Audio Engineering Society
Convention e-Brief 403
Presented at the 143rd Convention
2017 October 18–21, New York, NY, USA



# A Toolkit for Customizing the ambiX Ambisonics-to-Binaural Renderer

Joseph G. Tylka and Edgar Y. Choueiri

*3D Audio and Applied Acoustics Laboratory, Princeton University, Princeton, NJ, 08544, USA*

Correspondence should be addressed to Joseph G. Tylka (josephgt@princeton.edu)

**ABSTRACT**

An open-source collection of MATLAB functions, referred to as the SOFA/ambiX binaural rendering (SABRE) toolkit, is presented for generating custom ambisonics-to-binaural decoders for the ambiX binaural plug-in. Databases of head-related transfer functions (HRTFs) are becoming widely available in the recently-standardized "SOFA format" (spatially-oriented format for acoustics), but there is currently no (easy) way to use custom HRTFs with the ambiX binaural plug-in. This toolkit enables the user to generate custom binaural rendering configurations for the plug-in from any SOFA-formatted HRTFs or to add HRTFs to an existing ambisonics decoder. Also implemented in the toolkit are several methods of HRTF interpolation and equalization. The mathematical conventions, ambisonics theory, and signal processing implemented in the toolkit are described.

## 1 Introduction

Binaural rendering of ambisonics enables a user to convert the multichannel ambisonics representation of a 3D sound field into a spatially-accurate, two-channel representation suitable for playback over headphones. Ideally, when rendering, the user would apply their own individualized head-related transfer functions (HRTFs) in order to achieve the highest possible spatial fidelity and an externalized sound image. However, freely-available tools for creating such individualized binaural renderings are limited.

Recently, Kronlachner [1] released an open-source suite of ambisonics plug-ins (known as the "ambiX plug-ins") which includes a plug-in for rendering ambisonics to binaural. Additionally, HRTFs are becoming widely available in the recently-standardized "SOFA format" (spatially-oriented format for acoustics) [2], but there is currently no (easy) way to use custom HRTFs with the ambiX binaural plug-in.

Consequently, it is the goal of this work to provide a freely-available tool for users to create custom (and ideally, individualized) binaural renderings of ambisonics via the ambiX binaural plug-in. To that end, we present an open-source collection of MATLAB functions for the purpose of creating ambiX binaural rendering configurations (also called "decoder presets") from SOFA-formatted HRTFs.

In Sec. 2, we describe the ambiX mathematical conventions and subsequently, in Sec. 3, we describe the ambisonics decoding approaches implemented in this toolkit. Then, in Sec. 4, we discuss the various processing options that can be applied to the HRTFs. Finally, we summarize these contributions in Sec. 5.

## 2 Conventions

In accordance with the ambiX specification [3], we use real-valued spherical harmonics as given by

$$Y_l^m(\theta,\phi) = N_l^{|m|} P_l^{|m|}(\sin\theta) \times \begin{cases} \cos m\phi & \text{for } m \geq 0, \\ \sin |m|\phi & \text{for } m < 0, \end{cases}$$

where $P_l^m$ is the associated Legendre polynomial of degree $l$ and order $m$, as defined in the MATLAB `legendre` function by

$$P_l^m(x) = (-1)^m (1-x^2)^{m/2} \frac{d^m}{dx^m} P_l(x),$$
$$\text{with} \qquad P_l(x) = \frac{1}{2^l l!} \left[ \frac{d^l}{dx^l} (x^2-1)^l \right],$$

and $N_l^m$ is a normalization term which, for the Schmidt seminormalized (SN3D) spherical harmonics with Condon-Shortley phase,[1] is given by [3]

$$N_l^m = (-1)^m \sqrt{\frac{2-\delta_m}{4\pi} \frac{(l-m)!}{(l+m)!}},$$

where $\delta_m$ is the Kronecker delta. With an inner product defined by integrating over all directions, the squared-norm of these spherical harmonics is thus given by

$$\|Y_l^m\|^2 = \frac{1}{2l+1}.$$

We also adopt the ambisonics channel numbering (ACN) convention [3] such that for a spherical harmonic function of degree $l \in [0,\infty)$ and order $m \in [-l,l]$, the ACN index $n$ is given by $n = l(l+1)+m$ and we denote the spherical harmonic function by $Y_n \equiv Y_l^m$.

## 3 Decoding Ambisonics

Implemented in this toolkit are several basic methods of decoding ambisonics (described below), but the Ambisonic Decoder Toolbox (ADT)[2] is a much more comprehensive tool for creating state-of-the-art ambisonic decoders [4]. Consequently, one intended use of this toolkit is to add custom HRTFs to an existing ambiX decoder preset (such as those generated using the ADT).[3] Note that doing so requires the user to specify the grid of speaker positions, as that information is not explicitly contained in ambiX decoder presets.

Generally, the decoding matrix, $\mathbf{D}$, determines the loudspeaker signals, $x_q$, given by

$$\mathbf{x} = \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_Q(t) \end{bmatrix} = \mathbf{D} \cdot \begin{bmatrix} a_0(t) \\ a_1(t) \\ \vdots \\ a_{N-1}(t) \end{bmatrix} = \mathbf{D} \cdot \mathbf{a}, \qquad (1)$$

where $Q$ is the number of loudspeakers and $a_n$ is the ambisonic signal for ACN index $n$. For a thorough review of ambisonic decoding theory and practice, the reader is referred to the works of Heller et al. [4, 5].

Assuming a free field and ideal loudspeakers that are equidistant from the listener, the resulting binaural pressure signals are given by

$$p^{\mathrm{L,R}}(t) = \mathbf{h}^{\mathrm{L,R}} * \mathbf{x} = \sum_{q=1}^{Q} h_q^{\mathrm{L,R}}(t) * x_q(t), \qquad (2)$$

where '$*$' denotes convolution and

$$\mathbf{h}^{\mathrm{L,R}} = \begin{bmatrix} h_1^{\mathrm{L,R}}(t) & h_2^{\mathrm{L,R}}(t) & \cdots & h_Q^{\mathrm{L,R}}(t) \end{bmatrix} \qquad (3)$$

is a vector containing the head-related impulse responses (HRIRs) for a given listener and for the directions of each loudspeaker. The superscript "L,R" denotes the response at the left or right ear, respectively.

### 3.1 Basic Decoding

Following traditional ambisonic theory, we consider the ambisonic signals produced at the center of the loudspeaker array in response to the loudspeaker signals, given by

$$a_n(t) = \sum_{q=1}^{Q} Y_n(\hat{v}_q) x_q(t). \qquad (4)$$

[1] Note that including Condon-Shortley phase in the normalization term cancels it in the associated Legendre term.

[2] The ADT is available online at: `https://bitbucket.org/ambidecodertoolbox/adt.git`

[3] At present, the SABRE toolkit is only compatible with single-band decoders. Consequently, multi-band decoders should be implemented in parallel (as a bank of single-band decoders), downstream of a crossover network.

Equivalently, in matrix form, we have $\mathbf{a} = \mathbf{Y} \cdot \mathbf{x}$, where

$$\mathbf{Y} = \begin{bmatrix} Y_0(\hat{v}_1) & Y_0(\hat{v}_2) & \cdots & Y_0(\hat{v}_Q) \\ Y_1(\hat{v}_1) & Y_1(\hat{v}_2) & \cdots & Y_1(\hat{v}_Q) \\ \vdots & \vdots & \ddots & \vdots \\ Y_{N-1}(\hat{v}_1) & Y_{N-1}(\hat{v}_2) & \cdots & Y_{N-1}(\hat{v}_Q) \end{bmatrix}. \quad (5)$$

From this formulation, we obtain the basic (pseudoinverse) decoder, given by [5, Appendix A.1]

$$\mathbf{D} = \mathbf{Y}^{+}, \quad (6)$$

where $(\cdot)^{+}$ denotes pseudoinversion.

### 3.2 Quadrature Decoding

Given higher-order ambisonics signals, $a_n$, the so-called *signature function*, $\mu$, in the direction $\hat{v}_q$ is given by [6]

$$\mu(t, \hat{v}_q) = \sum_{n=0}^{N-1} a_n(t) \frac{Y_n(\hat{v}_q)}{\|Y_n\|^2}. \quad (7)$$

Equivalently, in matrix form, we have

$$\begin{bmatrix} \mu(t, \hat{v}_1) \\ \mu(t, \hat{v}_2) \\ \vdots \\ \mu(t, \hat{v}_Q) \end{bmatrix} = \mathbf{Y}^{\mathrm{T}} \cdot \mathbf{F}^{-1} \cdot \mathbf{a}, \quad (8)$$

where $Q$ is now the number of plane-wave terms and $\mathbf{F}$ is a diagonal matrix given by

$$\mathbf{F} = \mathrm{diag}\left\{ \begin{bmatrix} \|Y_0\|^2 & \|Y_1\|^2 & \cdots & \|Y_{N-1}\|^2 \end{bmatrix} \right\}. \quad (9)$$

The signature function represents the coefficients of a plane-wave decomposition of the sound field, such that the binaural pressure signals can be approximately reconstructed using a finite number of plane-waves, given by [6]

$$p^{\mathrm{L,R}}(t) \approx \sum_{q=1}^{Q} h_q^{\mathrm{L,R}}(t) * (w_q \mu(t, \hat{v}_q)) = \mathbf{h}^{\mathrm{L,R}} * \begin{bmatrix} w_1 \mu(t, \hat{v}_1) \\ w_2 \mu(t, \hat{v}_2) \\ \vdots \\ w_Q \mu(t, \hat{v}_Q) \end{bmatrix}, \quad (10)$$

where $w_q$ is the quadrature weight of the $q^{\mathrm{th}}$ plane-wave term and is dependent on the chosen grid of directions. Rearranging, we arrive at the quadrature decoder, given by

$$\mathbf{D} = \mathrm{diag}\left\{ \begin{bmatrix} w_1 & w_2 & \cdots & w_Q \end{bmatrix} \right\} \cdot \mathbf{Y}^{\mathrm{T}} \cdot \mathbf{F}^{-1}. \quad (11)$$

### 3.3 Compact Decoding

Now that we have established the typical decoding and binaural rendering signal chain, we can derive an equally valid but more computationally efficient approach. We first combine the HRIRs by performing the matrix multiplication with the decoder matrix, which yields a vector of $N$ compacted HRIRs

$$\tilde{\mathbf{h}}^{\mathrm{L,R}} = \begin{bmatrix} \tilde{h}_0^{\mathrm{L,R}}(t) & \tilde{h}_1^{\mathrm{L,R}}(t) & \cdots & \tilde{h}_{N-1}^{\mathrm{L,R}}(t) \end{bmatrix} = \mathbf{h}^{\mathrm{L,R}} \cdot \mathbf{D}. \quad (12)$$

This process simply combines the decoding matrix and per-direction HRIRs into a single set of filters.[4] The corresponding "compact" decoder is simply an $N \times N$ identity matrix, i.e., $\tilde{\mathbf{D}} = \mathbf{I}_{(N \times N)}$, such that $\tilde{\mathbf{h}}^{\mathrm{L,R}} \cdot \tilde{\mathbf{D}} = \mathbf{h}^{\mathrm{L,R}} \cdot \mathbf{D}$. It's worth noting that, in the case of the basic pseudoinverse decoder, given by Eq. (6), the compacting process described by Eq. (12) is equivalent to to computing the spherical-harmonics coefficients of the HRTFs, as done by Rafaely and Avni [7, Sec. IV].

### 3.4 Normalization

For each HRIR (compacted or not), we first compute the maximum gain, $\alpha_q$, across the entire frequency response. Subsequently, we attenuate each HRIR and amplify the corresponding row of the decoder matrix by that gain, i.e.,

$$\hat{\mathbf{h}}^{\mathrm{L,R}} = \mathbf{h}^{\mathrm{L,R}} \cdot \mathbf{G}^{-1}, \quad \text{and} \quad \hat{\mathbf{D}} = \mathbf{G} \cdot \mathbf{D}, \quad (13)$$

where

$$\mathbf{G} = \mathrm{diag}\left\{ \begin{bmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_Q \end{bmatrix} \right\}, \quad (14)$$

such that $\hat{\mathbf{h}}^{\mathrm{L,R}} \cdot \hat{\mathbf{D}} = \mathbf{h}^{\mathrm{L,R}} \cdot \mathbf{D}$. Finally, we normalize the overall decoder matrix such that the maximum absolute value of any element in matrix is unity.

## 4 HRTF Processing

This toolkit requires HRTFs to be stored stored in SOFA format [2]. The HRTF files contain, among other things, the measured HRIRs and the grid of corresponding measurement positions. Depending on the decoder used, the HRTFs may need to be interpolated, and depending on the intended playback system (e.g., type of headphones), the HRTFs may need to be equalized. Consequently, the SABRE toolkit contains several options for carrying out these processes.

[4] Conceptually, each compacted HRIR $\tilde{h}_n^{\mathrm{L,R}}(t)$ represents the signals at the ears in response to an impulse sent through the $n^{\mathrm{th}}$ HOA channel.

### 4.1 Interpolation

When measured HRTFs are not available at the desired grid positions, interpolation is performed through one of the following methods.

**Nearest Neighbor:** By default, we perform nearest-neighbor interpolation. This is carried out by minimizing the $\ell^2$ norm (Euclidean distance) between the desired position $\vec{v}_{q'}$ and each measurement position $\vec{u}_q$.

**Time Domain:** Alternatively, we can perform weighted-average interpolation of the HRIRs for three different interpolation schemes: natural neighbor, linear, and spherical-harmonic.[5] Generally, for some function $f_q$ measured at positions $\vec{u}_q$, the interpolated values, $f'_{q'}$, for all desired positions $\vec{v}_{q'}$, are given by

$$\begin{bmatrix} f'_1 & f'_2 & \cdots & f'_{Q'} \end{bmatrix} = \begin{bmatrix} f_1 & f_2 & \cdots & f_Q \end{bmatrix} \cdot \mathbf{W}, \quad (15)$$

where each element, $w_{q,q'}$, of $\mathbf{W}$ is the interpolation weight from measurement position $\vec{u}_q$ to the desired position $\vec{v}_{q'}$. Before interpolating, we first measure the onset delays, $\tau_q^{\mathrm{L,R}}$, using a 10% ($-20$ dB) threshold for each impulse response. We then align all of the impulse responses such that their onsets coincide at the earliest onset, and separately store the relative delays, given by

$$d_q^{\mathrm{L,R}} = \tau_q^{\mathrm{L,R}} - \min\left(\min_q \tau_q^{\mathrm{L}},\ \min_q \tau_q^{\mathrm{R}}\right). \quad (16)$$

Then we compute interpolation weights from each measurement position to each desired position and interpolate, using Eq. (15), the time-aligned impulse responses and the relative delays. Finally, we introduce the interpolated time delays to each interpolated impulse response.

**Frequency Domain:** We can also interpolate by first decomposing the HRTFs into magnitude spectra and time delays. The magnitude spectra are given (in dB) by

$$M_q^{\mathrm{L,R}}(f) = 10\log_{10}\left(\left|H_q^{\mathrm{L,R}}(f)\right|^2\right), \quad (17)$$

where $H$ denotes the Fourier transform of $h$. We then interpolate the magnitude responses and time delays using Eq. (15). The interpolated magnitude responses are then converted into minimum-phase impulse responses, and the interpolated onset delays are introduced to yield the final interpolated HRIRs.

[5] For spherical-harmonic interpolation, the interpolation weights are given as a matrix by $\mathbf{W} = \mathbf{Y}_Q^+ \mathbf{Y}_{Q'}$, where $\mathbf{Y}_Q$ is given by Eq. (5) for all measurement positions and up to some maximum order (by default, $L = 4$), and $\mathbf{Y}_{Q'}$ is the same for all desired positions.

#### 4.1.1 Interpolation Threshold

Optionally, we may apply a threshold to determine which desired positions are close enough to a measurement position such that they do not require interpolation. For each desired position, we find the nearest measurement position and compute the angular distance between the two, given by

$$\psi = \cos^{-1}\left(\frac{\vec{u}_q \cdot \vec{v}_{q'}}{\|\vec{u}_q\| \cdot \|\vec{v}_{q'}\|}\right), \quad (18)$$

where $\|\cdot\|$ denotes the $\ell^2$ norm of a vector. If this angular distance exceeds a user-specified threshold, then the selected interpolation method is carried out. Otherwise, nearest-neighbor interpolation is used.

### 4.2 Equalization

For optimal binaural playback, one should use individually equalized headphones [8]. As this may not always be possible, we provide several methods of equalization so that the user may try to compensate for the equalization of the headphones. We design the equalization filters using the full set of measured HRTFs and apply them to the (possibly interpolated) HRTFs for the desired positions.

**None:** By default, the HRTFs are not equalized. This option should only be used if the playback headphones will be individually equalized on the user's ears.

**Frontal:** For headphones that use frontal-incidence "free-field" equalization, we can equalize all HRTFs by the HRTF pair nearest to $(\theta, \phi) = (0, 0)$. The transfer function of the regularized inverse filter is given by [9]

$$Z(f) = \frac{H^*(f)}{H^*(f)H(f) + \beta(f)}, \quad (19)$$

where $(\cdot)^*$ denotes complex conjugation and $\beta$ is a frequency-dependent regularization function. This function is defined by a set of parameters, which are defined graphically in Fig. 1, and whose default values are given by

$$\begin{array}{lll} \beta_0 = 10^{-4}, & f_{L0} = 50 \text{ Hz}, & f_{H0} = 21 \text{ kHz}, \\ \beta_1 = 10^{-2}, & f_{L1} = 20 \text{ Hz}, & f_{H1} = 22 \text{ kHz}. \end{array}$$

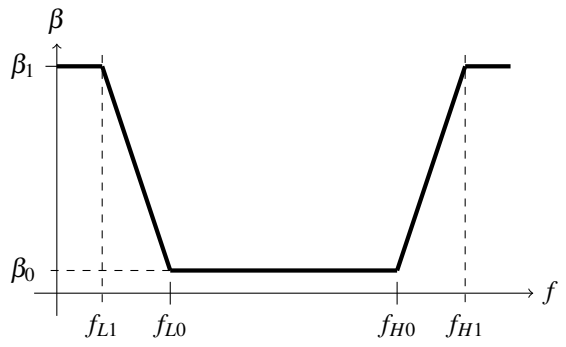


**Fig. 1:** Frequency-dependent regularization function of the inverse filters for HRTF equalization.

**Diffuse:** For headphones that employ diffuse-field equalization, we can equalize the HRTFs by the average magnitude spectrum over all directions. This is computed as the omnidirectional term of the spherical-harmonic decomposition of the HRTF magnitude spectra (in dB), where the decomposition is computed using the pseudoinverse of $\mathbf{Y}$, given by Eq. (5) for all measurement directions and up to order $L = 4$. The equalization filter is then computed for the average magnitude spectrum using Eq. (19).

**Horizontal:** Alternatively, we can compute an average HRTF over all horizontal-plane directions. The procedure for this is very similar to the diffuse-field equalization, but the average spectrum is computed using only the HRTFs with elevation $|\theta| < 5°$. The equalization filter is then computed for the average magnitude spectrum using Eq. (19).

## 5 Summary

The SOFA/ambiX binaural rendering (SABRE) toolkit presented here is an open-source collection of MATLAB functions for generating custom binaural decoders. The toolkit allows a user to construct, using any SOFA-formatted HRTFs, a custom binaural rendering configuration for the ambiX binaural plug-in. Also implemented in the toolkit are several methods of HRTF interpolation, as well as common HRTF equalization options. The toolkit is freely available online.[6]

[6]See: `https://github.com/PrincetonUniversity/3D3A-SABRE-Toolkit`

## Acknowledgements

This toolkit relies on the ambiX ambisonic plug-in suite[7] by M. Kronlachner and requires head-related transfer functions (HRTFs) stored in the SOFA format.[8] Accordingly, the SOFA API[9] for MATLAB is needed to import the HRTF files.

## References

[1] Kronlachner, M., "Ambisonics plug-in suite for production and performance usage," in *Linux Audio Conference*, 2013.

[2] AES69-2015, "AES69-2015: AES standard for file exchange - Spatial acoustic data file format," 2015.

[3] Nachbar, C., Zotter, F., Deleflie, E., and Sontacchi, A., "ambiX - A Suggested Ambisonics Format," in *Proceedings of the 3rd Ambisonics Symposium*, 2011.

[4] Heller, A. J., Benjamin, E. M., and Lee, R., "A Toolkit for the Design of Ambisonic Decoders," in *Linux Audio Conference*, 2012.

[5] Heller, A., Lee, R., and Benjamin, E., "Is My Decoder Ambisonic?" in *Audio Engineering Society Convention 125*, 2008.

[6] Duraiswami, R., Zotkin, D. N., Li, Z., Grassi, E., Gumerov, N. A., and Davis, L. S., "High Order Spatial Audio Capture and Its Binaural Head-Tracked Playback Over Headphones with HRTF Cues," in *Audio Engineering Society Convention 119*, 2005.

[7] Rafaely, B. and Avni, A., "Interaural cross correlation in a sound field represented by spherical harmonics," *The Journal of the Acoustical Society of America*, 127(2), pp. 823–828, 2010, doi: http://dx.doi.org/10.1121/1.3278605.

[8] Schärer, Z. and Lindau, A., "Evaluation of Equalization Methods for Binaural Signals," in *Audio Engineering Society Convention 126*, 2009.

[9] Farina, A., "Advancements in Impulse Response Measurements by Sine Sweeps," in *Audio Engineering Society Convention 122*, 2007.

[7]See: `http://www.matthiaskronlachner.com/?p=2015`

[8]See: `https://www.sofaconventions.org`

[9]See: `https://github.com/sofacoustics/sofa`