Behavior Research Methods
2007, 39 (1), 133-136

Loudspeaker equalization for auditory research

Justin A. MacDonald and Phuong K. Tran
Army Research Laboratory, Aberdeen Proving Ground, Maryland

The equalization of loudspeaker frequency response is necessary to conduct many types of well-controlled auditory experiments. This article introduces a program that includes functions to measure a loudspeaker's frequency response, design equalization filters, and apply the filters to a set of stimuli to be used in an auditory experiment. The filters can compensate for both magnitude and phase distortions introduced by the loudspeaker. A MATLAB script is included in the Appendix to illustrate the details of the equalization algorithm used in the program.

Loudspeakers introduce both magnitude and phase distortions into an audio signal. Magnitude distortions are introduced because the efficiency of any given loudspeaker varies with the frequency of its input. Loudspeakers also distort the phase of the signal; the delay between signal input and sound output varies with frequency. In addition, the degrees of magnitude and phase distortions vary between loudspeakers.

Human listeners are quite good at detecting very small differences in loudspeaker characteristics. Karjalainen, Piirilä, Järvinen, and Huopaniemi (1999) used an adaptive same–different procedure to measure the just-noticeable difference (JND) between a pair of frequency responses. Digital filters were applied to three auditory stimuli (pink noise, a speech signal, and a musical excerpt) to generate slightly different loudspeaker frequency responses. Listeners were able to discriminate between loudspeaker frequency magnitude responses that differed by no more than $\pm$1.5 dB across the audible range. Although less obvious than magnitude distortions, listeners are also able to perceive phase distortions introduced by loudspeakers. Blauert and Laws (1978) measured the JND for phase distortions at 0.5–1.1 msec. This figure was obtained by simulating a loudspeaker's frequency response and then varying the delay at several frequencies. Typical medium-quality loudspeakers exhibit minimal delay at middle and high frequencies, and therefore any differences in phase response between loudspeakers at these frequencies are likely to be less than the JND (Karjalainen et al., 1999). However, loudspeaker induced phase distortions are generally much more severe at lower frequencies. Suzuki, Morita, and Shindo (1980; see also Johansen & Rubak, 1996) demonstrated that listeners can distinguish between loudspeakers that have been equated for their frequency magnitude response. The differences in phase response alone were enough to allow for above-chance discrimination performance.

Distortions introduced by the loudspeaker can be problematic in psychoacoustics experiments, regardless of the number of loudspeakers used. In an auditory experiment, conclusions are valid only if they are arrived at on the basis of the sounds output from the loudspeaker, rather than of the signals input to the loudspeaker. Because of the wide variation in frequency response across loudspeakers, many experiment results are contingent upon the loudspeaker used to present the stimuli. An equalized loudspeaker ensures effectively identical inputs and outputs and eliminates the loudspeaker as a qualifier of experiment conclusions. Of course, loudspeaker equalization is less important in some experiment designs, such as those that use simple auditory stimuli. When complex stimuli are used, however, a suboptimal (nonflat) frequency response will change the makeup of the stimulus and could alter the results of the experiment. If the participant is asked to judge sound quality, for example, distortions introduced by the loudspeaker could have a pronounced effect.

Loudspeaker equalization is especially important for multiloudspeaker studies; even slight differences between loudspeakers could introduce noise into experimental data and lead to indeterminate results. A wide variety of psychoacoustics experiments fall into this category. Most obviously, any experiment that uses headphones for stimulus presentation is likely to require transducers with the same frequency magnitude and phase response. For example, sounds presented through a spatial audio interface are usually filtered to remove the distortions introduced by the headphones. As another example, any well-controlled experiment in sound localization or distance estimation that uses multiple loudspeakers for stimulus presentation will require loudspeaker equalization. Allowing the frequency response to vary across location adds additional cues that will aid in the determination of sound source location.

If the problem of frequency response in multiloudspeaker studies is addressed at all, a common solution is to use loudspeakers with similar frequency magnitude responses. This is not ideal, since it is possible that listeners will still be able to discriminate between loudspeak-

J. A. MacDonald, jmacd@nmsu.edu

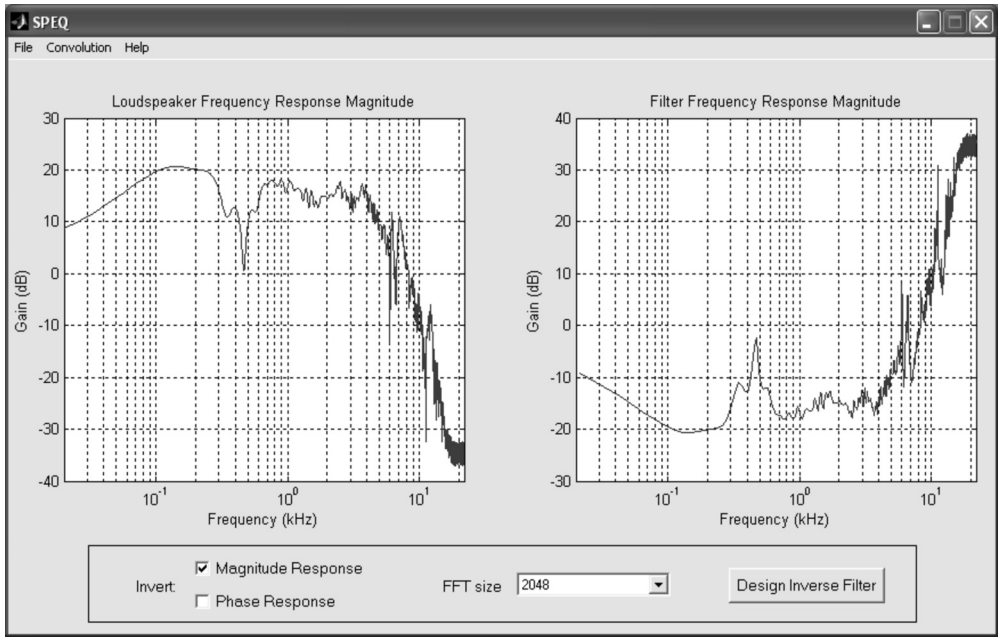

Figure 1. SPEQ program interface.

ers based solely upon their frequency responses. A better approach is to build a filter that "inverts" the frequency response of the loudspeaker in such a way that the signal is sent through the filter and output through the loudspeaker. This process effectively negates the distortions introduced by the loudspeaker, allowing for a practically flat frequency magnitude response. In the past, equalization filters were built using analog components; they provided a frequency magnitude response moderately close to the ideal. Analog filters are not easily configured to correct for the phase distortions introduced by the loudspeaker, but advances in digital signal processing techniques have allowed for the design of digital filters that can compensate for both magnitude and phase distortion. Digital filters can be implemented, in near real time, in hardware or software.

One of the major obstacles preventing the widespread use of this approach within psychoacoustics is the technical background required to design digital equalization filters. In an effort to overcome this problem, we have created a software program, intended for use by nonspecialists, that simplifies the design of equalization filters. The program, SPEQ (for speaker equalization), allows for the easy design of digital filters to equalize both the magnitude and phase response of loudspeakers or headphones. This article includes a brief overview of the features of the software program, as well as a more detailed explanation of the underlying equalization algorithm. An example of a MATLAB script is included in the Appendix for those who wish to customize the equalization routine for their own needs.

The SPEQ Program

SPEQ was developed in MATLAB on a Microsoft Windows platform and is available directly from the authors as MATLAB code or as a standalone Windows executable file. The code can be ported to Macintosh and Unix/Linux operating systems using the appropriate versions of MATLAB Compiler. A screen shot of the SPEQ program is shown in Figure 1. Within a single program, the user can measure the frequency response of the loudspeaker, design a filter that equalizes both the magnitude and phase responses, and apply the filter to a set of sound files to be saved to disk. The filtered files can then be used as stimuli in an experiment of the user's choice.

SPEQ includes functions to measure the frequency response of a loudspeaker using a computer sound card and a microphone. This is accomplished by playing a maximum length sequence (MLS) calibrated stimulus (see Rife & Vanderkooy, 1989) through the loudspeaker and recording the output using a microphone. Both the loudspeaker and microphone are connected to the computer's sound card. Following the onscreen instructions, the user presses a single button to play the MLS stimulus from the loudspeaker and record the result using the microphone. Assuming that the microphone has a flat frequency response (a common assumption, especially when a medium to high quality microphone is used), the program calculates the combined frequency response of the sound card and loudspeaker. The SPEQ program then produces equalization filters that negate the distortions introduced by the sound card–loudspeaker combination. The equalization filter can then be applied to the auditory stimuli that will be presented during the experiment using the same apparatus.

The placement of the loudspeaker and microphone for frequency response measurement varies with the type of equalization required. If the user is not concerned about room effects (for example, if wall reflections are unlikely to affect the outcome of the experiment), SPEQ can be

configured to truncate the recording of the loudspeaker output, so that room reverberations are not included in the measured frequency response. This allows for measurements to be made in semiechoic spaces; accurate response estimates are achievable as long as reverberations do not interfere with the direct reception of the loudspeaker output. In this arrangement, the microphone is directed toward the loudspeaker and placed a short distance away. Alternatively, reverberations can be included in the measured frequency response to compensate for room effects. This is usually necessary when the researcher wants to equalize the outputs of a particular configuration of loudspeakers in a particular room. This can be accomplished by placing the microphone at the listener's location and arranging the loudspeakers in the same configuration to be used in the experiment.

The program can design three types of equalization filters: magnitude, phase, or both. Magnitude equalization is accomplished by designing filters that effectively flatten the magnitude response of the loudspeaker. The resolution of the fast Fourier transforms (FFTs) used in the inversion algorithm can also be specified. Increased resolution leads to more accurate (and higher order) equalization filters. The user can change the FFT resolution to determine the best compromise between equalization accuracy and filter order. The user can either apply the equalization filter to a group of sound files to be saved to disk in Microsoft WAV format, or save the equalization filter to disk for later use.

The Equalization Algorithm

The equalization algorithm used in SPEQ is best understood by referring to the example of a MATLAB script in the Appendix. To equalize a set of loudspeakers, the frequency response of each loudspeaker must be measured. The script is written to accept the impulse response of a loudspeaker (or headphone), although the same information transferred into the frequency domain can be used quite easily. In the example script, the impulse response of each loudspeaker was represented as a 320-point finite impulse response (FIR) filter that approximates the actual impulse response of the loudspeaker. The impulse response should be sampled at the same rate as the stimuli to be filtered; if not, the impulse response should be resampled appropriately before equalization. The length of the impulse response is also an important consideration. Longer impulse responses provide a more accurate representation of the frequency response but require higher order equalization filters. In any case, the impulse response is transferred into the frequency domain via an FFT. The variable *n* sets resolution of the FFT. Some experimentation is required to find the appropriate value for this parameter. Setting *n* to a relatively small value can lead to equalization filters that do not possess the desired frequency characteristics, whereas a large value for *n* could lead to prohibitively large equalization filters. In any case, the magnitude and phase responses of the loudspeaker are stored as the complex vector IR. Throughout the script, variable labels printed in capital letters represent complex vectors in the frequency domain, and labels in lowercase represent real-valued vectors in the time domain.

The loudspeaker's frequency response (represented by the vector IR) can be split into two components: the "minimum phase" portion that specifies the magnitude and some of the phase response of the transducer, and the "allpass" portion that specifies the excess (remaining) phase response (see Greenfield & Hawksford, 1991; Mourjopoulos, Clarkson, & Hammond, 1982). It is common practice to focus exclusively upon the minimum phase portion, since it contains all of the magnitude response. The allpass portion is often ignored for two reasons: it is not easily invertible and it contains only phase response information. However, both magnitude and phase equalization are included in this script for illustrative purposes.

The details regarding the separation of the loudspeaker's frequency response into the minimum phase and allpass portions are beyond the scope of this article. In the script, MP represents the minimum phase portion of the filter and includes all of the frequency response magnitude characteristics of the loudspeaker. The remainder of the response is contained in the allpass portion of the frequency response, AP.

The next step is to design a pair of equalization filters to equalize the magnitude and phase responses. The magnitude equalization filter is designed by inverting each element of the vector MP. MPINV is converted back into the time domain with an inverse FFT, resulting in a magnitude equalization filter (mag_eq) that effectively flattens the frequency response magnitude of the loudspeaker. The phase equalizer is constructed using the procedure developed by Greenfield and Hawksford (1991; see also Hawksford, 1999). The allpass portion of the filter (AP) is translated into the time domain. The result (ap) is then time shifted and time reversed to produce the phase equalizing filter phase_eq. In effect, this procedure produces a pure delay filter that adds a delay to each frequency component, so that all frequencies sent through the loudspeaker are delayed by the same amount of time. Therefore, phase equalization is achieved at the expense of an overall group delay.

Final Comments

The main drawback of this equalization procedure is the high order FIR filters that are required. When a stimulus is processed by a FIR filter, the output includes a delay that is directly related to the length of the filter. In the example script, 1,024-point FFTs led to a 1,024-tap magnitude equalization filter and a 2,048-tap phase equalization filter. The use of these filters will introduce a significant delay—2,047 samples in this case, which corresponds to approximately 46 msec at a 44.1-kHz sampling rate. This delay could introduce an unacceptable asynchrony when multimodal stimuli are used; however, this problem is unlikely to be a concern to most speech and hearing researchers. The majority of detection and discrimination tasks do not require the real-time filtering of stimuli, and a constant delay across all stimuli should not be a problem for purely auditory tasks. There are several ways of decreasing this delay, including reducing the order of the filters by neglecting to consider the phase response; minimizing the length of the impulse response measured from

the loudspeaker; or implementing the phase equalizer as a lower order infinite impulse response (IIR) filter (Greenfield & Hawksford, 1991).

The SPEQ program does not offer real-time equalization capabilities, although this feature could presumably be implemented in software if a powerful computing platform were used. Hardware-based solutions are another possibility for near real-time filtering, although lower order filters would likely be required in this case. However, real-time equalization, although an elegant solution, is not really needed in most acoustic and psychoacoustic studies. Equalization using prefiltered sound files requires minimal processing power and no specialized hardware beyond a standard computer, sound card, and microphone.

Several software packages are available to measure the frequency response of a loudspeaker but do not allow for the design of equalization filters. Of the small number of programs available to design equalization filters, most are intended for use with DSP hardware and do not allow for the equalization of phase response. The SPEQ program allows for both frequency response measurement and equalization filter design within a single user-friendly software package. The equalization filters can conveniently be applied to auditory stimuli for use in future experiments.

AUTHOR NOTE

Address correspondence concerning this article to J. A. MacDonald, New Mexico State University, P.O. Box 30001/MSC 3452, Las Cruces, NM 88003 (e-mail: jmacd@nmsu.edu).

REFERENCES

Blauert, J., & Laws, P. (1978). Group delay distortions in electroacoustical systems. *Journal of the Acoustical Society of America*, **63**, 1478-1483.

Greenfield, R., & Hawksford, M. O. (1991). Efficient filter design for loudspeaker equalization. *Journal of the Audio Engineering Society*, **39**, 739-751.

Hawksford, M. O. (1999). MATLAB program for loudspeaker equalization and crossover design. *Journal of the Audio Engineering Society*, **47**, 706-719.

Johansen, L. G., & Rubak, P. (1996, April). *The excess phase in loudspeaker/room transfer functions: Can it be ignored in equalization tasks?* Paper presented at the Audio Engineering Society 100th Convention, Copenhagen.

Karjalainen, M., Piirilä, E., Järvinen, A., & Huopaniemi, J. (1999). Comparison of loudspeaker equalization methods based on DSP techniques. *Journal of the Audio Engineering Society*, **47**, 14-31.

Mourjopoulos, J., Clarkson, P. M., & Hammond, J. K. (1982). A comparative study of least-squares and homomorphic techniques for the inversion of mixed-phase signals. In *Proceedings of ICASSP 82: The IEEE International Conference on Acoustics, Speech, and Signal Processing* (pp. 1858-1861). New York: IEEE.

Rife, D. D., & Vanderkooy, J. (1989). Transfer-function measurement with maximum-length sequences. *Journal of the Audio Engineering Society*, **37**, 419-444.

Suzuki, H., Morita, S., & Shindo, T. (1980). On the perception of phase distortion. *Journal of the Audio Engineering Society*, **28**, 570-574.

APPENDIX
MATLAB Script Example

```
Function [mag_eq,phase_eq] = equalize(filename)
% This matlab function will return the inverse of the minimum phase, mpeq,
% and the inverse of the allpass, apeq, of a measured impulse response.
% Filename is the name of the text file that includes the impulse response.
n = 1024;
ir_file = fopen(filename,'rt');
ir = fscanf(ir_file,'%f');
fclose(ir_file);
ir = ir ./ max(abs(ir));% Normalize the impulse response.
%Compute minimum phase portion of the frequency response.
IR = fft(ir,n);
ir_cepstrum = real(ifft(log(abs(IR))));
w = [1; 2*ones((n/2) - 1); ones(1 - rem(n,2),1); zeros((n/2) - 1,1)];
MP = exp(fft(w .* ir_cepstrum));
% Compute the allpass portion of the frequency response.
AP = IR ./ MP;
% Compute the magnitude equalization filter (mag_eq).
MPINV = ones(n,1) ./ MP;
mag_eq = real(ifft(MPINV));
% Compute the phase equalization filter (phase_eq).
ap = real(ifft(AP));
phase_eq = [zeros(n,1); flipud(ap)].
```

(Manuscript received May 31, 2004;
revision accepted for publication December 13, 2005.)