$\cos\omega = e^{j\omega} + e^{-j\omega}$

Electrical Engineering and Computer Science
Syracuse University

ELE 352, Spring 2015

Final Project

The students are given a piece of audio clip from a very popular movie to enjoy. Not surprisingly, it has been doctored by your TA and, without some cleaning up effort, the sound is genuinely annoying! At first, a single frequency real sinusoid was sneaked into the audio piece. To make the matter more complicated, the distorted signal was further sent through a nine-tap FIR system (i.e., the system impulse response has length nine). Thus, the venture starts ...

The file finalproject.mat can be downloaded from the course website. The first task, of course, is to save it in a specified directory. Load the file (use **"load finalproject"** in MATLAB – you need to make sure it is in the correct directory). You will find out (using the **"whos"** commend in MATLAB) that there are six variables:

- a training sequence pair **straining** and **xtraining** that are related by $xtraining[n] = h[n] \otimes straining[n]$.
- a vector **y** that contains the doctored sound file, obtained by

$$y[n] = (\underbrace{s[n] + \cos(2\pi f \frac{n}{fs})}_{x[n]}) \otimes h[n]$$

(a) $X[k] = h[k]\,S[k]$
$h[k] = \frac{x_k}{s_k}$
$h'[n]$

  where the $h[n]$ is the same as above.

$x[n] = \frac{y[n]}{h[n]}$

- f_s: the sample frequency (in Hz);
- **N**: is the number of taps of the FIR system $h[n]$ used to distort the signal.
- **stest**. This is an original sound piece that has the same sampling frequency. This is for the students to test their algorithm (more on this later).

$y[n] = x[n] \otimes h[n]$
$y_k = x[k]\,H[k]$
$x[k] = \frac{y[k]}{H_k}$
$x[n] =$

If you use

sound(y,f_s);

in MATLAB, you will hear it (make sure you turn on your speakers first![1]).

(3) noise determine

Objective:

Recover, using your best effort, the original sequence $s[n]$ from the tainted output $y[n]$. This can be broken down into four tasks, all indispensable to obtain the ultimate prize:

(4)

- Use the training sequences **straining[n]** and **xtraining[n]** to find $h[n]$. (Hint: Use DFT with zero padding to find H[k] first, and then change to time domain. How many zeros do you add in zero padding?) added 8 zeros : found h[n]
- Using the obtained $h[n]$, recover the following sequence from $y[n]$ found x[n]

$$x[n] = s[n] + \cos(2\pi f \frac{n}{f_s})$$

  (Hint: Again, use DFT with zero padding to find the sequence in DFT domain first and then change it to time domain.)

[1]Disclaimer: Be sure to cover your ears if you have a decent pair of speakers tuned on at maximum volumn.

Manuel Nyagisere

(a) identify $h[n]$

$$x_{training}[n] = h[n] \circledast s_{training}[n]$$

DFT $\quad x_{training}[k] = h[k]\, s_{training}[k]$

$$h[k] = \frac{x_{training}[k]}{s_{training}[k]}$$ $\longleftrightarrow$ straining[n] is zero padded because it's shorter than xtraining[n); 8 zeros are added.

$$\boxed{h[n] = ifft(h[k])}$$

(b) Recover $s[n] + \cos(2\pi f \frac{n}{fs})$

let $s[n] + \cos(2\pi f \frac{n}{fs}) = x[n]$

so $\quad y[n] = x[n] \circledast h[n]$

DFT $\quad Y[k] = X[k]\, H[k]$

$$X[k] = \frac{Y[k]}{H[k]}$$ $\longleftrightarrow$ h[n] is zero padded to make it same length as y[n]; fft(hn, 100008)

$$\boxed{x[n] = ifft(X[k]) = s[n] + \cos(2\pi f \frac{n}{fs})}$$

(c) frequency of single tone interference.

$$X[k] = X(e^{j\omega})\Big|_{\omega = \frac{2\pi(n-1)}{N}}$$

$$\boxed{\begin{aligned} \omega_1 &= \frac{2\pi(5000)}{100008} = 0.3141 \\ \omega_2 &= \frac{2\pi(95007)}{100008} = 1.9\pi \end{aligned}}$$

using plots [xmax, idxmax] = max(abs(xk(1: round(length(xk/2))):))

idxmax$_1$ = 50001

Idxmax$_2$ = 95008

(d) $h[n] = [1 \ \alpha \ \beta]$

$H(z) = 1 + \alpha z^{-1} + \beta z^{-2} \longleftrightarrow$ simple notch filter design
$H(z) = 1 - 2\cos\omega_0 z^{-1} + z^{-2}$

$$= \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

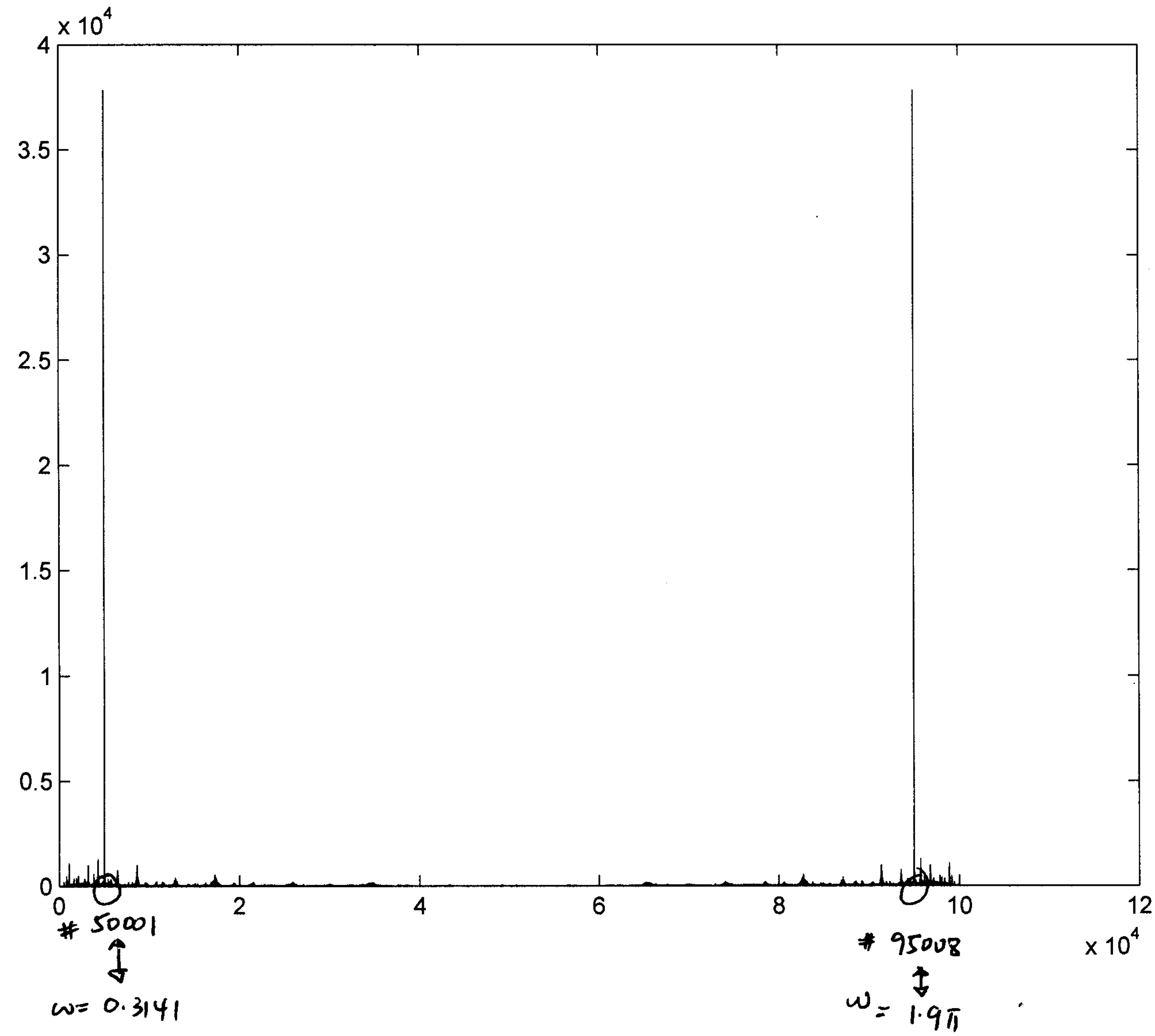
$$= e^{j\omega_0}, \ e^{-j\omega_0} = -2\cos\omega$$

$$\boxed{h[n] = [1 \ -2\cos(\omega_0) \ 1]}$$

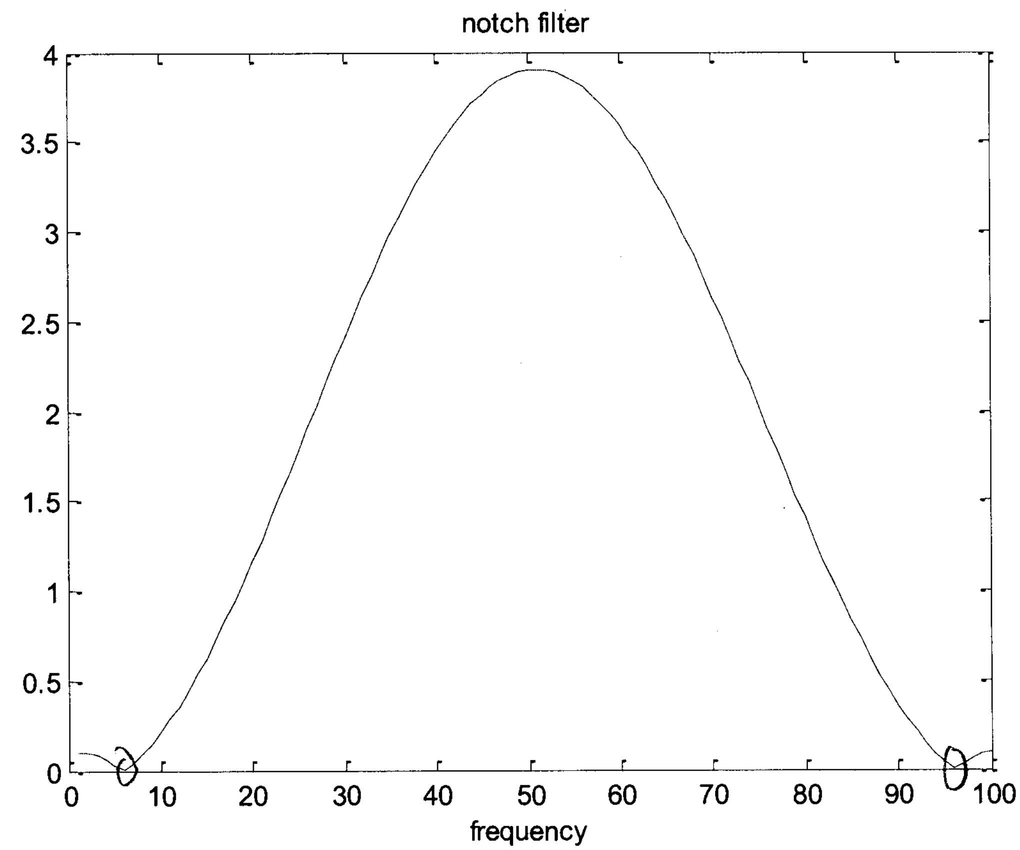
impulse response of the filter in time domain.

~~h[n] = ... 2cos(ω₀)~~

(e) the original audio clip is not recovered perfectly because our filter is not ideal.



x 10^4
4
3.5
3
2.5
2
1.5
1
0.5
0
0
2
4
6
8
10
12
x 10^4
50001
ω= 0.3141
95008
ω= 1.9π


notch filter
4
3.5
3
2.5
2
1.5
1
0.5
0
0
10
20
30
40
50
60
70
80
90
100
frequency