Loud Speaker Equalization Project

By:
Chris Metzler
Abhijit Navlekar

Loud Speaker Equalization Project

By:
Chris Metzler
Abhijit Navlekar

Online:
< http://cnx.org/content/col11257/1.1/ >

C O N N E X I O N S

Rice University, Houston, Texas

Collection structure revised: December 14, 2010
PDF generated: February 6, 2011
For copyright and attribution information for the modules contained in this collection, see p. 12.

Table of Contents

Loud Speaker Equalization Project Overview 1

1 Procedure and Testing

1.1 Frequency Response 3
1.2 Sampling and Smoothing 4
1.3 Low Frequency Resonance 5
1.4 Inverting the Response 6
1.5 FIR Filter on TI Chip 7
1.6 Frequency Response Correction Verification 8

2 Loud Speaker Equalization Project Conclusions 11
Attributions 12

Loud Speaker Equalization Project Overview[1]

Loudspeaker Equalization

Overview

A loudspeaker converts sound from its electric form to an audible form. In doing so it introduces notable distortions (in both magnitude and phase) in the sound thus leading to poor sound quality. Under the assumption that the speaker behaves somewhat linearly, loudspeaker equalization has been developed to deal with such distortions and minimize them to a great extent. Loudspeaker equalization involves processing and correcting the frequency response of a loudspeaker so that the spectrum of the resulting output is close to the spectrum of the input. One way to implement loudspeaker equalization is to use MATLAB fir design tools and the TI 3245 EVM which has a built in FIR filter.

To find the frequency response of a speaker one can play white noise through the loudspeaker and record it onto a computer. The recording is then processed in MATLAB with the attached code to generate a frequency response of the white noise. Next invert the frequency response to find an ideal inverse filter to correct for the distortion of the speaker and give the system an overall flat frequency response. Finally use MATLAB's fir2 design tool to create a length 64 fir filter which we can be implemented onto the TI chip.

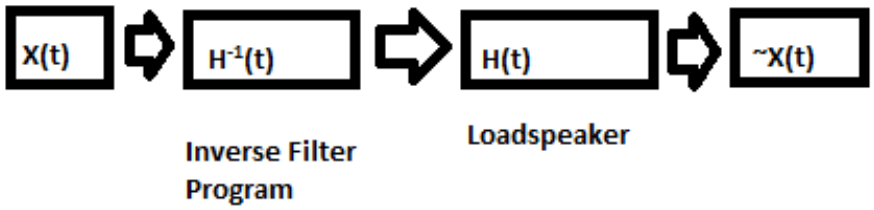

Figure 1

Objectives:

1. Find an accurate frequency response for the loudspeaker.
2. Invert this frequency response to get the inverse filter.

[1] This content is available online at <http://cnx.org/content/m36578/1.1/>.

3. Implement it on the chip.
4. Verify the speaker sounds better.

Chapter 1

Procedure and Testing

1.1 Frequency Response[1]

1.1.1 Procedure:

1.1.1.1 1. Frequency response:

The frequency response of the speaker describes everything from low frequency resonance to high frequency distortion. One of the ways to find a frequency response of a speaker is to blast white noise through the speaker and record the result. White noise contains all frequencies and thus can be used to describe the speaker's response to any input signal. To obtain the frequency response of a speaker, generate white noise and record the response and then use Matlab's fft command and a log log plot to view the response.

[1] This content is available online at <http://cnx.org/content/m36489/1.1/>.

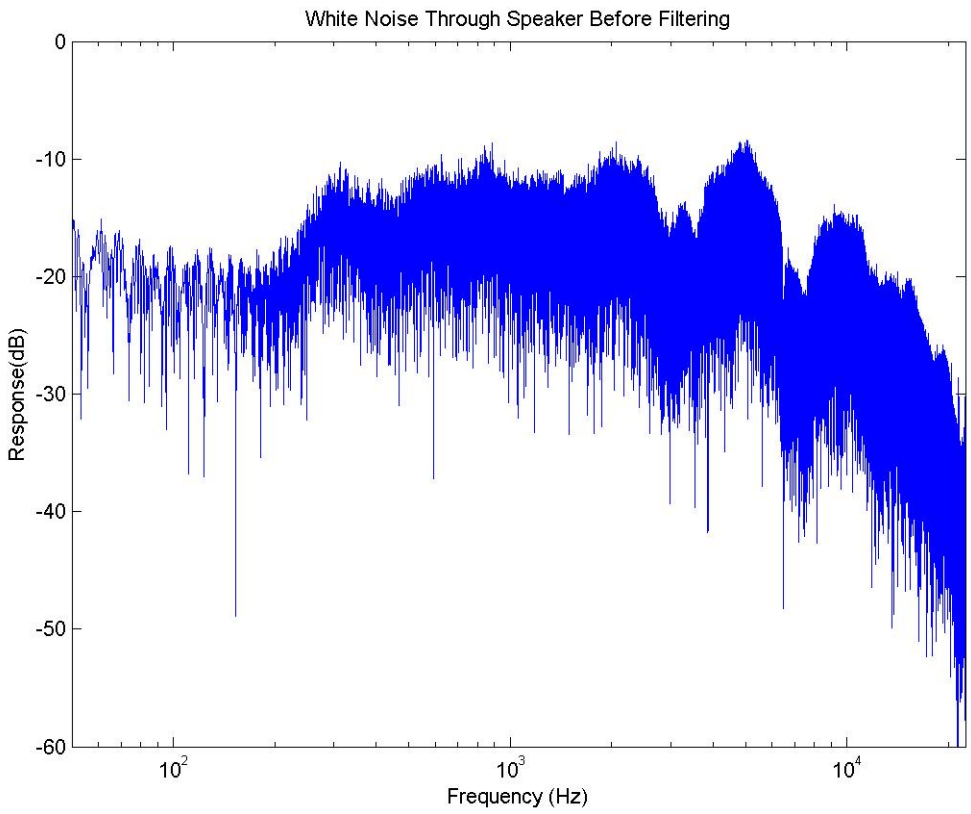

Figure 1.1

The resulting frequency response describes the speaker but is very noisy.

1.2 Sampling and Smoothing[2]

1.2.1 2. Sampling and Smoothing the Response.

Next sample the frequency response logarithmically so that low frequencies will be weighted more importantly by the filter. This allows the logarithmic plot of an ideal inverse filter and the inverse filter implemented by FIR coefficients to match (rather than matching only high frequencies). Next use MATLAB's built in Savitzky Golay [y = sgolayfilt(x,k,f)] filter to smooth the curve so as to reduce the noise of the response. The motivation behind this was is to prevent the fir2 algorithm from trying to create an inverse filter with the inverse of the noise of the frequency response.

[2]This content is available online at <http://cnx.org/content/m36587/1.1/>.

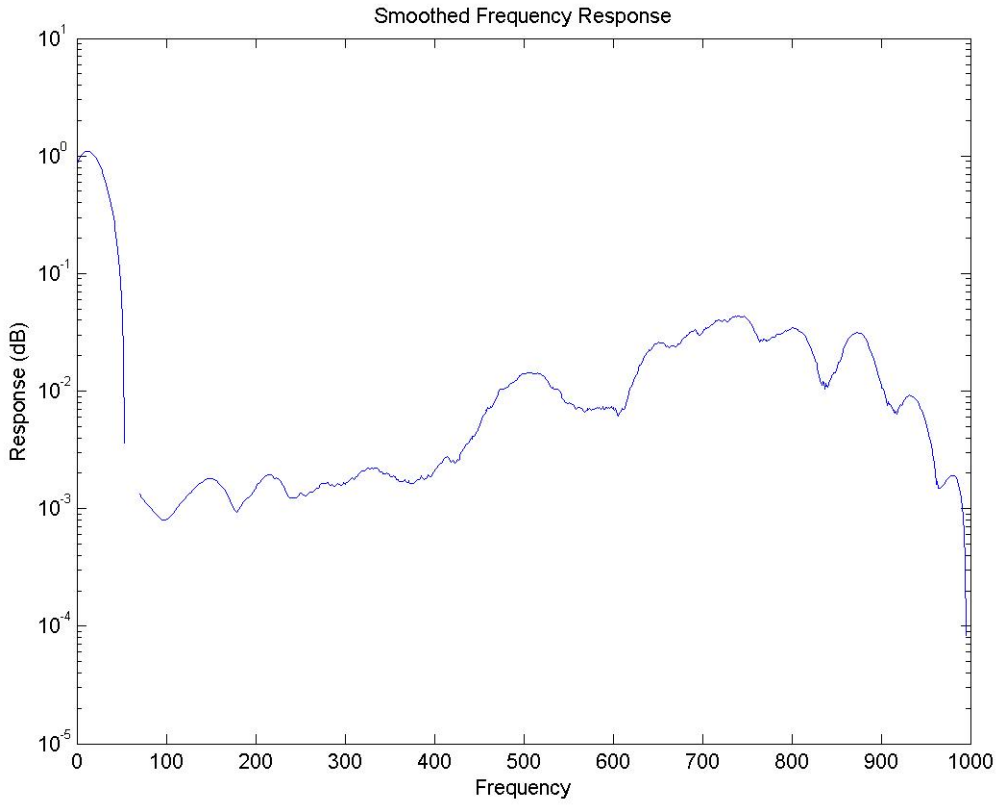

Figure 1.2

```
sampfft=thisfft(round(10.^(linspace(0,log10(length(thisfft)),1000))));  %sample fft logrithmicly at 1000 points
smoothfft=sgolayfilt(sampfft, 3, 71);
smoothfft(smoothfft<0)=0; %fix smoothing errors
semilogy(smoothfft)
title('Smoothed Frequency Response');
xlabel('Frequency');
ylabel('Response (dB)');
```

1.3 Low Frequency Resonance[3]

1.3.1 3. Low Frequency Resonance:

The resonant frequency of a loudspeaker is the frequency at which it is most susceptible to an electric response and any departure from this frequency causes the response to drop sharply. Because it is difficult to

[3]This content is available online at <http://cnx.org/content/m36594/1.1/>.

flatten such large peaks and valleys located next to one another, it's necessary to highpass filter the input to remove all frequencies at and below the resonant frequency. In the exemplar speaker, the resonant frequency zone was identified by a small bump followed by sharp drop in the frequency response at ∼500 Hz.

1.4 Inverting the Response[4]

1.4.1 4. Inverting the Response

After smoothing and logarithmically sampling the frequency response of a speaker one only has to invert this data to create an ideal inverse filter for the speaker. Simply flip the frequency response of the loudspeaker over the frequency axis and scale it appropriately.

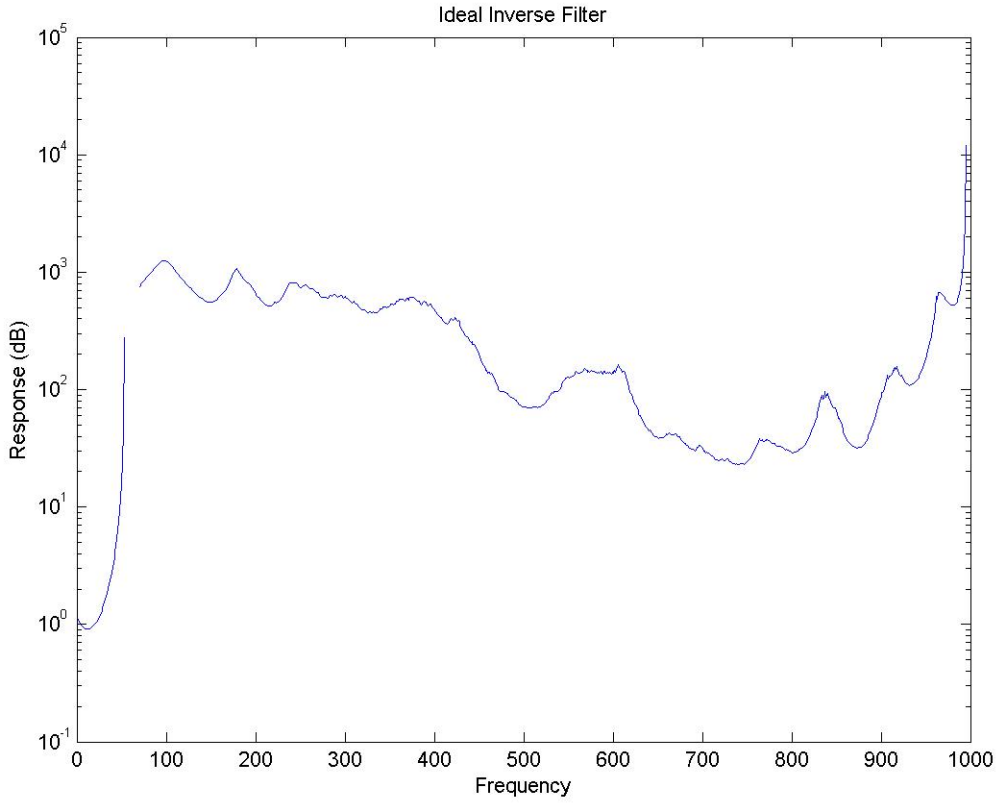

Figure 1.3

```
targetfft=(1./smoothfft);
semilogy(targetfft)
title('Ideal Inverse Filter');
xlabel('Frequency');
ylabel('Response (dB)');
```

[4]This content is available online at <http://cnx.org/content/m36551/1.1/>.

1.5 FIR Filter on TI Chip[5]

1.5.1 5. FIR Filter on the TI Chip.

To create a FIR filter which matches a desired inverse filter, use the built in MATLAB filter design program called fir2 [b = fir2(n,f,m)]. This designs an n order FIR filter which attempts to create an FIR filter whose spectrum matches a linear interpolation between input amplitudes m at frequencies f. The coefficients (b) are obtained by applying an inverse Fourier transform to m at frequencies f and multiplying by a window. In creating an inverse filter, the default Hamming window is acceptable because it provides a balance between the dynamic range and the resolution of the signal. The response of the inverse filter at different frequencies should not be so different that a huge dynamic range is required. Finally enter these coefficients into the TI chip and produce an FIR filter.

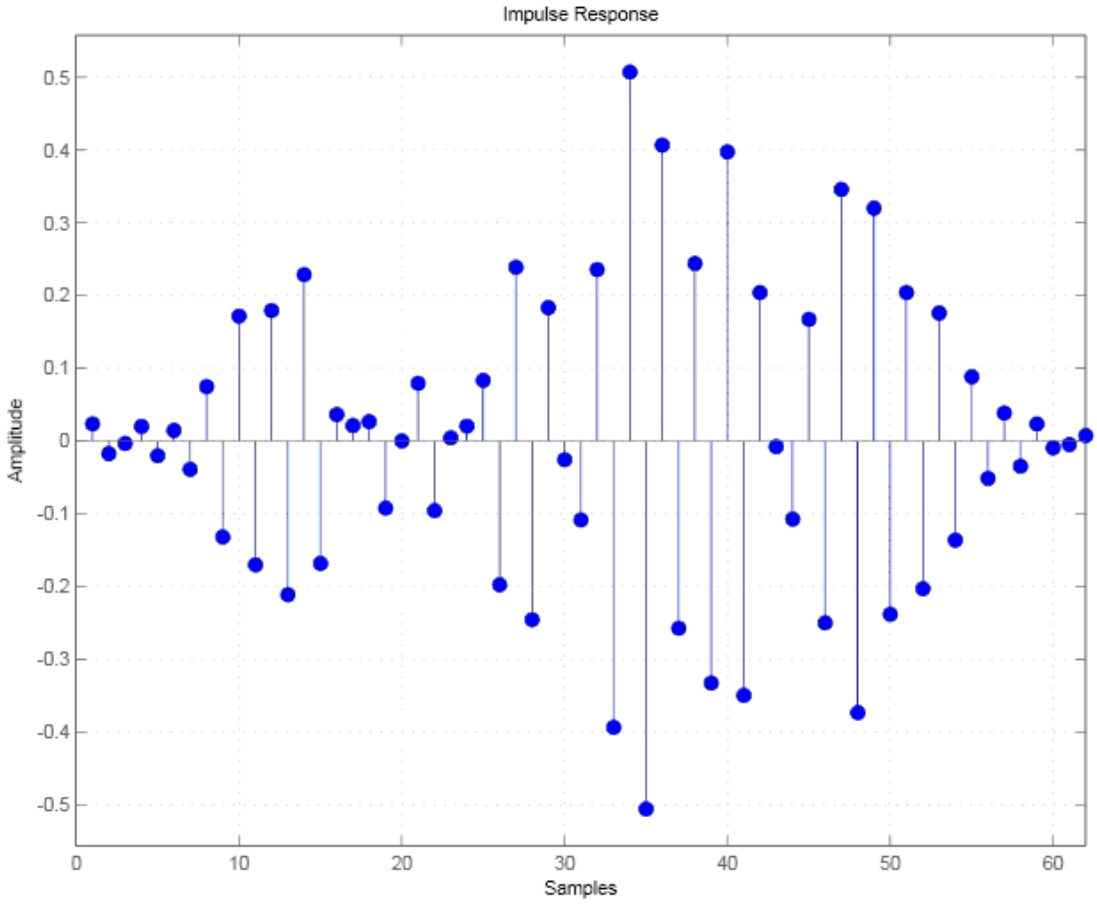

Figure 1.4

h=fir2(62, ((10.^(linspace(0,1,length(targetfft))'))-1)/9,targetfft);

[5]This content is available online at <http://cnx.org/content/m36463/1.1/>.

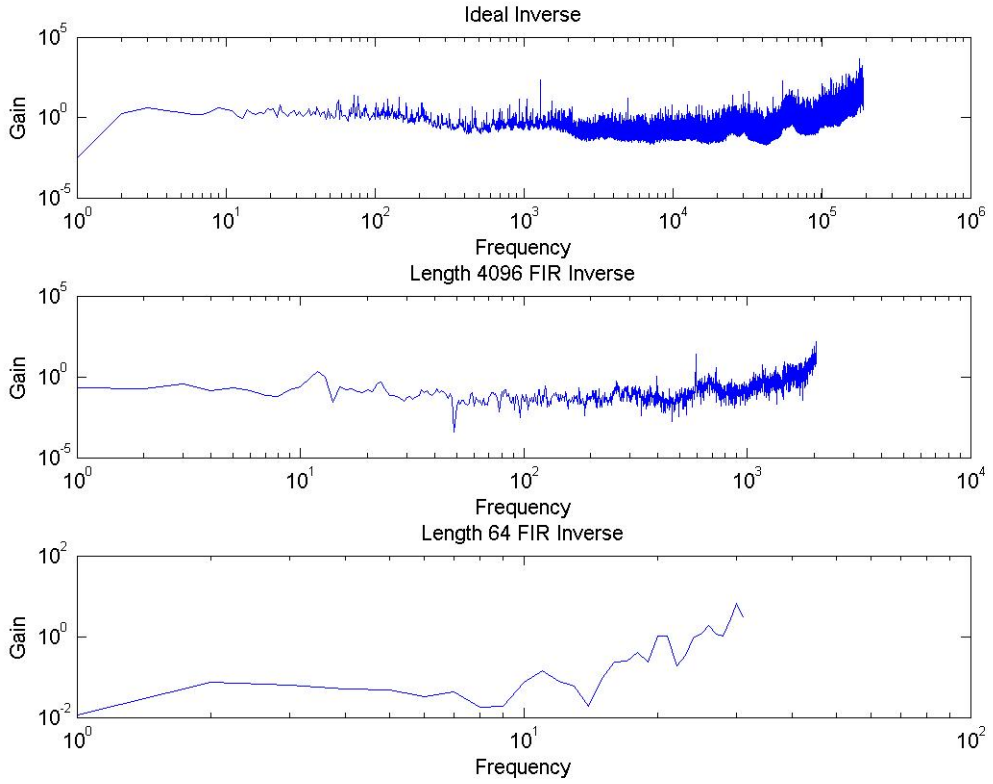

Figure 1.5

The FIR2 command with 64 coefficients does an acceptable job of matching the shape of an ideal inverse filter. However, given more memory and more coefficients one could do any even better job of matching the ideal.

1.6 Frequency Response Correction Verification[6]

1.6.1 6. Verification

Finally to verify that the inverse filter works, play white noise through the speaker again and record the response with and without the filter.

[6] This content is available online at <http://cnx.org/content/m36539/1.1/>.

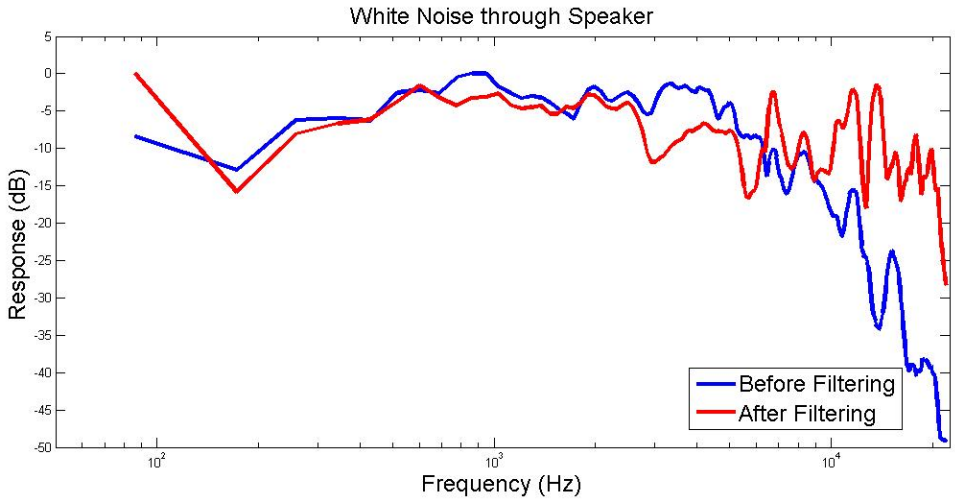

Figure 1.6

The filter isn't able to completely flatten the frequency response but it does smooth it significantly, especially at higher frequencies.

Chapter 2

Loud Speaker Equalization Project Conclusions[1]

2.1 Conclusion: Get a Better Mic

Using white noise and MATLAB's FIR design tools successfully implements an inverse filter on TI's 3245 EVM which flattens the systems frequency response. Unfortunately, when music was played through the exemplar speaker, various high frequencies would be over-amplified, creating hissing. Throughout the design of its inverse filter a relatively low end microphone, designed primarily to pick up speech signals, was used. After comparing what the frequency response of our system looks like compared to how it actually sounds, it's clear it was the microphone itself that was causing the attenuation of higher frequencies. Thus by amplifying higher frequencies, the inverse filter flattened the frequency response of the system but spoilt the frequency response of the speaker.

[1] This content is available online at <http://cnx.org/content/m36562/1.1/>.

Attributions

Loud Speaker Equalization Project
Elec 301 Project. Equalizing the frequency response of a speaker using an inverse filter.

About Connexions
Since 1999, Connexions has been pioneering a global system where anyone can create course materials and make them fully accessible and easily reusable free of charge. We are a Web-based authoring, teaching and learning environment open to anyone interested in education, including students, teachers, professors and lifelong learners. We connect ideas and facilitate educational communities.

Connexions's modular, interactive courses are in use worldwide by universities, community colleges, K-12 schools, distance learners, and lifelong learners. Connexions materials are in many languages, including English, Spanish, Chinese, Japanese, Italian, Vietnamese, French, Portuguese, and Thai. Connexions is part of an exciting new information distribution system that allows for **Print on Demand Books**. Connexions has partnered with innovative on-demand publisher QOOP to accelerate the delivery of printed course materials and textbooks into classrooms worldwide at lower prices than traditional academic publishers.