Audio Engineering Society

Convention Paper 8892

Presented at the 134th Convention
2013 May 4–7 Rome, Italy

Implementation of an intelligent equalization tool using Yule-Walker for music mixing and mastering

Zheng Ma[1], Joshua D. Reiss[1], and Dawn A.A. Black [1]

[1] Centre for Digital Music, Queen Mary University of London, Mile End Road, London, E1 4NS, UK

zheng.ma@eecs.qmul.ac.uk
josh.reiss@eecs.qmul.ac.uk
dawn.black@eecs.qmul.ac.uk

ABSTRACT

A new approach for automatically equalizing an audio signal towards a target frequency spectrum is presented. The algorithm is based on the Yule-Walker method and designs recursive IIR digital filters using Least-Squares fitting to any desired frequency response. The target equalization curve is obtained from the spectral distribution analysis of a large dataset of popular commercial recordings. A real-time C++ VST plug-in and an off-line Matlab implementation have been created. Straightforward objective evaluation is provided, where the output frequency spectra are compared against the target equalization curve and the ones produced by an alternative equalization method.

Index items – Time-varying equalization, real-time, spectral distribution, automatic mixing, and audio production.

1. INTRODUCTION

For as long as spectral analysis has been a viable tool in the commercial sectors, audio engineers have looked at integrated spectral responses as possible answers for audio quality. Bob Katz [1] suggests that the tonal balance of a symphony orchestra can be used as an ideal reference for the spectral distribution of music. Through a series of interviews and discussions with mixing and mastering engineers, [2] found that many mixing engineers mix towards a subconscious target frequency response curve. In [3], we analysed the long-term averaged spectral distributions of a large dataset of popular commercial recordings and results showed there was a consistent leaning towards a target equalization curve that stems from practices in the music industry as well as from the natural, acoustic spectra of ensembles.

Applying a parametric or graphic equalizer (EQ) to approach a target spectral distribution is a non-trivial task. Expert knowledge is required to understand which parameter change results in a desired effect. In this paper, we proposed a new method for automatically equalizing the audio signal towards a target spectral distribution. The motivation behind this idea, as with the majority of the work by the authors in the field of intelligent systems for mixing multichannel audio [4], is to exploit best practices in sound engineering to appropriately modify the signals.

Research on intelligent audio equalization has been explored in two directions. The proposed methods in [5-10] are all based on off-line machine learning approaches, where humans need to manually train the system first in order map the user's preference into the parameter setting. In [11], a different, cross-adaptive method was proposed based on the extraction and analysis of the perceptual features to achieve equal average loudness on all frequency bands. However, most methods mentioned were restricted to a fixed number of frequency bands.

One of the prime challenges behind this equalization algorithm is to design an efficient and stable filter with an arbitrary magnitude response. Finite impulse response (FIR) filter design based on the Least-Squares method provides a quick solution [12]–[17]. But there are caveats with FFT convolution methods to do with loss of precision, computational complexity, quantization, dither and the effects of the inevitable FIR windowing when filtering the input signal with FIRs. IIRs can avoid many of these disadvantages. But an IIR filter is a complex feedback network. There is a dearth of good methods to design these once moving_away from classic filter transfer functions. [18] described a method of fitting infinite impulse response (IIR) filters to an arbitrary frequency response using Singular Value Decomposition (SVD). However, evaluation showed that this method also lacked accuracy in the lower frequencies and not suitable for low-pass, high-pass and band-pass filters with classic response shapes. The Yule-Walker method of Autoregressive Moving Average (ARMA) spectral estimation [17] was found to provide a better spectral accuracy, where the computational cost was reasonable.

A few commercial plug-ins are capable of matching the spectrum of one piece of audio to another, such as Logic Pro's Match EQ, iZotope's Ozone, DUY's MagicSpectrum. However, none of them are truly real-time. A learning process of the spectral content of both input and source file is needed before actual filtering. In most cases, a single EQ curve (time-constant) is calculated and applied to the whole signal using either FIR filters or parametric filters to fulfil the roles. In addition, although there are some target spectra defined in terms of genre in Logic's Match EQ, no academic or statistical study has been conducted to validate and evaluate the target curves.

In this paper, a new real-time approach to match the magnitude spectrum of the input audio to a target curve was presented. The algorithm is based on Yule-Walker IIR filter design using a recursive Least-Squares fitting to a specified frequency response. The proposed approach is able to apply a time-varying EQ curve of arbitrary magnitude shape to the audio signal. An off-line Matlab implementation and a real-time VST plug-in, adapting a similar implementation architecture for intelligent audio effect plugins that was proposed in [19] [20], were implemented.

2. TARGET EQUALIZATION CURVE

The target equalization curve used in this paper was extracted from previous spectral analysis on a large dataset of nearly 800 uncompressed songs that have been number ones in either the US or the UK charts over the last 60 years. The result of averaging the spectra of all songs in the dataset is shown in Figure 1. The trend seen in the average spectrum is consistent with what can be observed for individual distributions and the 95% confidence intervals are so narrow that

they are not perceptible on the shown scale. It seems that spectra of professionally produced commercial recordings show consistent trends, which can roughly be described as a linearly decaying distribution of around 5 dB per octave between 100 and 4000 Hz, becoming gradually steeper with higher frequencies, and a severe low-cut around 60 Hz.

The averaged spectrum could be used as a frequency balance reference for commercial recording mixing as best practices. In this paper, we use a smoothed version of the average spectrum (see Figure 2) as the target equalization curve. A 17-point moving average filter is applied using the equation:

$$y'(n) = \frac{y(n-8)+y(n-7)+\cdots+y(n)+y(n+1)+\cdots+y(n+8)}{17} \quad (1)$$

We only apply the smoothing mechanism on the frequency range from 200 Hz onward, so that the peak frequency and peak magnitude on lower frequency are preserved, while higher frequency bin values are smoothed to filter out the raggedness (comb-like shape) of the mid-distribution.

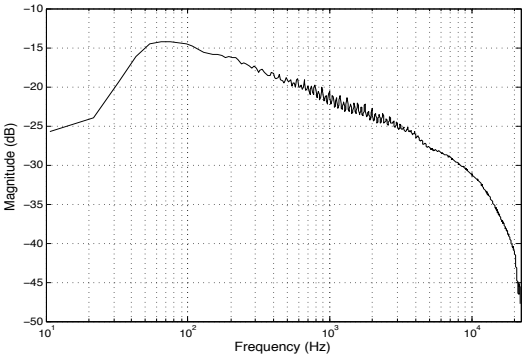

Figure 1. The average spectrum of the whole commercial recording dataset

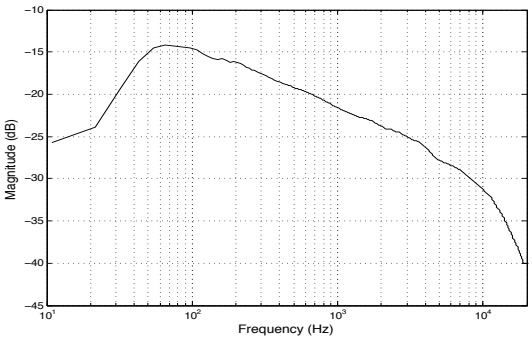

Figure 2. Smoothed version of the average spectrum as the target equalization curve

3. ALGORITHMS AND IMPLEMENTATION

After the target equalization curve has been found, the next stage is to design an algorithm to filter the input audio signal so that its spectrum matches the target. The overall block diagram of the full system is shown in Figure 3. In summary, the filtering process is first controlled by a noise gate, which determines from a frame's energy whether it can be considered to be active. Only active frames enter the filter design stage. For inactive frames, the filter curve is kept stationary. The spectrum of the active frame is then analysed and matched against the target, creating a filter curve using the Yule-Walker method. Filter curves are smoothed within and between frames to minimize the artifacts. Details about each procedure is described in the later sections.

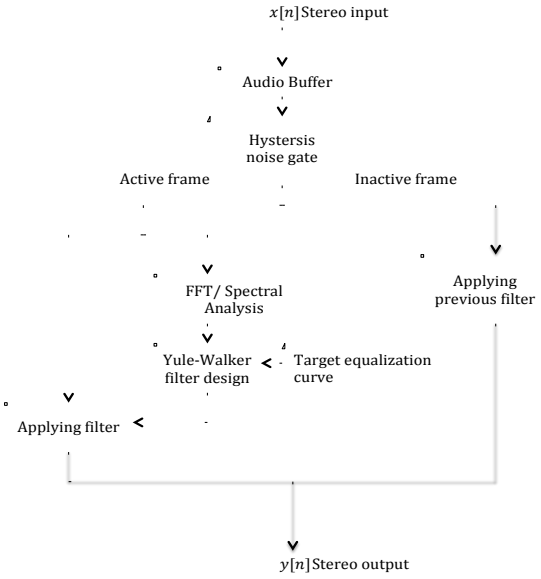

Figure 3. System block diagram

Software implementations of the algorithm have been developed in both Matlab and C++. Both operate on a frame-by-frame basis, but the C++ implementation uses a sample-based approach to realize real-time, low latency processing for practical use. The C++ version deploys a host/plug-in structure, where the host defines the frame size. Buffering of audio data where necessary occurs within the plug-in. The Matlab implementation is mainly used for exploration of the algorithm and performing visualization and evaluation.

3.1. Hysteresis noise gate

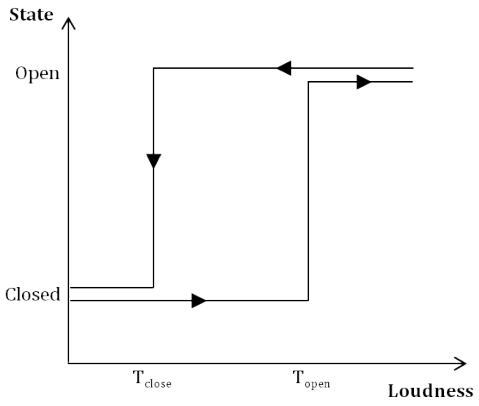

Figure 4. Noise gate hysteresis operation

We adapt the noise gate with hysteresis algorithm in [19] to classify the input frames to be either active or inactive based on their loudness estimated by the R-128 loudness measure [22]. We assume that a frame must be active to contribute to the next stage, where the filter curve is created and applied to match the target equalization curve. If inactive, the same filter curve that was applied on the previous active frame will be applied. Hysteresis thresholds at -25 and -30 LUFS are used to help prevent excessive switching of states. After crossing one threshold the gate maintains its current state until the loudness level moves below or above the other threshold. This is a concept employed in the Schmitt trigger [8] and is portrayed in Figure 4.

3.2. Spectral analysis

A 4096-point sliding FFT with Hanning windows was performed. Since the host itself defines the frame size and it's usually less than 4096 samples, we need to create a buffer of 4096 samples to store enough samples. The FFT was only performed on the active frames in order to cut down the computational cost and achieve a low latency.

Since the calculated magnitude spectrum will act as a denominator in a later analysis stage to obtain the desired transfer function (filter curve), a problem could arise if the amplitude values at one or more frequency components are too small. As a result, we end up with a transfer function with unreasonable peaks, which is difficult to estimate and produces unpleasant sound artifacts. It may also make the IIR filter highly unstable. To avoid this, a simple threshold technique is applied. We opt to use a threshold of 0.0001 to filter the magnitude spectrum. Values less than 0.0001 are usually found only at very high frequencies (the target equalization curve for example, see Figure 1). So the thresholding mechanism will have an insignificant effect on the accuracy of the filter design. Later, we normalized the spectra by dividing the magnitudes by the maximum magnitude for spectrum comparison.

3.3. IIR filter design

The design of an IIR filter with arbitrary magnitude response using the Yule-Walker Least-Squares fitting approach is described in this section.

3.3.1. Obtain desired magnitude response

Let $|X(\omega)|$ denote the thresholded, normalized magnitude spectrum of the active frame, and $|T(\omega)|$ denote the target equalization curve. Therefore, the desired transfer function $|H_d(\omega)$ can be simply obtained from the equation:

$$|H_d(\omega)| = \frac{|T(\omega)|}{|X(\omega)|}, \qquad \omega \in [0, \pi) \quad (2)$$

The values of $|H_d(\omega)|$ are calculated at every 1/3 octave center frequency, namely: 16, 20, 25, 31.5, 40, 50, 63, 80, 100, 125, 160, 200, 250, 315, 400, 500, 630, 800, 1000, 1250, 1600, 2000, 2500, 3150, 4000, 5000, 6300, 8000, 10000, 12220, 16000, 20000 Hz.

1/3 octave frequency representation is strongly linked to the perception of sound by a human ear and it allows a compression of the amount of information. 33 frequency

bands are large enough to capture the transition of the impulse response with an arbitrary shape while the computational cost is reduced significantly compared with the one of using 2048 linear-spaced frequency points. Afterwards, we normalize $|H_d(\omega)|$ into the range (0,1) to prevent overshooting.

In the practical C++ implementation, the actual values of $|T(\omega)|$ are weighted values between the target equalization curve and magnitude spectrum of the processed frame defined as follow:

$$\left|T'^{(\omega)}\right| = |T(\omega)| * a + (1-a) * X(\omega), \quad a \in [0,1] \qquad (3)$$

The value of a is left as one of the user control parameters. Users can either increase the value of a to match the target curve more or decrease it to preserve the original spectral content more based on their personal listening evaluation.

3.3.2. Filter curve smoothing

As the algorithm produces time-varying filter curves operating on audio signals, variable smoothing on desired IIR filters' magnitude responses within a single frame and between adjacent frames is necessary to avoid sound artifacts.

Also since the intelligent EQ tool runs in real-time with sample-based and short-frame processing employed in the algorithm, an efficient and reliable long-term average measure is necessary throughout to produce useful and smoothly varying data variables. Exponential moving average (EMA) filters are used extensively to fulfill this role [19]. The EMA filter is a first order IIR filter described by the following difference equation:

$$Y[n] = \alpha \cdot Y[n-1] + (1-\alpha) \cdot X[n] \qquad (4)$$

Where $\alpha = e^{-1/(\tau f_s)}$. f_s is the sample rate and τ corresponds to the time that takes the system to reach 1-1/e of its final value i.e., $y[\tau f_s] = 1 - 1/e$. α determines the degree of filtering between adjacent samples: the higher the value the less the rate of decay.

EMA filters are first applied to the desired magnitude response $H_d(\omega)$ of the filter curve within one active frame as follows:

$$H_d{}'(\omega_n) = \alpha \cdot H_d{}'(\omega_{n-1}) + (1-\alpha) \cdot H_d(\omega_n) \qquad (5)$$

In this case, τ_1 is set to 0.5 ms based on empirical experiments.

EMA filters are also applied to smooth the overall variations of filtering curves between consecutive frames. In this case, the EMA equation becomes:

$$H_m{}'(\omega) = \alpha \cdot H_{m-1}{}'(\omega) + (1-\alpha) \cdot H_m(\omega) \qquad (6)$$

Where $H_m{}'(\omega)$ corresponds to the new value of $H_m(\omega)$ for current frame m; $H_{m-1}{}'(\omega)$ corresponds to the transfer function value for previous frame $(m-1)$. New filter curves are calculated once every frame. Let W denote the frame size which is decided by host itself, then the frequency of new filter values is:

$$f_w = \frac{f_s}{w} \qquad (7)$$

τ_2 here is set to 1.28 s for a typical frame size of 64 samples to prevent filter curves from changing wildly from frame to frame.

3.3.3. Obtain IIR filter coefficients using Yule-Walker

We adapt the Yule-Walker method to perform a Least-Squares fitting to the desired frequency response $H_d(\omega)$ to find a causal stable rational function:

$$H(z) = \frac{B(z)}{A(z)} \qquad (8)$$

that best approximates $|H_d(\omega)|$. The Yule-Walker method finds the N-th order recursive filter coefficients B and A such that the filter:

$$\frac{B(z)}{A(z)} = \frac{b(0) + b(1) \cdot z^{-1} + \cdots + b(n) \cdot z^{-n}}{1 + a(1) \cdot z^{-1} + \cdots + a(n) \cdot z^{-n}} \qquad (9)$$

Where $\{b(0),\dots,b(n)\},\{a(0),\dots,a(n)\}$ are the denominator and numerator coefficients of the desirable IIR filter, and $a(0)=1$. The denominator are computed by the so called "modified Yule Walker" equations, using Nitrate Reductase (NR) correlation coefficients computed by inverse Fourier transformation of the specified frequency response $H_d(\omega)$. The numerator is computed by a four-step procedure:

1. A numerator polynomial corresponding to an additive decomposition of the power frequency response is computed.

2. The complete frequency response corresponding to the numerator and denominator polynomials is evaluated.

3. A spectral factorization technique is used to obtain the impulse response of the filter.

4. The numerator polynomial is obtained by a Least-Squares fitting to this impulse response.

The Yule-Walker estimation method is mathematically complicated. For a detailed explanation of the algorithm see [17].

Based on a series of listening experiments, *N* is set to 16 to balance the trade-off between computational cost and filter performance. IIR filter of a slightly high order gives us a good approximation and does not cause any latency problems.

3.4. Filter applying

The final step is the actual filtering. We filter the audio samples with an IIR filter described by its denominator coefficients $B\{b(0),\dots,b(n)\}$and numerator coefficients $A\{a(0),\dots,a(n)\}$. The filtering process is implemented as a difference equation:

$$y[n]=b(1)\cdot x[n]+b(2)\cdot x[n-1]+\cdots+b(17)\cdot x[n-16]-a(2)\cdot y[n-1]-\cdots-a(17)\cdot y[n-16] \quad (10)$$

Where $x[n]$ is the current input audio sample, $y[n]$ is the current output. Since the deployed host/plug-in structure operates on frame-by-frame basis. Two audio buffers, one to store previous values of $x[n-1]$ to $x[n-16]$, another to store previous values of $y[n-1]$ to $y[n-16]$, are needed to realize the filtering process across consecutive frames.

3.5. Interface and user control

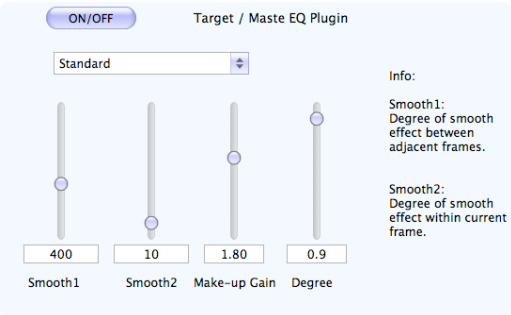

Figure 5. Screenshot of the plug-in interface.

In the practical C++ implementation of the algorithm, the parameters τ_1 and τ_2 that control the smoothing effect on the filter curves within and between frames are left for user control to correct unusual artifacts caused by improper time-varying filtering (mainly through adjusting τ_w') and to choose the degree of EQ effect of on the signal (mainly through adjusting τ_1). However, the optimal values of $\tau_1=0.5\,ms$ and $\tau_2'=1.28\,s$ (as explained before) are set as default. In addition, an on/off button and make-up gain to adjust the output volume as well as weighting parameter are also implemented. The screenshot of the plugin interface is shown in Figure 5.

4. RESULTS AND OBJECTIVE EVALUATION

A straightforward objective evaluation in the form of comparing the before-and-after magnitude spectrums of the signal to see whether the algorithms are able to achieve the objectives is performed using the Matlab implementation, followed by a quick comparison between our C++ implementation of Yule-Walk algorithm against an alternative target EQ implementation. The time-varying filtering algorithm behind this alternative target EQ is based on traditional fixed-band graphic equalizers.

First, we used a white noise signal as testing signal. The result is shown in Figure 6. The top line denotes the target equalization curve; the middle line denotes the long-term averaged spectrum of the signal after being processed by proposed master EQ, and the bottom line denotes the initial spectrum of the white noise signal.

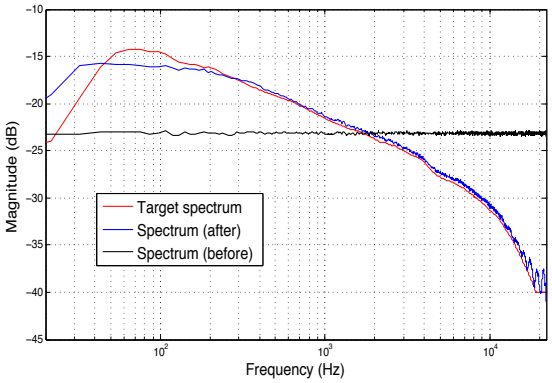

Figure 6. Testing result using white noise signal.

As shown in Figure 6, the initial spectrum of white noise is nearly constant. However after being processed by the intelligent EQ tool, its output spectrum lies close to the target curve, especially on the frequency range of 100 Hz afterward. The noticeable bias between output spectrum and target spectrum appearing on the low frequency range (20 Hz to 100 Hz) is possibly due to the fact that the filter design lacks defined spectral resolution on lower frequency range. But the difference at the low end is hardly perceptible to human ears.

Next, we tested on a real music signal which is uncompressed (CD quality) and has a typical 44.1 kHz sampling rate. The result is presented in Figure 7 as below.

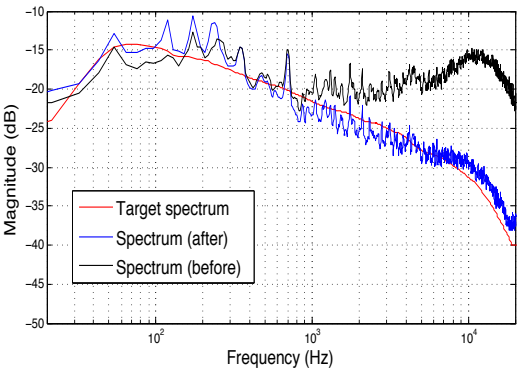

Figure 7. Test result using real music signal.

The tested music signal has a boost on the high frequency range starting from 2 kHz. After being processed by the plug-in, as Figure 7 shows, the output spectrum (denoted by the blue line) lies again close to the target equalization curve (denoted by the red line) and the high frequency content has been attenuated to match the target spectrum. The order N of the IIR filter is set to 16 and τ_1, τ_2 are set to optimal values for both tests.

Next, we attempted to evaluate the performance of the algorithm in terms of the order of IIR. The selected results are shown in Figure 8.

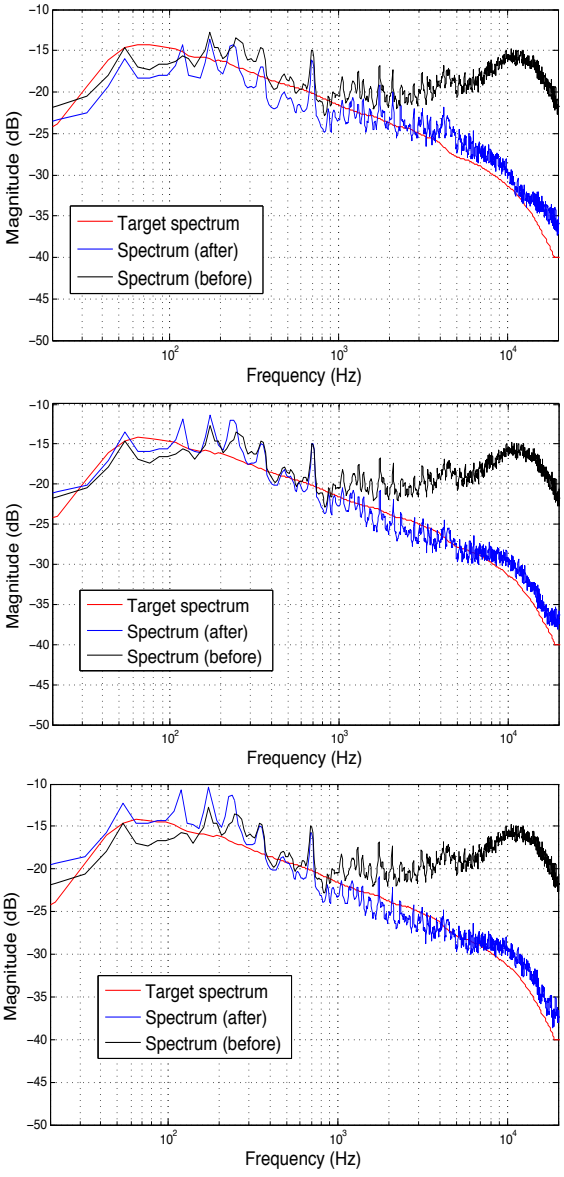

Figure 8. Results of IIR filter with 2-nd order (top), 8-th order (middle), 32-nd order (bottom)

The proposed method produces a better performance when using a higher order. The output magnitude spectrums of using IIR order of 8-th, 16th and 32nd (see Figure7 and Figure 8 bottom) are quite close. In fact, the small improvement of using higher IIR order then 16 is hardly perceptible.

We also compared our practical C++ VST implementation of the Yule-Walker algorithm against an alternative target EQ implementation. First, the same white noise signal was fed into both plug-ins. All user control parameters are set to optimal values. The results are depicted in Figure 9.

The spectrum curve obtained from the alternative target EQ shows relatively sharp peaks around 250 Hz and 10 kHz with an up-climbing slope at the high end possibly owing to the fact that it uses fixed frequency bands equalization method. The spectrum curve produced by our target EQ sustains a flat response at middle range and constant exponential decrease at both low and high end. Regarding spectrum matching toward a specific target, the Yule-Walker method shows its advantage over fixed frequency bands limitation.

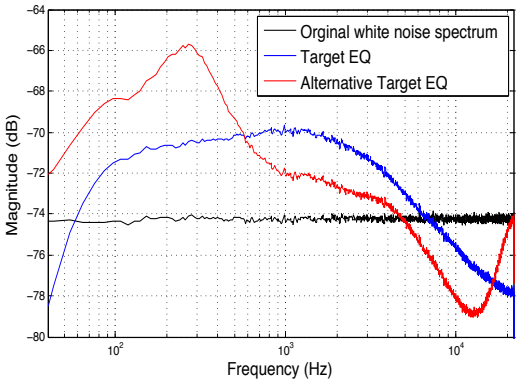

Figure 9. Test result using white noise signal

Following the same process, a real modern music signal is also tested in this case. The results are presented in Figure 10. The averaged output spectrums of the song after being processed by both plug-ins appear to lie close to each other in general. However, by examining the actual difference between the spectrum produced by our target EQ and the one of the alternative target EQ depicted in Figure 11, we see irregular variations across the whole frequency range.

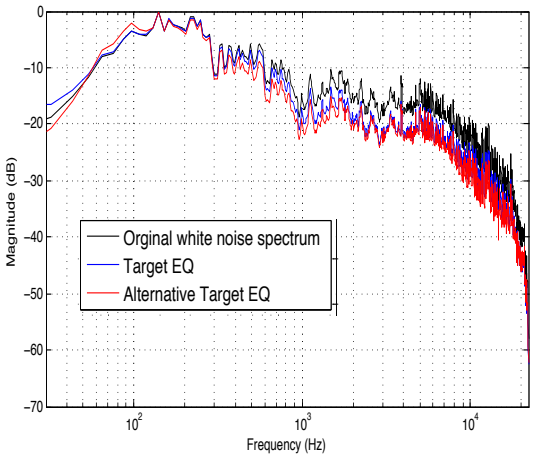

Figure 10. Test result using white noise signal

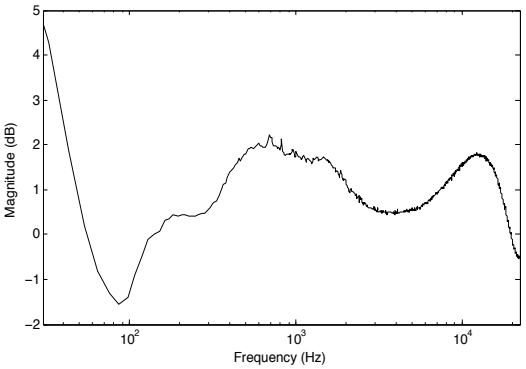

Figure 11. The difference between the spectrums obtains from our target EQ and the alternative target EQ.

Subjective evaluation in the form of formal listening tests may be performed as future work to assess the ability of both target EQ implementations for improving the sound quality.

5. CONCLUSION AND FUTURE WORK

A target equalization curve has been extracted from previous research on spectral characteristic analysis of a large dataset of popular commercial recordings. It can be used as a reference for spectral balance as best practice in audio mixing. A new time-varying equalization approach to match the spectral distribution of input signal to the target equalization curve based on Yule-Walker algorithms was presented. A real-time C++ VST plug-in and Matlab program had been implemented. Objective evaluation of the algorithm

shows that the algorithm is able to fulfil the matching objective with appropriate parameter setting. With the motivation to develop an intelligent mixing and mastering tool to allow the emulation of real-time decisions made by a sound engineer, subjective evaluation to see whether the plug-in can improve the quality of music mixing is necessary. Subjective evaluation should be conducted as future work in the form of a multiple stimulus listening test, similar to those used in MUSHRAM framework [21], where the method is compared against the results of a manual equalization and other match EQ implementations. This research can be expanded and used in the field of intelligent mixing systems.

6. ACKNOWLEDGEMENTS

Thanks to Stuart Mansbridge of Mix Genius and Tandem Launch Technologies for sharing an example target equalization implementation (the 'alternate target EQ' referred to previously).

7. REFERENCES

[1] R. Izhaki, *Mixing Audio — Concepts, Practices and Tools*, Elsevier Science & Technology, Oxford, first edition, 2008.

[2] P. D. Pestana, "*Personal interviews and discussions with mix engineers," 2010-2012*

[3] P.D. Pestana, Z. Ma, "*Spectral Characteristics Of Popular Commercial Recordings 1950-2010*," in Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2013 (under reviewing).

[4] J. D. Reiss, "*Intelligent Systems for Mixing Multichannel Audio*," in 17th International Conference on Digital Signal Processing (DSP), 2011, pp. 1–6.

[5] J.Loviscach, *"Graphical Control of a Parametric Equalizer, "*, AES Paper 7437, 124th Convention, 2008.

[6] J.Loviscach, *"A Real-Time Rhythmic Analyzer and Equalizer,"* AES Paper 6973 ,121st Convention, 2006.

[7] S. Heise, J.Loviscach, *"A Computer-Aided Audio Effect Setup Procedure for Untrained Users,"* AES Paper 8005, 128th AES Convention, 2010

[8] Mecklenburg, S. and J. Loviscach, "*Subject: Controlling an equalizer through subjective terms,*" Extended Abstracts of the Proc. of ACM CHI 2006. NY: ACM Press. : Montreal, Canada. p. 1109-1114.

[9] Bryan Pardo, David Little, and Darren Gergle. "*Building a Personalized Audio Equalizer Interface with Transfer Learning and Active Learning,*" 2nd International ACM Workshop on Music Information Retrieval with User-Centered and Multimodal Strategies (MIRUM), Nara, Japan, November 2, 2012.

[10] Andy Sabin and Bryan Pardo. "*A Method for Rapid Personalization of Audio Equalization Parameters,*" ACM Multimedia 2009, Beijing, China, October 19 - 24, 2009.

[8] M. Filanovsky, H. Baltes, "*CMOS Schmitt Trigger Briefs*", in IEEE Trans. Circuits Syst. I, Fundamental Theory and Applications 1994, Vol 41, No. 1, pp. 46-49

[9] D. C. Farden and L. L. Scharf, "*Statistical design of nonrecursive digital filters*," IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-22, pp. 188–196, June 1974.

[10] V. R. Algazi and M. Suk, "*On the frequency weighted Least-Squares design of finite duration filter*," IEEE Trans. Circuits Syst., vol. CAS-22, pp. 943–953, Dec. 1975.

[11] E. Perez Gonzalez and J. D. Reiss, "*Automatic equalization of multi-channel audio using cross-adaptive methods,*" Proceedings of the 127th AES Convention, New York, October 2009

[12] V. R. Algazi, M. Suk, and C. S. Rim,"*Design of almost minimax FIR filters in one and two dimensions by WLS technique*," IEEE Trans. Circuits Syst., vol. CAS-33, pp. 590–596, June 1986.

[13] M. O. Ahmad and J. D. Wang, "*An analytical Least-Squares solution to the design problem of two-dimensional FIR filters with quadrantally symmetric or antisymmetric frequency response*," IEEE Trans. Circuits Syst., vol. 36, pp. 968–979, July 1989.

[14] T. Kobayashi and S. Imai, "*Design of IIR digital filters with arbitrary log magnitude function by WLS technique*," IEEE Trans. Acoust., Speech, Signal Processing, vol. 38, pp. 247–252, Feb. 1990.

[15] Y. C. Lim, J. H. Lee, C. K. Chen, and R. H. Yang, "*A weighted Least-Squares algorithm for quasi-equiripple FIR and IIR digital filter design*," IEEE Trans. Acoust., Speech, Signal Processing, vol. 40, pp. 551–558, Mar. 1992.

[15] S. C. Pei and J. J. Shyu, "*Design of arbitrary FIR log filters by weighted Least-Squares technique*," IEEE Trans. Signal Processing, vol. 42, pp. 2495–2499, Sept. 1994.

[16] S. Sunder and V. Ramachandran, "*Design of recursive differentiators with constant group delay characteristics*," Signal Process., vol. 39, pp. 79–88, Sept. 1994.

[17] B. Friedlander, B. Porat, "*The Modified Yule- Walker Method of ARMA Spectral Estimation*," IEEE Transactions on Aerospace Electronic Systems, AES-20, no. 2, pp. 158-173, Mar. 1984.

[18] Richard Lee, *"Simple arbitrary IIRs,"* in *AES* 125th Convention, 2008.

[19] S.P. Mansbridge and J.D.Reiss,*"Implementation and evaluation of autonomous multitrack fader controls for automatic mixing," in Proceedings of the 132nd Audio Engineering Society Convention*, Budapest, Hungary, April 2012.

[20] S.P. Mansbridge, S. Finn and J.D. Reiss, *"An Autonomous System for Multi-track Stereo Pan Positioning,"* in Proceedings of the 133nd Audio Engineering Society Convention, San Francisco, October 2012.

[21] International Telecommunication Union, "*Multiple Stimuli with Hidden Reference and Anchor*", ITU-R BS.1534-1, 2003.

[22] International Telecommunication Union. Rec. ITU-R BS.1770-2, "*Algorithms to measure audio programme loudness and true-peak audio level*". Geneva, 2011.