An Equalized Comparison

Michael Chemistruck

***Index Terms*—Equalizers, FIR, IIR, Phase Distortion**

I. Introduction

THERE are two main types of Linear Time Invariant (LTI) digital filtering algorithms – Finite Impulse Response (FIR) and Infinite Impulse Response (IIR)[1]. These two algorithms have long been studied in terms of the mathematics behind them – which filter is more memory efficient? Which filter has the least phase distortion? However, very little emphasis is placed on how the filters actually sound. This paper aims to give a brief introduction of the different filtering techniques and analyze the audible effects of each.

II. A Brief History Of Filtering

Until the advent of personal computers, analog filtering was the only choice for audio engineers looking to modify the frequency content of their tracks. Recording studios would pay top dollar for a well-designed hardware EQ. It was not uncommon to see racks full of different EQs, all with slightly different tonal characteristics. Any slight difference in components could completely change the output of the hardware. Even temperature changes would affect the operation of transistors and thus the effects on the sound [7].

The digitization of audio freed audio engineers from the inconsistencies of analog. Filters could now be implemented with a few lines of code – their effects never changing. Gone were the worries of perfectly matching resistors and making sure to mount transistors on heat sinks properly. Digital audio brought with it the reliability of computers to always perform the same code in the same manner.

Through the use of computer programming, many digital filters could now be implemented in a rather short period of time without the headache of analog implementation. With solid design equations, a skilled programmer could implement filters quickly and hear the result the same day without ever touching a circuit board.

Digital filters come with their own problems, though. All LTI filters involve adding a delayed version of the original audio signal back with itself. Introducing delay into an audio system can be a complete nightmare in live situations. Sometimes this delay is hardly noticeable, but higher order filters require more delay – thus increasing latency to the point of being unacceptable [4].

Phase distortion is another consideration. Some filtering techniques introduce phase irregularities as a result of delaying both input and output samples, then summing them back together. As with comb filtering caused by bad microphone placement in a recording context (and the phase mismatch due to the distance between close mics and room mics), digital phase-shifts can cause unwanted filtering and ruin the response of an otherwise good filter.

III. Mathematical Analysis

Generally, filter analysis is done through the transfer function. A filter's transfer function describes exactly how that filter will behave for all frequencies. This implies that transfer functions are in the frequency-domain. In digital filters, it is often very useful to analyze filters in the time domain as well. Since digital filters rely on delaying input and output samples, knowing which samples to delay and by how much will make implementation very easy.

Time domain analysis on a digital filter is done via difference equations. The difference equation describes the output in terms of the input as well as any delayed samples. Since difference equations and transfer functions depend heavily on which type of filter is being implemented, examples will be provided for each case as needed.

IV. Finite Impulse Response

FIR filters are purely feed-forward filters [1]. That is to say, they operate only with delayed input samples. The transfer function can be defined as follows:

$$H(z) = \sum_{k=0}^{n-1} h_k z^{-k} \tag{1}$$

Where n is the order of the filter. This equation demonstrates that FIR filters are comprised solely of zeros. In the time domain, FIR filters are represented as:

$$y(t) = \sum_{n=0}^{N-1} h_n x_{t-n} \tag{2}$$

Where N is the order of the filter. Because FIR filters are feed-forward, they are always stable (their output will never approach infinity). Only one buffer is needed in memory, but this buffer can grow quite large. The main advantage of Finite Impulse Response filters is that they have linear phase characteristics. This fact allows the use of FIR filters as a good base case.

Manuscript received December 2, 2010.

M. Chemistruck is a student at the University of Miami, Coral Gables, FL 33146 USA. He is an undergraduate studying Music Engineering Technology (e-mail: m.chemistruck@umiami.edu).

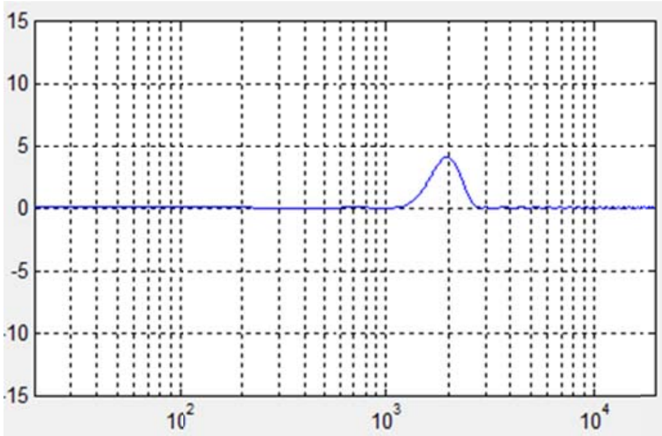

Figure 1: FIR Response

Although FIR filters generally require more coefficients [2] and thus, more delay, filtering is being performed out of real time – eliminating any latency considerations for the purposes of this study.

V. Infinite Impulse Response

IIR filters are feed-back filters [1]. They sum both delayed input (optional) and output (required) samples with the current input sample. In its most general form, the transfer function is defined as:

$$H(z) = \frac{\sum_{k=0}^{n-1} b_k z^{-k}}{1 + \sum_{l=1}^{m-1} a_l z^{-k}} \tag{3}$$

Where b_k represents the zeros of the function, a_l represents the poles, n is the order of magnitude of zeros, and m is the order of magnitude of poles for the filter. Alternatively, in the time domain:

$$y(t) = \sum_{n=0}^{N-1} \sum_{m=1}^{M} a_n x_{t-n} - b_m y_{t-m} \tag{4}$$

Where N is the order of magnitude of zeros and M is the order of magnitude of poles. Note that the bounds of the second sigma start at $m=1$ because the value $m=0$ represents the current output sample, $y(t)$ [6].

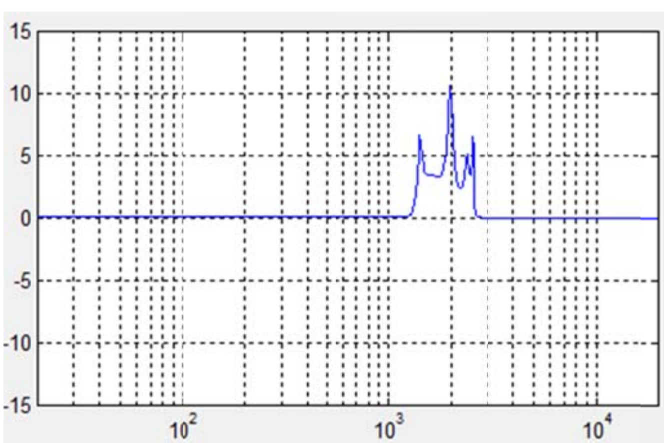

Figure 2: IIR Response

Because IIR filters rely on both poles and zeros, they are more complex to design but require significantly less computations [5]. Generally, IIR filters have very few coefficients, allowing for a small buffer size and fast implementation. Two buffers are required, though, for both input and output (as seen by the two sigmas in equations 3 and 4 above). It is important to note, however, that IIR filters will become unstable if one of the poles falls outside the unit circle. This will cause resonances that will likely distort the audio signal as they grow increasingly towards infinity. This instability is caused by feedback from the output – just as in analog filter design [3]. Also, IIR filters are notorious for having non-linear phase [6]. It is this last property that we will focus on in this study. How much can we hear the nonlinear phase of IIR filters when compared to FIR filters?

VI. Test Layout

While it would have been possible to conduct this study using command line operations in MATLAB, I felt that creating a graphical interface would make for a more interesting experience. This interface has sliders for center frequency, gain, and bandwidth of the filter, all changeable at the click of a mouse. At the core of the interface is a set of axes used to display the magnitude response of the filter as the user changes parameters. There are also controls to load wave files and play them with the designed filter.

When designing the filter, it is possible to switch between IIR and FIR algorithms by clicking the corresponding radio button on the top left of the interface. The actual filter design algorithms are described in section VII.

After picking suitable parameters for the filter, clicking the Play/Pause button will play the selected wave file with the filter applied. Processing is done before playback, so allow some calculation time for large files. When ready, the axes will display the filtered frequency response of the wave file. This is displayed by taking the 2048-point Fast Fourier Transform (FFT) of the filtered wave file and plotting the first 1024 points on a logarithmic scale.

VII. Filter Design Techniques

All computations are done in MATLAB. FIR filter coefficients are found using the function FIR2. This function takes as its parameters the desired filter order N and magnitude response H at which frequency bins W. It returns a vector b as the filter coefficients. A simple example of this function:

$$b = fir2(4, [0\ 0.25\ 0.5\ 0.75\ 1], [0\ 0\ 0\ 1\ 1]);$$
$$b = [0.0081\ -0.1550\ 0.3760\ -0.1550\ 0.0081]$$

The above example produces a high pass filter with the cutoff frequency at *0.75*Fn*, where *Fn* is the Nyquist frequency. In this study, I chose to design 100-order FIR filters.

The IIR design uses the YULEWALK function. Its parameters are the same as those for FIR2, but it returns 2 vectors *[b, a]* for the numerator and denominator coefficients, respectively. In this study, I chose to design 11-order IIR filters in both the numerator and denominator.

Both filters are applied to the audio signal using the FILTER function. The audio file is then output to the speakers after normalization to avoid clipping.

VIII. Subjective Comparison

Since the purpose of this study is to compare the audible effects of FIR and IIR filters, subjective testing was necessary. There was not sufficient time to conduct a true listening test with multiple non-biased subjects, but the graphical results helped keep me honest with my evaluations.

Overall, the phase distortion caused by the IIR filter was very noticeable. The phase distortion adversely affected the magnitude response of the filter – making it less reliable than the FIR filter. This is mathematically due to the process of finding the magnitude of any complex number:

$$|x + Ai| = \sqrt{x^2 + A^2} \tag{6}$$

Where x is the real part and Ai is the imaginary part. From this equation, it is obvious that even a small phase distortion can get amplified into a major change in magnitude response.

It is interesting to note that while the IIR filter sounded different than the FIR filter, it didn't sound bad. I often found that I liked the sound of the IIR filter better than the more predictable FIR. The FIR filter always has a nice, smooth curve to it. The IIR filter, however, tends to be irregular as the center frequency decreases. These irregularities still generally have the same effect, but also affect neighboring frequencies more so than the soft curve of the FIR.

In a few test cases, I was able to get the IIR filter to "blow up" and resonate infinitely. It was still having an effect as a filter, but there was also a resonant tone present in the signal. This tone, however, appeared as a pure, non-distorted tone because I applied scaling to the output signal prior to sending it to the speakers. It was interesting to experience both resonance and filtering at the same time.

IX. Conclusion

While FIR filters are more true to the parameters specified during the design process, IIR filters often sound better because of the inadvertent filtering caused by phase distortion. This is purely my opinion and has not been either verified or refuted as there was not enough time to conduct a proper listening test. I did prove that FIR and IIR filters do, in fact, sound quite different from each other and provided a tool to allow for an easy demonstration of those differences. Perhaps this tool could be expanded to include more filter types (cascaded biquads, elliptical filters, etc.). It is definitely a useful tool in demonstrating that the differences in the algorithm are both mathematical and audible.

Acknowledgment

M. Chemistruck would like to thank Professor Colby Leider for guiding him through his first steps on a lifelong journey through time and frequency. Without Prof. Leider's guidance, he surely would have gone astray.

References

[1] C. Leider, *Digital Audio Workstation*. New York, NY: McGraw-Hill Professional, 2004, pp. 123-127.

[2] F. Taylor, G. Zelniker, *Advanced Digital Signal Processing: Theory and Applications.* New York, NY: Marcel Dekker, Inc., 1994, pp. 165–219.

[3] D. Mellor, "Digital Mixing and Filtering", *Audio Engineer's Reference Book.* Great Britain: Elsevier Science Ltd., 1999, pp. 4-17.

[4] A. Mornington-West, "Principles of Digital Audio", *Audio and Hi-Fi Handbook*. Woburn, MA: Newnes, 2000, pp. 55-57.

[5] G. Louie, G.White *The Audio Dictionary*. 3rd Ed. Seattle, WA: University of Washington Press, 2005, pp. 193.

[6] R. Oshana, *Development Techniques for Embedded and Real-Time Systems*. Burlington, MA: Newnes, 2006, pp. 95-100.

[7] A. Delagrange. "Analog Design – Thought Process, Bag of Tricks, Trial and Error, or Dumb Luck?", *The Art and Science of Analog Circuit Design*. Woburn, MA: Newnes, 1998, pp. 339-342.