Teaching Digital Filter Design Techniques
Used in High-Fidelity Audio Applications

Venkatraman Atti, Andreas Spanias, Constantinos Panayiotou, Yu Song
E-mail: [atti, spanias, costasp, yu.song] @asu.edu
Department of Electrical Engineering, MIDL Lab
Arizona State University, Tempe, AZ 85287-5706, USA

Abstract
This paper presents web-based computer laboratory experiments and related assessment results for digital filter design modules that have recently been integrated into the ASU’s J-DSP tool. Filter design experiments (included in EEE407 DSP class) based on windowing, frequency-sampling, the Kaiser-design, Min-Max design, and IIR analog filter approximations have been discussed in the context of perceptual and lossless audio coding. We complement these filter design techniques and the laboratory experiments with some high-fidelity audio applications involving, reverberation, echo generator, bass/treble control, and shelving/peaking digital filters. On-line evaluation forms to assess student learning experiences have been carefully designed in the form of an XML database.

1. Introduction
The launch of high-end storage formats like the DVD-Audio and the Super Audio CD (SACD) in the consumer market, and the advancements in the multimedia processors and the storage capacity eventually spurred the need for high-fidelity audio coding schemes. Lossy (perceptual) and lossless audio coding techniques form the two key areas of audio coding research today. In lossy audio coding (LAC) schemes, the encoder discards the information that is not perceptually important, while this is not so in the case of lossless audio coding (L^2 AC) methods, where the audio data is merely ‘packed’ to obtain a bit-for-
bit representation of the original. Moreover, $L^2$AC schemes obtain compression by eliminating the statistical dependencies associated with the signal via FIR/IIR-based prediction techniques.

This paper addresses these FIR/IIR prediction techniques along with brief descriptions on digital filters employed in simulating audio effects such as echo, reverberation, and bass/treble controls. A great deal of interest among the student community to learn the aspects associated with digital audio coders, MP3, DTS, and Dolby Digital, essentially motivated us to develop these filter design modules and demos. These modules are incorporated as part of the J-DSP\(^1\) tool. In particular, at Arizona State University (ASU), as part of an effort to introduce undergraduates in DSP to application-oriented content, we are developing a series of on-line modules that include Java software, animated demonstrations, computer exercises, and video streamed lectures. In this paper, we describe the web-based educational software and the associated on-line labs tailored to expose students in a multimedia course to the basics of digital audio coding and audio effects.

J-DSP’s role, significance, and contribution to the teaching process have been presented in some detail earlier in [1]-[12]; and a summary of the same follows. J-DSP is an educational tool that was developed to enable on-line simulations and web-based computer laboratories for the DSP-related courses in Electrical Engineering. The J-DSP Version 1 (CD-ROM ISBN 0-9724984-0-0) is approximately 42,000 lines of Java code and is accompanied by a series of J-DSP laboratory exercises and a manual posted on the supporting web site. The J-DSP editor includes a suite of built-in signal processing functions ranging from simple signal manipulators to complex filter design functions as well as speech processing algorithms. We note that the NSF-funded J-DSP was built from the ground up to deliver web-based laboratory experiences to undergraduate students and therefore is different from platform-specific commercial simulation packages such as MATLAB/SIMULINK.

The rest of the paper is organized as follows. Section-II reviews the filter design modules developed in J-DSP. Digital filters in audio effects simulations are discussed in Section-III. Section-IV deals with

\(^{1}\) Work funded in part by the NSF-CCLI program (NSF-DUE-CCLI-0089075). J-DSP (http://jdsp.asu.edu) has been rated in the top three educational resources for 2003 by the NEEDS panel and recognized as such at IEEE FIE 2003.
the QMF bank simulation. Finite and infinite impulse response (FIR and IIR) filters in high-fidelity audio applications are dealt in detail in Section-V. Section-VI presents the assessment results.

II. Filter Design Modules in J-DSP

J-DSP supports FIR filter design based on – windowing method, the Kaiser-design, the Parks-McClellan (Min-Max) algorithm, and frequency-sampling method (both recursive and least-squares realization). IIR filter design experiments based on IIR analog filter approximations (Butterworth, Chebychev, and elliptic filter realizations) can also be performed in J-DSP.

1) Windowing Method

The windowing FIR filter design method is a straightforward technique implemented in J-DSP by expanding the frequency response of an ideal filter in a Fourier series and then truncating and smoothing the response using a window. The user needs to supply the following information (Figure 2(a)): Window type: Hamming, Hanning, Blackman, Bartlett, rectangular, or Kaiser; Filter order (maximum is 64); Type:
low-pass, high-pass, pass-band, or stop-band. Cut-off frequencies \( f_c \), take values from 0 to 1, where \( f_c = 1 \) corresponds to half-the-sampling frequency.

2) **Kaiser Design and Min. Max. Methods**

The Kaiser design process involves calculating the Fourier series of the ideal filter and then multiplying it with a Kaiser window that best fits the filter specifications. Filter specifications are (Figure 2(b)): Filter type: can be low-pass, high-pass, stop-band or pass-band; \( W_{p1}, W_{s1} \) – pass-band and stop-band edge cut-off frequencies, respectively; \( W_{p2}, W_{s2} \) – second pass-band and stop-band edge cut-off frequencies, respectively (for pass-band filters); \( PB, SB \) – pass-band and stop-band tolerances in dB.

FIR filter design experiments based on the Parks-McClellan algorithm (Figure 2(c)) form an important laboratory exercise that expose students to the concepts of optimal filter design, linear phase filters, minimum phase design considerations, and stability issues of the designed filter.

3) **Frequency-sampling Method**

The Frequency-sampling (Figure 2(e)) functionality in J-DSP enables digital filter design with three possible realizations, namely, non-recursive (non-R), recursive (R) and non-recursive using the least-squares method (non-R/LS). The user specifies the number of lines and the number of samples used in the frequency-sampling method and draws the desired frequency response on a grid. Consecutive placement of points on the grid creates auxiliary lines automatically to assist the user to visualize the resulting frequency response.

4) **Butterworth, Chebychev, and Elliptic Filters**

IIR filter design (Figure 2(d)) is based on bilinear transformation. Butterworth, Chebyshev type I and II, and elliptic filters can be designed. First a continuous-time, low-pass prototype meeting the desired filter specifications is generated. This prototype is then converted to the specified filter type (pass-band, high-pass, or low-pass) through a frequency transformation. The bilinear transformation is then applied, to transform the continuous-time filter response to discrete-time. From the discrete-time filter transfer function, the coefficients are calculated that describe the filter that best fits the specified filter parameters.
Figure 2. Digital filter design modules in J-DSP based on (a) windowing method (b) Kaiser-design (c) Parks-McClellan algorithm (d) IIR analog approximation (e) Frequency-sampling method
III. Digital Filters in Audio Effects Simulations

J-DSP simulations relating filter design with some real-life applications – echo, reverberation, bass/treble control, and graphic equalizer – are discussed next. These filter-design techniques form a special class of digital filters in audio processing and high-fidelity systems. The audio effects simulations supported [12] in J-DSP include: DTMF, MIDI, echo, reverberation, graphic equalizer, and shelving/peaking digital filters. Descriptions on J-DSP simulations involving DTMF and MIDI tone generator have appeared earlier in [12].

1) Echo (Feed-forward Effect)

The echo effect is obtained by mixing the input signal with its delayed version (Figure 3(a)). The proportion of the delayed signal to the “clean” original signal determines how obvious the echo is, and the delay signifies the echo period. The feed-forward effect can be simulated using a first-order FIR filter, whose difference equation is given by,

\[ y(n) = x(n) + b x(n - R) \]  

(1)

where \( x(n) \) and \( y(n) \) are the input and output signals, respectively, \( R \) is the echo delay, and \( b \) is the attenuation constant (\(|b|<1\)).

2) Reverberation (Feedback Effect)

Reverberation, on the other hand, is obtained by mixing the input signal with the delayed versions of its feedback. The effect of the feedback results in multiple echoes (Figure 3(b)). This is given by,

\[ y(n) = x(n) + b y(n - R) \]  

(2)

3) Bass and Treble Control

Low-pass and high-pass shelving filters are used to implement bass and treble controls of the type used in high-fidelity stereo systems, respectively. In particular, low frequencies are affected by bass adjustments with the input signal being processed through low-pass shelving filters, while, high frequencies are affected by treble adjustments through high-pass shelving filters. An expression for the low-pass shelving filter transfer function implemented in J-DSP is given by,
$$H_{\mu_k}(z) = \frac{(1 + k_{\mu})}{(1 + k)} \left( \frac{z - \left(1 - \frac{k_{\mu}}{1 + k}\right)}{z - \left(1 - \frac{k}{1 + k}\right)} \right)$$

where

$$k = \left(\frac{4}{1 + \mu}\right) \tan\left(\frac{\Omega}{2}\right)$$

and

$$\mu = 10^{\frac{f_{\Omega}}{20}}; \quad \Omega = 2\pi f_s / f_c$$

4) **Peaking Filter Design**

The peaking filter design allows users to attenuate audio signal components outside a specified frequency range. In particular, the peaking filter design introduces users to the concept of a band-pass filter.

$$H_{\mu_k}(z) = \left(\frac{1 + k_{\mu}}{1 + k}\right) \left( \frac{(z - z_1)(z - z_2)}{(p_1 - p)(p_2 - p)} \right)$$

where

$$z_1 = \frac{\cos(\Omega) + \sqrt{\cos^2(\Omega) - k_{\mu}^2\mu^2}}{1 + k_{\mu}}, \quad z_2 = \frac{\cos(\Omega) - \sqrt{\cos^2(\Omega) - k_{\mu}^2\mu^2}}{1 + k_{\mu}}$$

$$p_1 = \frac{\cos(\Omega) + \sqrt{\cos^2(\Omega) - k_{\mu}^2}}{1 + k_{\mu}}, \quad p_2 = \frac{\cos(\Omega) - \sqrt{\cos^2(\Omega) - k_{\mu}^2}}{1 + k_{\mu}}$$

$$k_{\mu} = \left(\frac{4}{1 + \mu}\right) \tan\left(\frac{\Omega}{2}\right)$$

and

$$\mu = 10^{\frac{f_{\Omega}}{20}}$$

Figure 3. Filter design for audio effects simulations (a) echo (b) reverberation
5) **Graphic Equalizer (GE)**

The graphic equalizer (Figure 4) functionality in J-DSP consists of five cascaded peaking filters that divide the frequency spectrum into five frequency bands. In particular, the graphic equalizer block alters the frequency response of each band independently using the available slide bars (Figure 4). The peaking filters maintain constant center frequencies (170-Hz, 310-Hz, 600-Hz, 1-kHz, 3-kHz respectively) but support variable gains. These gains can be altered using the slide-bars on the dialog window of this block, with each bar corresponding to a specific band. It is interesting to note that all the aforementioned 'audio effects' functions can support on-line labs for use in high-school and freshman engineering classes.

![Figure 4. Graphic equalizer simulation in J-DSP](image)

**IV. Filterbank Design in Perceptual Audio Coding**

The choice of an appropriate filterbank is critical to the success of a perceptual audio coder. Therefore, it is important that the students have a thorough knowledge of filterbank design considerations as employed in a typical audio coding application. To this end, work is being done in the development of J-DSP
modules to perform QMF and filterbank simulations. In addition, graphical-modules that expose students to the analysis/synthesis filterbank structures and subband filtering concepts have also been developed.

1) **QMF bank simulation**

Figure 5 shows the two-channel QMF simulation in J-DSP. Some audio coding applications require dividing the processed signals into subbands. This gives the advantage that bands can be processed separately taking into consideration each band’s properties. Dividing signals into sub-bands introduces aliasing noise to the signal due to the overlapping bands. This aliasing cancellation is achieved using QMF banks. QMF banks consist of antialiasing filters, a down-sampling stage, an up-sampling stage, and an interpolation filter (Figure 5).

![Figure 5. Two-channel quadrature mirror filterbank simulation](image)

**V. FIR and IIR Filter Design in Lossless Audio Coding**

In our J-DSP laboratory experiments, we consider two L²AC algorithms to demonstrate the principles of lossless audio coding, i.e., the SHORTEN (FIR-based) and the DVD algorithm (IIR-based). SHORTEN algorithm, proposed by Robinson [13], employs a tenth-order FIR filter to predict the residual error. Craven et al. developed a scheme called the *DVD standard* [14] that uses a third-order IIR predictor. An
insight to both FIR and IIR prediction techniques towards the lossless compression is given in [14], a classical paper by Craven and Gerzon.

![Diagram of SHORTEN algorithm in J-DSP](image)

Figure 6. A simple flowgram in J-DSP to simulate the SHORTEN algorithm.

1) **FIR-based Prediction**

Prediction and residual coding are the two primary stages associated with the SHORTEN algorithm [13]. The algorithm operates on 16-bit linear PCM signals sampled at 16 kHz. First, the audio signal is divided into blocks of 16ms corresponding to 256 samples. Next, a tenth-order FIR filter is used to perform the linear prediction (LP) analysis in order to compute the LP coefficients. These prediction coefficients along with the residual are quantized and are made available in the synthesis process. Figure 6 illustrates the SHORTEN algorithm simulation in J-DSP. Note that in the picture residual quantization/encoding, multiplexer modules are not shown.

2) **IIR-based Prediction**

The IIR-based prediction used in the DVD algorithm can be simulated in J-DSP as follows. First, the input audio is divided into frames of lengths that are integral multiple of 384. A third-order \((N_b = N_a = 3)\) IIR predictor with fine coefficient quantization is employed.
J-DSP labs that compare the performance of IIR and FIR prediction schemes are included. In particular, the following concepts are introduced through web-based experiments – a tenth-order FIR predictor fails to flatten the drop (~50dB above 20 kHz) in the input spectrum; a third-order IIR predictor is able to do this very efficiently; IIR prediction schemes provide superior performance over FIR prediction techniques for the cases where control of both average and peak data rates is equally important.

VI. Assessment Results

The assessment questions are directly related to the technical aspects of the J-DSP on-line laboratories. The concept-specific forms focus on each exercise by posing questions that determine whether the student has learned a specific DSP concept. For instance, 87% of the students agreed that the filter design exercise helped them understand which window is suitable for sharp transitions, 88% of the students understood better the signal symmetries in the FFT spectra because of J-DSP visualization, and 91% of the students reported that the Z-transform exercise helped them understand the relation between the pole-zero locations and the frequency response plots. More results are given in Table 1.

Table 1. Statistics based on the concept-specific assessment. Total Number of Students that contributed the data = 87 [From the spring and fall semesters of the year 2003]

<table>
<thead>
<tr>
<th>Evaluation Questions</th>
<th>Strongly Agree (%)</th>
<th>Agree (%)</th>
<th>Neutral (%)</th>
<th>Disagree (%)</th>
<th>Strongly Disagree (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. My understanding of the concepts of FIR and IIR filter design is enhanced by the J-DSP labs</td>
<td>42</td>
<td>47</td>
<td>8</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>2. I have learned how to generate a sinusoid with a digital filter</td>
<td>29</td>
<td>55</td>
<td>11</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>3. The relationship between the impulse response and the transfer function is clear</td>
<td>95</td>
<td>N/A</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. After performing the J-DSP lab it is clear that the FFT spectral resolution is limited by the FFT size, the window type, and the window size</td>
<td>99</td>
<td>N/A</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5. J-DSP labs enhanced my learning of the basic DSP concepts</td>
<td>92</td>
<td>N/A</td>
<td>8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Bibliography


