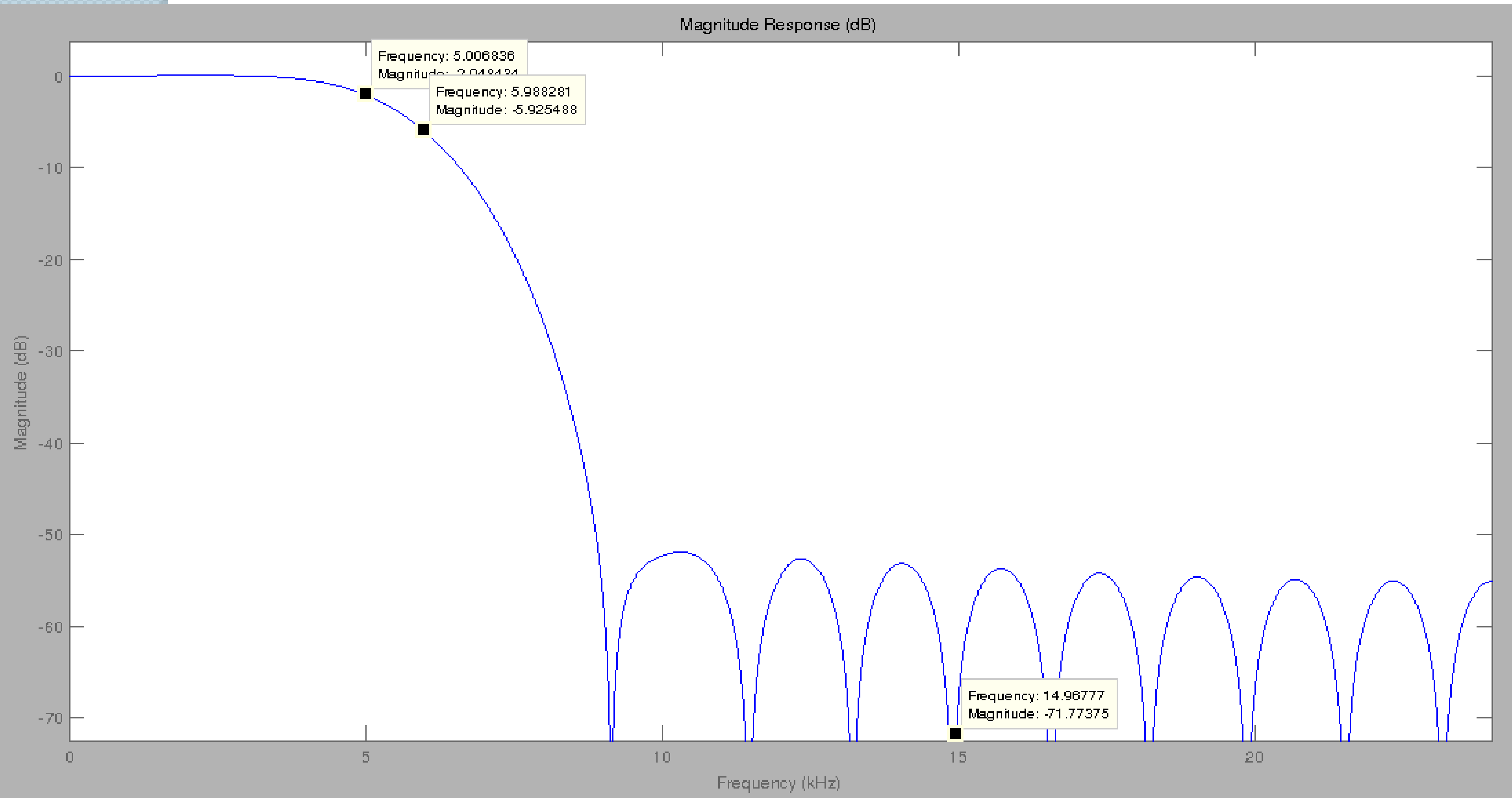
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
>> sample_rate = 48000;
>> nsamples = 256;
>> t = (0:nsamples-1) / sample_rate;
>> s1=sin(2*pi*1000*t);
>> s2=0.5*sin(2*pi*15000*t);
>> s3=s1+s2;
>> %importarlos a sptool con una velocidad de muestreo de 48000
>> sptool
>> %analizar el espectro de frecuencias e identificar ambas frecuencias (1000 y 15000)
>> %establecer frecuencia de corte
>> cutoff_hz = 6000;
>> nyq_freq = sample_rate / 2;
>> %normalizacion de la frecuencia de corte
>> cutoff_norm = cutoff_hz / nyq_freq;
>> %orden del filtro
>> order = 28;
>> %ontención de los coeficientes del filtro
>> fir_coeff = fir1(order, cutoff_norm);
>> %filtrar la señal
>> filtered_signal = filter(fir_coeff, 1, s3);
>> %analizar la señal filtrada y su espectro de frecuencias
>> fir_coeff

fir_coeff =

  Columns 1 through 10

   -0.0018   -0.0016    0.0000    0.0037    0.0081    0.0085   -0.0000   -0.0174   -0.0341   -0.0334

  Columns 11 through 20

    0.0000    0.0676    0.1522    0.2229    0.2505    0.2229    0.1522    0.0676    0.0000   -0.0334

  Columns 21 through 29

   -0.0341   -0.0174   -0.0000    0.0085    0.0081    0.0037    0.0000   -0.0016   -0.0018

fx >>
```

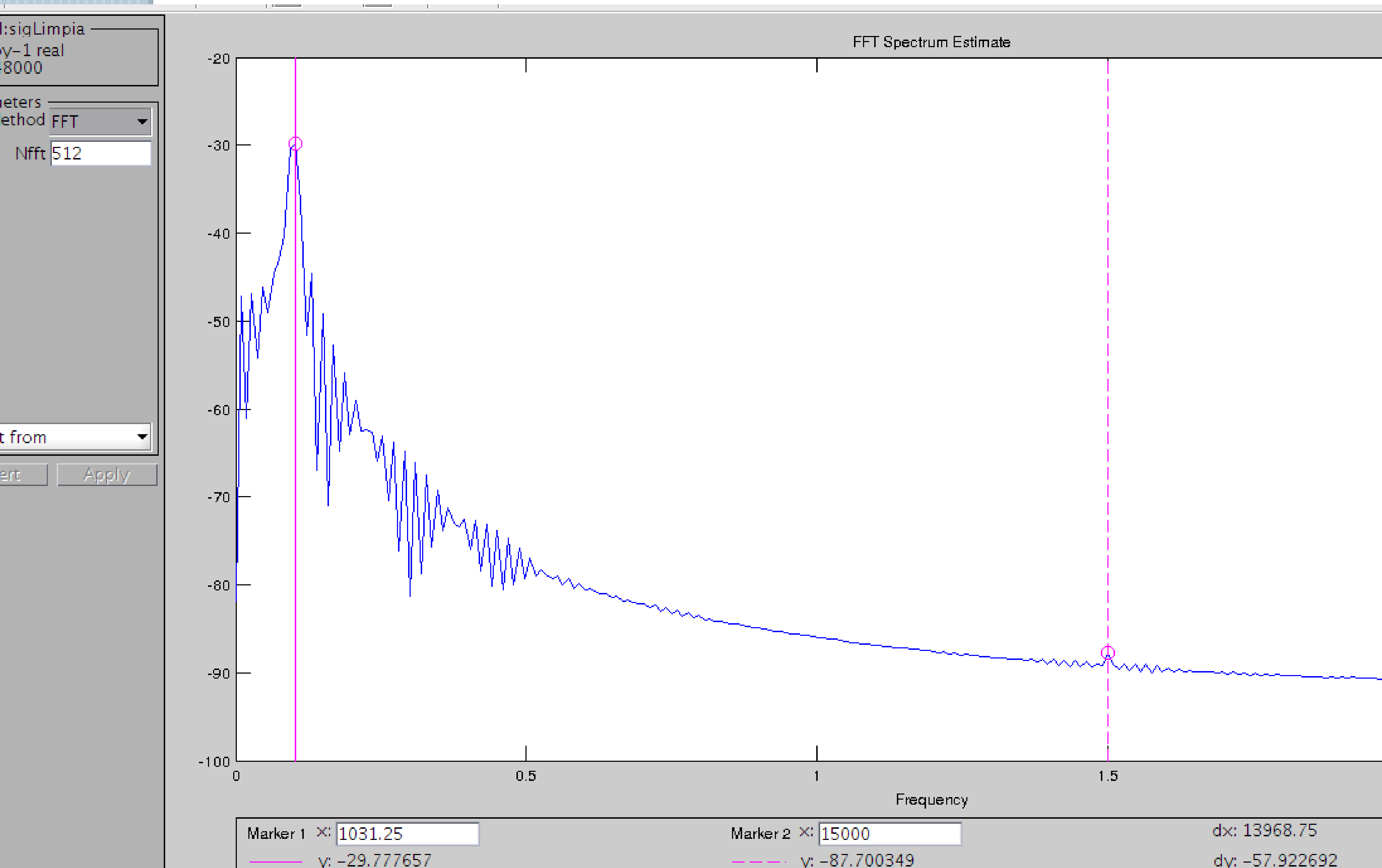


sigOriginal
1.5
1
0.5
0
-0.5
-1
-1.5
Amplitude
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
Time (ms)

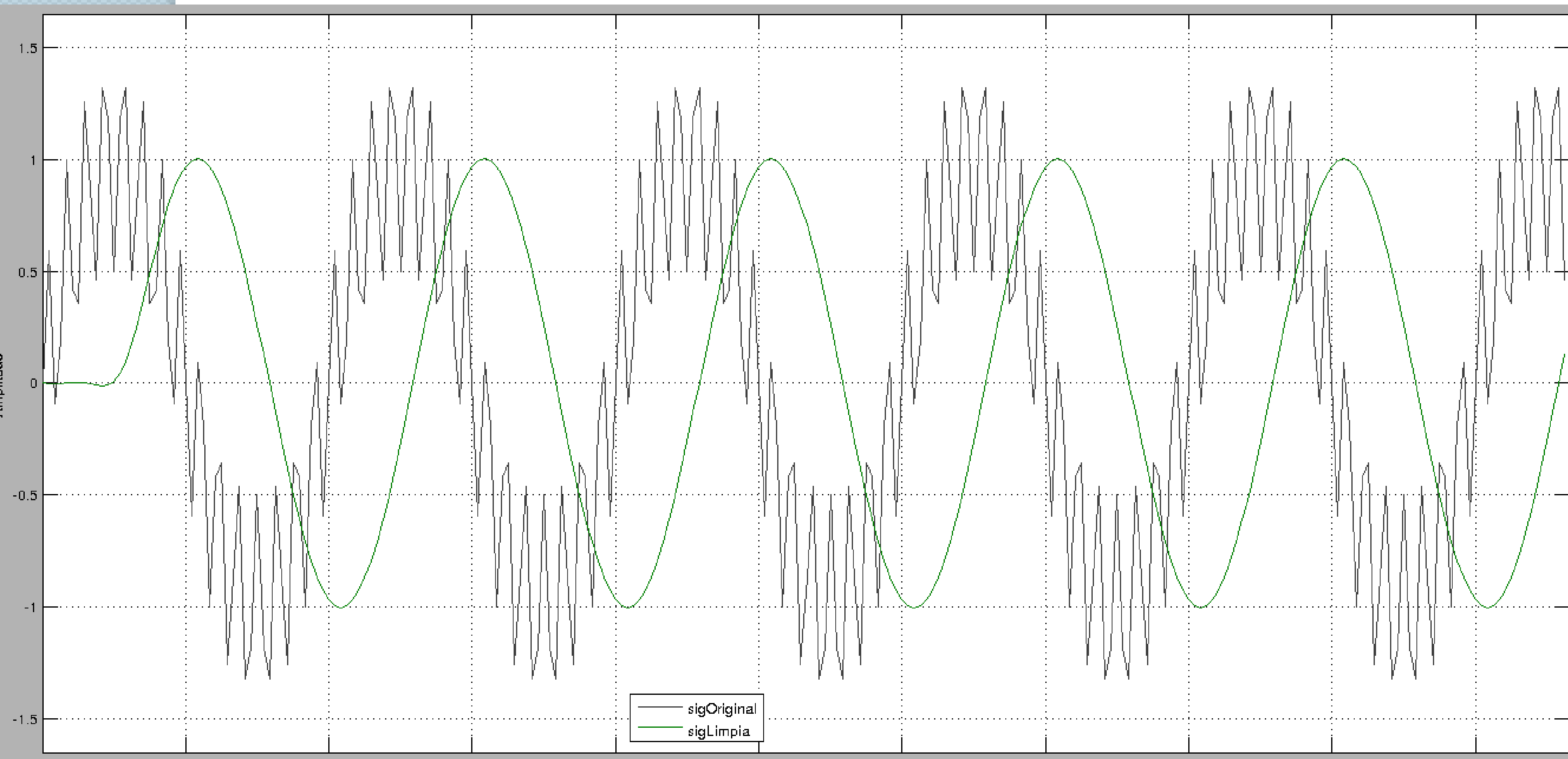

FFT Spectrum Estimate
-30
-35
-40
-45
-50
-55
-60
-65
-70
-75
-80
0
0.5
1
1.5
2
Frequency
x 10^4
Marker 1 X: 937.5
y: -30.289914
Marker 2 X: 15000
y: -34.810542
dx: 14062.5
dy: -4.5206279


Magnitude Response (dB)
Frequency: 5.006836
Frequency: 5.988281
Magnitude: -5.925488
Frequency: 14.96777
Magnitude: -71.77375
0
-10
-20
-30
-40
-50
-60
-70
Magnitude (dB)
0
5
10
15
20
Frequency (kHz)

```
Numerator:
-0.0018225230387647777842430318884225926O1
-0.0015879293765789033367891169845620424 89
 0.00000000000000000122600883254451600608 6
 0.0036977508277888340888772678027862639 28
 0.0080754302633316028120491480990494892 47
 0.0085302216834420643842840092929691309 11
-0.000000000000000000427345674999743423OO54
-0.0173976983943203815374811682659128564 41
-0.0341458607045116643852189497465587919 58
-0.0333591564739013426987845889470918336 88
 0.000000000000000000807356214645249543 6928
 0.0676308394716027272419722748963977210 22
 0.1522061834698553717259983386611565947 53
 0.2229246956249595623678061429018271155 66
 0.2504960932941937556961420341394841670 99
 0.2229246956249595623678061429018271155 66
 0.1522061834698553717259983386611565947 53
 0.0676308394716027272419722748963977210 22
 0.000000000000000000807356214645249543 6928
-0.0333591564739013426987845889470918336 88
-0.0341458607045116643852189497465587919 58
-0.0173976983943203815374811682659128564 41
-0.000000000000000000427345674999743423OO54
 0.0085302216834420643842840092929691309 11
 0.0080754302633316028120491480990494892 47
 0.0036977508277888340888772678027862639 28
 0.00000000000000000122600883254451600608 6
-0.0015879293765789033367891169845620424 89
-0.0018225230387647777842430318884225926O1
Denominator:
1
```



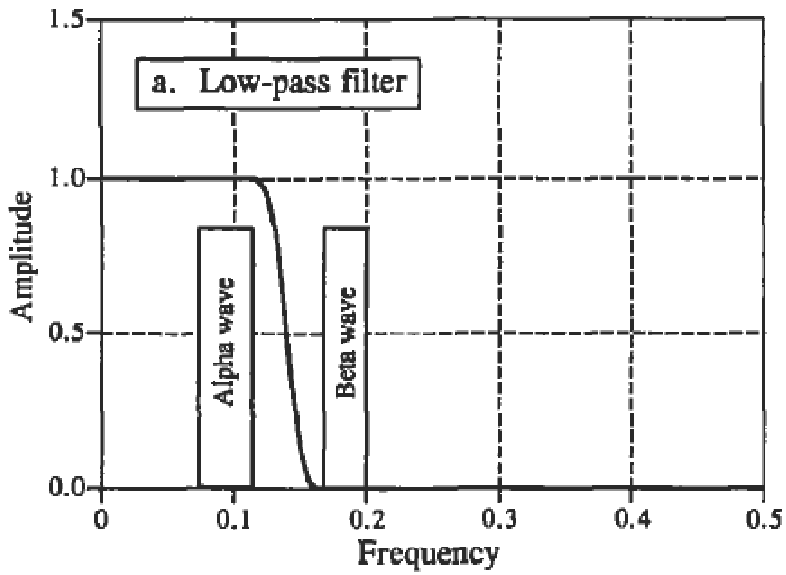
l:sigLimpia
y-1 real
8000
eters
ethod FFT
Nfft 512
t from
ert
Apply
FFT Spectrum Estimate
-20
-30
-40
-50
-60
-70
-80
-90
-100
0
0.5
1
1.5
Frequency
Marker 1 x: 1031.25
y: -29.777657
Marker 2 x: 15000
y: -87.700349
dx: 13968.75
dy: -57.922692

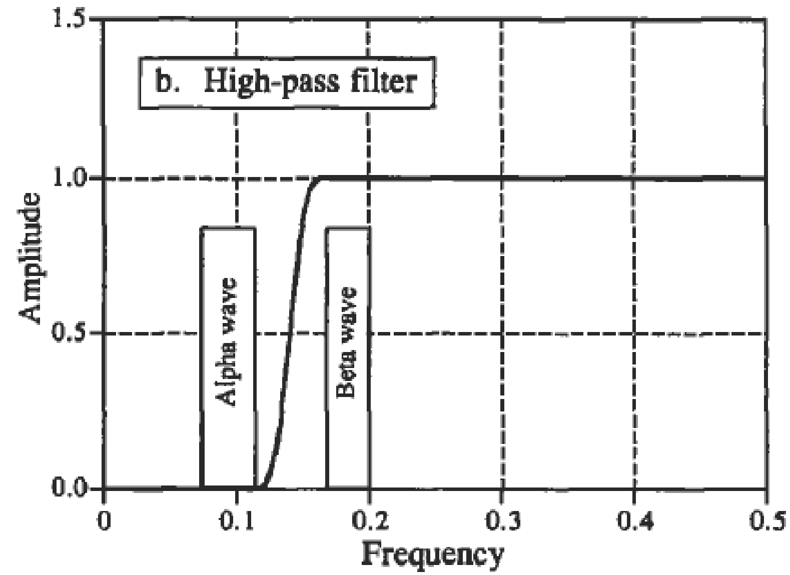

1.5
1
0.5
0
-0.5
-1
-1.5
sigOriginal
sigLimpia

Exercise

- Now eliminate the 1000 Hz frequency!!!

Example of Windowed-Sinc Filters, electroencephalogram





Example of Windowed-Sinc Filters, band-passfilter