AMTatARI Documentation

AMTatARI version: 7.0.26

last update: 02.11.2021 by Michael Mihocic

Table of Contents

License

Description

AMTatARI is a software application based on the ExpSuite framework, designed to measure transfer functions like room impulse responses or head-related transfer functions. This documentation will guide you from the first steps of installation to the optimization of parameters.

ExpSuite

For further details about ExpSuite (the framework), its applications, development, and a detailed explanation of the main window it is recommended to read the *Getting Started* manual provided on Sourceforge: https://sourceforge.net/p/expsuite/code/HEAD/tree/FrameWork/net-1.1/doc/

Installation

For the first installation please run the full ExpSuite installer package including some required software packages. For every following update it is enough to update AMTatARI only.

System Requirements

Please make sure that your system fulfills the following requirements:

- Windows 10 operating system
- AMTatARI is using several MATLAB scripts for generating acoustical stimuli and to plot result figures. AMTatARI is optimized for MATLAB R2018b, or newer.
- Microsoft .NET FrameWork 3.5 is required to use any ExpSuite application.
- Audio interface (ASIO is supported)

Optional Hardware

ExpSuite and AMTatARI support the following optional hardware:

- Turntable FourAudio ANT (USB port)
- Turntable Outline ST2 (LPT port)
- Tracking system Flock-of-bird (electromagnetic)
- Tracking system Oculus Rift (optical)

ExpSuite Installer Package

For the first installation please use the ExpSuite installer available at Sourceforge (http://sourceforge.net/projects/expsuite/) and run the “AMTatARI installation”, including DirectX libraries and pd & YAMI (Audio player). Please check that you don’t accidentally close one of the installation windows before the installations are completed. In case it has happened anyway you can repeat the installation.

Updates

Once AMTatARI is running on your system there are two ways of checking for updates:

Automatic updates

In the top menu run Help → Check for updates. AMTatARI will scan several Sourceforge mirrors (and if available also local network paths), and check if an update is available.

In AMTatARI options you can also enable periodic automatic checks for updates (top menu: View → Options → Check for updates every … days).

From time to time Sourceforge changes its mirror links, in that case you can always run a manual update.

Manual update

Check on https://sourceforge.net/projects/expsuite/files/Applications/AMTatARI/ for the latest version available and run the installer.

Configuration of Experiments in AMTatARI

Experiments parameters are defined on three different levels:

- Options: Options are the global hardware and system depending parameters. Usually the options are configured once for every machine, and do not need to be changed significantly later. Examples for options are the selection of audio interface(s), a turntable, or a position tracking system. Options usually do not change between experiment types or experiments. For further details see more detailed chapter Options.
- Settings: Settings are usually defined for a specific type of subject, room or type of experiment. Settings can (but do not need to) vary between measurement and measurement. For further details see more detailed chapter Settings.
- Item lists: Item lists are tables that describe a list of conditions for one experiment or one set of measurements. Every item (= row) in the item list contains specific parameters for one stimulus presentation (for example which loudspeakers are being stimulated, amplitude,...) and a status description of the current item. Item lists can (but do not need to) vary between measurement and measurement. For further details see chapter Item List.

Options

The Options can be opened by clicking in the top menu on View → Options. Some of the tabs are irrelevant in AMTatARI but are part of the ExpSuite framework. The options are automatically saved (and the next time re-loaded) on your computer when you click on “ok” or “apply”. The following tabs can be configured:

- General: Define some general application parameters.

- Audio (pd): Define some important parameters here:
    - YAMI start file: YAMI supports up to 24 channels (DAC), YAMI2 supports up to 2 channels (DAC) and YAMI100 supports up to 100 channels (DAC).
    - Define Audio device(s) for playback (DAC) and record (ADC). You can either use the names of the devices (only when using ASIO), or the index.
        - If you are using names they have to be typed correctly. To display a list of registered audio interfaces press the button List Devices.
        - If you are using the indices it is recommended to leave them at "0" for the first time you are connecting to pd, and use pd to find out the proper indices (top menu: Media → Audio Settings → count the index of your desired interface). Once you know your indices for DAC and ADC disconnect AMTatARI, close pd, and connect AMTatARI again. Now the proper sound interfaces should be used.
        - After connecting to pd you can find out which interfaces are used: in pd: top menu: Media → Audio Settings. If you change them here your experiment will work but only until you connect to pd the next time. The recommended long-term solution is to set proper names or indices in AMTatARI options.
    - The number of max channels should be adapted to your system. If your system uses a different range (for example: not the channels from 1-24 but the range of 65-88) be aware that you should cover the higher limit of your range (in our example → 88).
- Audio (Unity): irrelevant in AMTatARI (can be used for audio playback with Unity; ADCs are not implemented, so recordings are not possible with Unity yet)
- RIB: irrelevant in AMTatARI (can be used for electric stimulation with cochlear implants)
- RIB2: irrelevant in AMTatARI (can be used for electric stimulation with cochlear implants)
- Matlab: set Matlab folder; the default parameters should be fine
- CSV Export/Import: keep default values unless you want to use a different comma separator
- Tracker: define if you want to use a tracker
- Turntable: define if you want to use a turntable
- ViWo: irrelevant in AMTatARI (control a virtual world, a graphical interface, presented via head-mounted display like Oculus Rift)
- Joypad: irrelevant in AMTatARI (can be used for user feedback)
- Remote Monitor: irrelevant in AMTatARI (can be used to monitor application from a different computer)

The options are stored locally as a text file on every computer, in the directory
%ProgramData%\Expsuite\AMTatARI\AMTatARI.ini
(this address be copied to Windows Explorer).

Settings

The Options can be opened by clicking in the top menu on View → Settings. Some of the tabs are irrelevant in AMTatARI but are part of the ExpSuite framework. Settings can be saved and loaded for future experiments (top menu: File → Open settings / Save settings as). The following tabs can be configured:

- General:
    - Signal output and input: Audio (pd) recommended; connection to output only required if

stimuli will be played or recorded (but not for post-processing)

  - Output directory: directory used for the created and recorded files, should be adapted for post-processing when an existing folder is used to calculate and add more files (eg. SOFA files); clicking the button with the three exclamation marks (“!!!”) changes the directory to the settings file’s location, by clicking on the “A” button a recommended configuration of an analysis setting is prepared
- Fitting Left: irrelevant in AMTatARI (electric stimulation with cochlear implants)
- Fitting Right: irrelevant in AMTatARI (electric stimulation with cochlear implants)
- Description: The experiment ID is used as prefix for used scripts and output filed. This is an important value in AMTatARI as you will find out in later chapters. If you want to use the same set of Matlab scripts please do not change the experiment ID between experiments.
  The description can be filled with any information that you want to store together with the settings file.
- Experiment Screen: define parameters of the experiment window, like its positions. Not important unless the experiment screen is hiding something important during measurements.
- Signal: Every output channel needs to be defined as a play channel. Every channel ID or elevation (Variables) needs to be linked one of these play channel in Signal tab; can be used as a kind of translation between an elevation number and a channel number of the sound interface.
  FS to SPL offset allows using individual level offsets between channels.
  Important (at least for HRTF measurements): Every channel has a different latency, depending on the measurement system and hardware, and the exact distance between loudspeaker and microphone may vary slightly between measurements!), this is why the latency should be determined for every measurement. Details see in chapter: Copy latency latLIN to Settings/Signal/System Latency.
- Audio: Define your desired sampling rate and resolution.
- Procedure: Define the experiment type, for details and the differences see chapter Experiment Types (Measurement Methods).
- Variables: Define parameters that (can) vary within one experiment; for details see chapter Variables List (all experiment types).
- Constants: Define parameters that are constant within one experiment; for details see the sub-chapters “Parameters (Constants)” in Variables List (all experiment types).
- Tracker: only relevant if you use a tracking system
- ViWo: irrelevant in AMTatARI (display a virtual world)

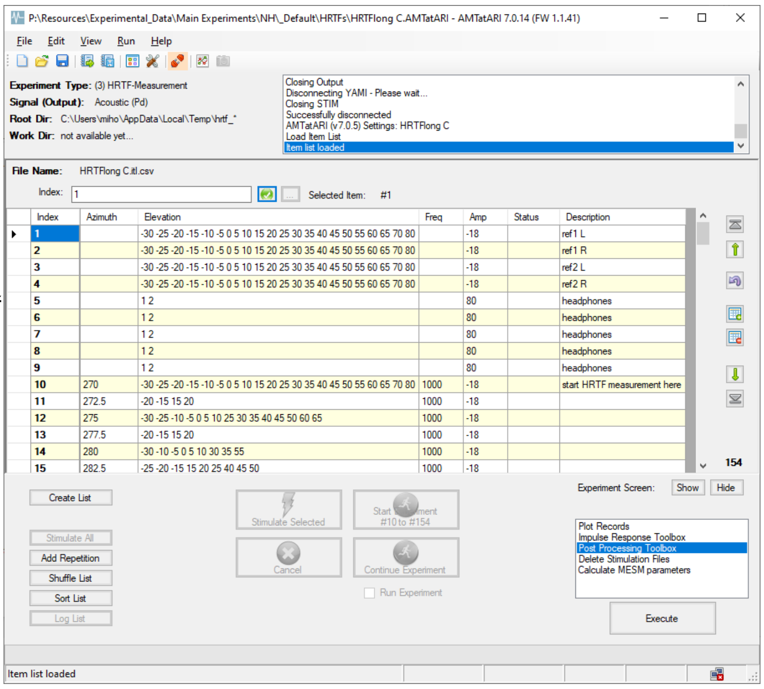
Item List

An item list, based on your Settings is created by pressing the button “Create List” on the main window. It is recommended to prepare the settings according to your measurement setup as good as you can. But you still can modify single cells or operate on rows or columns afterwards (see ExpSuite documentation for details). Item lists can also be saved and loaded (top menu: File → Load item list / Save item list as).

Columns of an item list

- Index: usually an incrementing number, starting with “1” (one). It is highly recommended to use unique indices for every item because the recorded file name for the corresponding item uses the index for its file name. Using equal index numbers in more than one item can cause recordings being overwritten.

- Azimuth: If an azimuth defined for the current item and a turntable is used then the turntable will be rotated to the defined position before the stimulus is played.
  The azimuth value is also used for the post processing and stored in the meta data (SOFA files) so it can make sense to define values although no turntable is used.
- Elevation / Channel ID: Once the position is reached (azimuth), all the defined channel IDs are being played. If elevations are used they are translated to the play channels, according the definitions in Settings → Signal tab.
- Frequency: The frequency of sinus tone in "Cosine" mode, or the frequency of the warning tone in case of tracker out of range. Not relevant for HRTF measurements if no tracker is used.


P:\Resources\Experimental_Data\Main Experiments\NH_Default\HRTFs\HRTFlong C.AMTatARI - AMTatARI 7.0.14 (FW 1.1.41)

File Edit View Run Help

Experiment Type: (3) HRTF-Measurement
Signal (Output): Acoustic (Pd)
Root Dir: C:\Users\miho\AppData\Local\Temp\hrtf_*
Work Dir: not available yet...

```
Closing Output
Disconnecting YAMI - Please wait...
Closing STIM
Successfully disconnected
AMTatARI (v7.0.5) Settings: HRTFlong C
Load Item List
Item list loaded
```

File Name: HRTFlong C.itl.csv
Index: 1 Selected Item: #1

Index	Azimuth	Elevation	Freq	Amp	Status	Description
1		-30 -25 -20 -15 -10 -5 0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 80		-18		ref1 L
2		-30 -25 -20 -15 -10 -5 0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 80		-18		ref1 R
3		-30 -25 -20 -15 -10 -5 0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 80		-18		ref2 L
4		-30 -25 -20 -15 -10 -5 0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 80		-18		ref2 R
5		1 2		80		headphones
6		1 2		80		headphones
7		1 2		80		headphones
8		1 2		80		headphones
9		1 2		80		headphones
10	270	-30 -25 -20 -15 -10 -5 0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 80	1000	-18		start HRTF measurement here
11	272.5	-20 -15 15 20	1000	-18		
12	275	-30 -25 -10 -5 0 5 10 25 30 35 40 45 50 60 65	1000	-18		
13	277.5	-20 -15 15 20	1000	-18		
14	280	-30 -10 -5 0 5 10 30 35 55	1000	-18		
15	282.5	-25 -20 -15 15 20 25 40 45 50	1000	-18		

154

Item list loaded

Figure 1: AMTatARI: Main window with item list (item 1 selected)

- Amplitude: Global amplitude (dB FS) for all channels. If the output channels use individual level offsets please define them in Settings → Signal tab.
- Status: Describing the status of the current item; examples are: recorded, IR calculated, IR plotted, error, and some others. The field can be overwritten by AMTatARI anytime so it should not be used for user notes. The column will be saved with the item list.
- Description: This field can (optionally) be used for user notes and will be saved in the item list, and in the SOFA object. The user has full control about this field, AMTatARI will not change anything.

Connect to Output

Once your options are defined and you prepared/loaded a settings file you can connect to output. "Connecting to output" establishes a connection (launched automatically to:

- Matlab
- pd
- a tracker (optionally)
- a turntable (optionally)

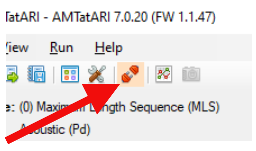

tatARI - AMTatARI 7.0.20 (FW 1.1.47)
iew Run Help
: (0) Maximum Length Sequence (MLS)
Acoustic (Pd)

Figure 2: Connect button

Connecting happens by clicking the red button in the main window, or by using the top menu: Run → Connect. Both options can be used to disconnect from output after an experiment.

After connecting to output you cannot change the options, and some of the settings anymore since some of these parameters are relevant while connecting.

The connection allows you use the provided features of Matlab (signal processing, plotting figures) and pd (audio playback and recording). It is required before you stimulate items!

Once you finished your measurement or results you should disconnect again before closing

AMTatARI.

Stimulate Item(s)

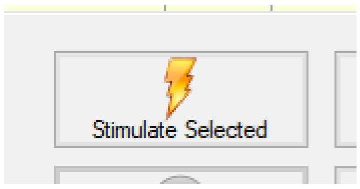


Figure 3: Stimulate Selected button

Once your options are defined, you prepared/loaded a settings file, you are connected to output, and you created an item list (button: Create List) you are ready to stimulate items. Stimulating items happens when you select any item (row) in the item list and use the button: Stimulate Selected. In case you use/enabled a turntable and an Azimuth value is set in your selected item list the turntable will rotate to that angle. Depending on the defined Elevation/Channel ID values the mapped loudspeakers (Settings → Signal tab) will stimulate your signals, and the defined ADCs (Settings → Variables tab) will record them.

The button "Start Experiment" will stimulate all items in the item list (or a part of it if a range is defined).

Experiment Types (Measurement Methods)

AMTatARI includes 4 different experiment types. The most common ones are "Multiple Exponential Sweeps" and "HRTF Measurement", so it might be that there is more focus on these procedures within this documentation. To change the experiment type please open the Settings (top menu: View → Settings → Procedure tab)

Maximum Length Sequence (MLS)

Goals

Measurement of Impulse Response (IR) of a system

Method

Play a MLS, record the system response, calculate IR

Parameters (Constants)

- MLS Order
- MLS Repetition
- Stimulus Length (is calculated from MLS Order, MLS Repetition and Sampling Rate)

Multiple Exponential Sweeps (ExpSweep)

The difference between Multiple Exponential Sweeps (ExpSweep) and HRTF Measurement is that the output channels are addressed as indices (channels) or elevations, respectively.

Goals

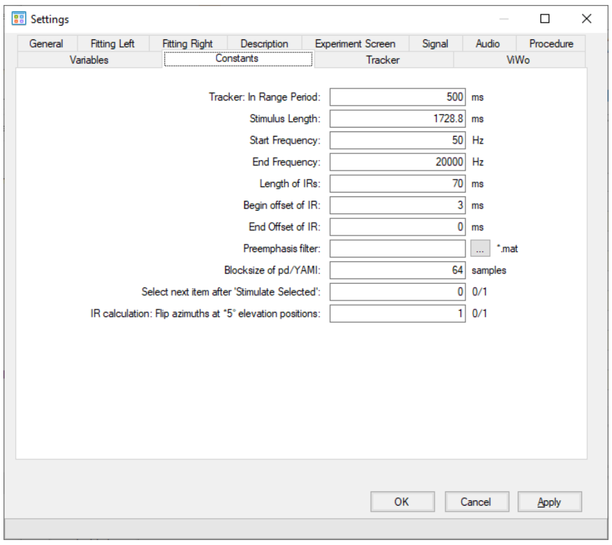
Measurement of IRs of multiple systems; measurement of a subject's HRTF with exponential sweeps (recommended if loudspeakers are addressed as indices and not elevations)

Method

Play exponential sweeps for every channel ID in current item, delayed by ISD (Inter Sweep

Distance), record the linear superposition of all responses, calculate the matrix of Irs.

Parameters (Constants)



Tracker: In Range Period:	500	ms
Stimulus Length:	1728.8	ms
Start Frequency:	50	Hz
End Frequency:	20000	Hz
Length of IRs:	70	ms
Begin offset of IR:	3	ms
End Offset of IR:	0	ms
Preemphasis filter:		*.mat
Blocksize of pd/YAMI:	64	samples
Select next item after 'Stimulate Selected':	0	0/1
IR calculation: Flip azimuths at *5° elevation positions:	1	0/1

Figure 4: Constants in experiment type "HRTF-Measurement"

- Tracker: In Range Period (only important if a tracking system is used): Sets the minimal period, in which the tracker sensor must remain in range before continuing measurement. Unit: ms
- Stimulus Length: Sets the length of stimulus (sweep). Unit: ms
- Start Frequency: Sets the start frequency of the sweep. Unit: Hz
- End Frequency: Sets the end frequency of the sweep. Unit: Hz
- IR Length: Sets the length of each IR. The total length of IR will be: begin offset + length + end offset. Unit: ms
- IR begin offset: Sets the additional offset to the begin of the IR in ms. The length of IR will be: begin offset + length + end offset. Unit: ms
- IR end offset: Sets the additional offset to the end of the IR in ms. The length of IR will be: begin offset + length + end offset. Unit: ms
- Preemphasis filter: Optional; sets the filename of preemphasis filters in order to preemphasize the exponential sweep with a filter. See AA_Preemphasize.m for details. Unit: *.mat file
- Blocksize od pd/YAMI: A lower block size reduces the latency but increases the risk of cracking sounds. Unit: samples
- Select next item after 'Stimulate Selected': Automatically select next item after stimulating an item in item list; 0: no, 1: yes.
- IR calculation: Flip azimuths at *5° elevation positions: When performing the calculation of the impulse responses, flip azimuth values by 180° if elevation is in *5° positions; 0: don't flip, 1: flip by 180°. Examples (if set to 1):
  - azi 0, ele 0 → no change
  - azi 0, ele <u>5</u>→ changed to azi 180, ele 5
  - azi 0, ele 10 → no change
  - azi 0, ele 1<u>5</u> → changed to azi 180, ele 15

Relevant Parameters for HRTF Measurement

The definition of signals for HRTF measurements is highly dependent on the system (interfaces, amplifiers, loudspeakers, and in-ear microphones) and the room used for the measurement (room size-dependent reflections, reverberation time).

The following parameters are describing the signal parameters:

- Stimulus length (Settings → Constants)
- Start frequency (Settings → Constants)
- End frequency (Settings → Constants)

- Amplitude (Settings → Variables)
- ISD (Settings → Variables): Inter Sweep Distance

Cosine Oscillation

Goals

Measurement of THD and/or SINAD

Method

Play a stationary cosine tone, record the system response. Calculate THD or SINAD for given frequency.

Parameters

- Stimulus Length
- Frequency Span
- THD: theta
- SINAD: Notch filter order

HRTF Measurement

The difference between Multiple Exponential Sweeps (ExpSweep) and HRTF Measurement is that the output channels are addressed as indices (channels) or elevations, respectively.

Goals

Measurement of a subject's HRTF with exponential sweeps.

Method

Play exponential sweeps for every <u>elevation</u> in current item, delayed by ISD (Inter Sweep Distance), record the linear superposition of all responses, calculate the matrix of IRs.

Parameters (Constants)

- See Parameters (Constants) in Exponential Sweeps (ExpSweep)

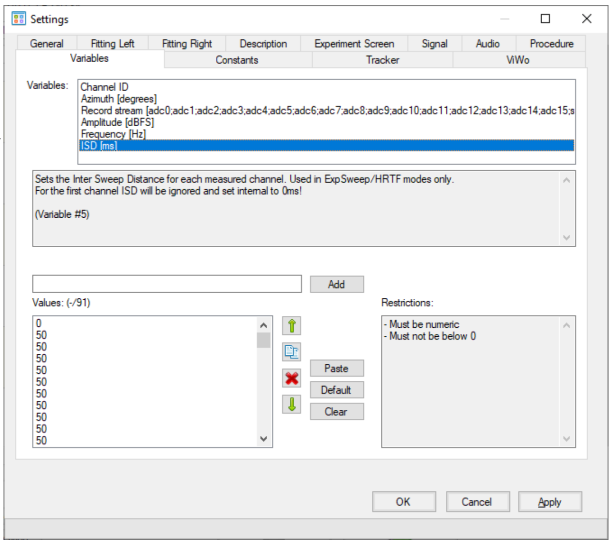
Relevant Parameters for HRTF Measurement

- See Relevant Parameters for HRTF Measurement in Exponential Sweeps (ExpSweep)

Variables List (all experiment types)

- Elevation / Channel ID: Which elevations or channel IDs are being recorded. Each elevation / ID must match a channel ID specified in the Signal tab (Settings).

- Azimuth (group): which azimuths are being recorded. Can also be grouped, syntax: begin resolution end (example: 0 2.5 45). Sets the azimuth angle of the turntable. Leave blank if you don't want to use turntable.
- Record streams: Sets the stream to record:
  - adcX: data from ADC, channel X (supported: adc0 to adc15; 16 channels)
  - synX: data from synthesizer unit X (adc0, adc1)
  - playX: data from WAV file, unit X (play0, play1)
  - silence: zero stream (no data) (silence)
- Amplitude: Sets the global signal amplitude, additional to the channel amplitudes set in Signal tab: 0: full scale → -100: no signal
- Frequency (Cosine Oscillator only): Sets the frequency of sinus tone in 'Cosine' mode and the frequency of the warning tone in case of tracker out of range.
- Inter Sweep Distance (ISD) (ExpSweep / HRTF measurement only): Sets the Inter Sweep Distance for each measured channel. Used in ExpSweep/HRTF modes only. For the first channel ISD will be ignored and set internal to 0ms. See chapter MESM Parameters (Multiple Exponential Sweep) how to define your ISDs.



Variables:
Channel ID
Azimuth [degrees]
Record stream [adc0;adc1;adc2;adc3;adc4;adc5;adc6;adc7;adc8;adc9;adc10;adc11;adc12;adc13;adc14;adc15;s
Amplitude [dBFS]
Frequency [Hz]
ISD [ms]

Sets the Inter Sweep Distance for each measured channel. Used in ExpSweep/HRTF modes only.
For the first channel ISD will be ignored and set internal to 0ms!

(Variable #5)

Values: (-/91)
0
50
50
50
50
50
50
50
50
50
50

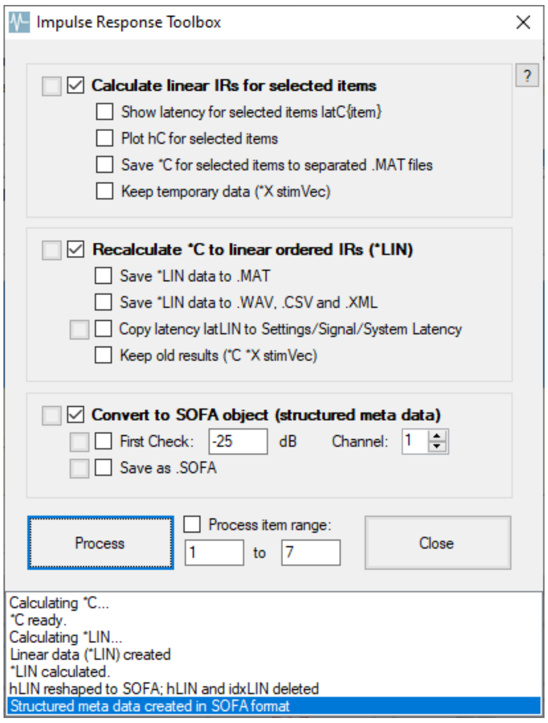
Restrictions:
- Must be numeric
- Must not be below 0

Figure 5: Variables in AMTatARI

Impulse Response Toolbox

The GUI to run impulse response (IR) operations can be launched by executing the function “Impulse Response Toolbox” right at the bottom of the main window, in the Results menu. The IR Toolbox is used to process your measured data, in order to calculate further parameters, save processed data (SOFA) or prepare your data for post processing.

The GUI consists of some sections: On the top operations for linear impulse responses can be selected; below operations for ordered impulse responses can be selected, and below some functions for SOFA objects are available. Usually the items selected in the item list are used for the calculations. The check box “Process item range” can overwrite the range being used for a calculation (makes sense after a very long measurement, and a large range of items would have needed to be selected. On the bottom a (scrollable) history of previously performed functions is displayed.



✓ Calculate linear IRs for selected items
☐ Show latency for selected items latC{item}
☐ Plot hC for selected items
☐ Save *C for selected items to separated .MAT files
☐ Keep temporary data (*X stimVec)

✓ Recalculate *C to linear ordered IRs (*LIN)
☐ Save *LIN data to .MAT
☐ Save *LIN data to .WAV, .CSV and .XML
☐ Copy latency latLIN to Settings/Signal/System Latency
☐ Keep old results (*C *X stimVec)

✓ Convert to SOFA object (structured meta data)
☐ First Check: -25 dB Channel: 1
☐ Save as .SOFA

☐ Process item range: 1 to 7

```
Calculating *C...
*C ready.
Calculating *LIN...
Linear data (*LIN) created
*LIN calculated.
hLIN reshaped to SOFA; hLIN and idxLIN deleted
Structured meta data created in SOFA format
```

Figure 6: Impulse Response Toolbox

Functions can be performed by clicking on the individual small buttons left to the check boxes, or by checking the desired check boxes on this GUI and clicking on “Process”. In the second case the selected functions are performed from the top to the bottom. Functions that are tabbed to the right require that its title function has been performed, or that its output is still in the workspace. For example: “Show latency for selected items latC{item}” requires that “Calculate linear IRs for selected items” has been performed first.

Tooltips (hoover mouse over text) are available for some of these functions. The question mark on the top right corner opens the documentation folder.

The following sub-chapters describe some variables and the most important functions more detailed.

Definitions of used variables

Some of the following definitions might be used in this chapter of the documentation:

- `REC`: number of recorded streams (eg. 2 for a binaural recording of HRTFs)
- `azi`: azimuth of a system (from the item list)
- `ele`: elevation or channel ID of a system (from Item List or Settings/Signal)
- `channel`: play channel of a system (from Settings/Signal)
- `ibegQ`: first item selected (in item list) for current operation
- `iendQ`: last item selected (in item list) for current operation
- `item`: item selected (item list), run time variable for current operation `ibegQ<=item<=iendQ`
- `itemNR`: number of selected items, data available
- `*X`: temporay variables
- `hM`: synonym for a structured meta data or a SOFA object, used in AMTatARI.

Calculate linear IRs for selected items (*C)

- Matlab script: `CalcSweepToIR`
- output: Matlab variables `*C`:
  - `hC{item}`: linear IRs in linear order
  - `posC{item}`: position of a system, format: [`azi ele channel`]
  - `idxC{item}`: index of a system in `hC{item}`, format: [“first_sample” “last_sample”]
  - `latC{item}`: latency of a system, relative to Settings/Signal/”System Latency”, in samples
  - `ampC{item}`: amplitude of the excitation signal for a system, in dB PD
  - `aziC{item}`: azimuth(s) of current item (item list)
  - `descC{item}`: description of current item (item list)
  - `freqC{item}`: frequency of current item (item list)

Show latency for selected items latC{item}

- `latX = latC{item}`
- the relative latency is converted to ms and displayed in a result form

Plot hC for selected items

Plot hC in Matlab.

Save *C for selected items to separated .MAT files

Save hLIN, latLIN, idxLIN, itemLIN, posLIN, ampLIN to Matlab file. One file per item will be created and saved to work directory.

Keep temporary data (*X stimVec)

If checked, all temporary Matlab variables *X and stimVec will not be deleted after processing. It is not recommended to be checked because it can cause relic data. This function can be used for debug purpose.

*Recalculate *C to linear ordered IRs (*LIN)*

- Matlab Script: `AA_CalcLinearMatrix` (using `ibegQ` and `iendQ`) is performed
- output: Matlab variables `*Q` and Matlab variables `*LIN`

hLIN

- size: [“sum of lengths of all IRs” `REC`]
- data of all IRs from `hC{item}`, in linear order

idxLIN

- Contains index of begin and end of an IR in `hLIN`
- size: [“number of IRs” `2`]
  - column 1: Index of begin of IR in `hLIN`
  - column 2: Index of end of IR in `hLIN`

IRnrLIN

- Contains number of IRs in one Item
- size: [`itemNR 1`]

itemnrLIN

- Usually the index number of item in item list. Empty items might be skipped.
- size: [`itemNR 1`]
- For all items containing data `itemnrLIN` contains a vector: `ibegQ:iendQ`

itemidxLIN

- Item indices of all IRs
- size: [“number of IRs” `1`]

latLIN

- Latency time of impulse responses, relative to: Settings/Signal → System Latency, in samples
- size: [“number if IRs” `1`]

posLIN

- Contains positions of source(in °) of every impulse response
- size: [“number of IRs” `7`]
  - column 1: `azimuth` (0° to +359.9°)
  - column 2: `elevation` (-90° to +90°)
  - column 3: `channel` (number)
  - column 4: `azimuth` (-90° to +90°)
  - column 5: `elevation` (-90° to +270°)
  - column 6: `lateral angle` (-90° to +90°)
  - column 7: `polar angle` (-90° to +270°)
- Defining elevation and channel for every impulse response allows mapping of impulse responses to channels (important for equalization).

ampLIN

- amplitude of the excitation signal for an item in dB PD
- size: `[itemNR 1]`

aziLIN

- azimuth of an item
- size: `[itemNR 1]`

descLIN

- description of an item (item list)
- size: `[1 itemNR]` (cell array)

freqLIN

- frequency of an item (item list)
- size: `[itemNR 1]`

Useful commands:

- `hM=reshape(hLIN,idxLIN(1,2),[],REC);` addressing all IRs in matrix form `hM`, size: ["length of one IR" "number of IRs" "number of channes"]
- `mn_plot(etc(double(hM)));` Dia-show of all IRs in `hM` (requires: MN_Toolbox)
- `hITEM=hM(idxLIN(sum(itemnrLIN(1:item-1))+1,1):idxLIN(sum(itemnrLIN(1:item)),2))';` resulting: IRs for `item`
- `itemnrLIN(find(cumsum(IRnrLIN)<IRnr,1,'last'))` providing number of items in item list for the `IRnr`th IR in `hLIN`, can be used to find out the amplitude from `ampLIN` for an item.

*Save *LIN data to .MAT*

Usually the data export happens at a later step of processing. This export can be used for debug purposes, or to import this step to a 3rd party application.

- Save `hLIN`, `idxLIN`, `IRnrLIN`, `itemnrLIN`, `latLIN`, `posLIN`, `ampLIN`, `stimPar` to `ID_LIN.mat` file (for Matlab for example)

*Save *LIN data to .WAV, .CSV and .XML*

Usually the data export happens at a later step of processing. This export can be used for debug purposes, or to import this step to a 3rd party application. These parameters are adapted to the STX format (Sound Tools Extended).

- Save normalized hLIN to ID_ALL_IR.wav, resolution and sampling rate are equivalent to Settings/Audio/...
- Save posLIN to ID_ALL_pos.csv
- Save idxLIN to ID_ALL_idx.csv
- Save following parameters as metadata to ID_ALL_IR.wav.xml using STX (Sound Tools Extended) style sheet: P, (begin of an IR), L (length of an IR), CH (always 0), azimuth, elevation, latency, amplitude, `itemnr`, DACchannel (=`channel`)

Copy latency latLIN to Settings/Signal/System Latency

The goal is to set the correct latency for each channel. The latency is correct when the IRs (time domain) begin at the same time for all channels. This can be visualized by plotting all IRs in the time domain – the beginning peaks must overlap at the same time.

The procedure of setting the correct latency is performed once for the measurement series and can be adjusted after the measurements. In optimal case the latency is set after a reference measurement, before the start of the entire measurement series.

The procedure works as follows:

- In settings, set the "Begin Offset of IR" constant to zero.
- Set the latency to a common value for all channels. The value must be chosen, so that you see the IRs in the time domain plot. It will be little bit higher than twice the latency of the sound interface you use. To set the latency to a common value:
  - Calculate the hC, reshape to hLIN
  - start "Copy latency latLIN to Settings", set the option "set to constant value", overwrite the suggestion of the input the value with your estimation of the latency and click "OK".
- Now calculate hC, reshape to hLIN and to SOFA object (structured meta data, hM) and plot the IRs in the time domain. You should see the IRs. They may be spread a little bit along the time axis, because you use a common latency for all channels. Now let's change that
- Start "Copy latency latLIN to Settings" and set the option "copy values from hLIN". Click "OK".
- Change the constant "Begin Offset of IR" to something like 3 ms. This determines how much of the IR you want to have before the highest peak in the IR.
- Calculate hC, reshape to hLIN and to SOFA object (structured meta data, hM) and plot the IRs in the time domain. Now, the IRs should begin exactly at the same time, particularly at the time defined by "Begin Offset of IR".
- Don't forget to save the settings!

*Keep old results (*C *X stimVec)*

If checked, all temporary Matlab variables *C *X and stimVec will not be deleted after processing. It is not recommended to be checked because it can cause relic data. This function can be used for debug purpose.

Convert to SOFA object (structured meta data; internal name: hM)

- Structure `hM` is created from matrix `hLIN`: `hM=reshape(hLIN,idxLIN(1,2),[],REC);`
- `hLIN` and `idxLIN` are deleted because all other `*LIN` belong to `hM` now
- `meta` contains meta data belonging to `hM: pos` (equivalent to `posLIN`), `itemidx` (`itemidxLIN`), `lat` (`latLIN`), `amp` (`ampLIN`)

First Check

This function can be used as a quick check for calculated data. All measurements for the selected channel are checked if they reach a specific level in dB FS. The two input values (level, channel) can be adapted to individual measurements.

The output of this function is a message box with a list of measurements not reaching the input level.

Save as .SOFA

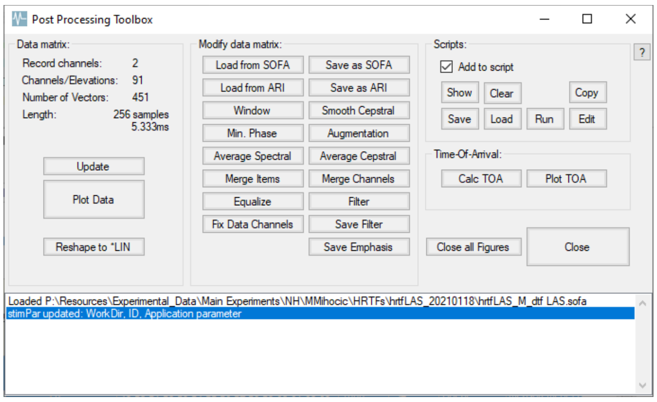
This function saves the data in the SOFA conventions format. In the following dialogue windows the file name can be entered, and the SOFA convention can be selected. Files are saved according to the scheme: [ID]_M_[name].sofa (ID: defined in Settings → Description, see part Description in chapter Settings).

The list of SOFA conventions being displayed depends on the files in the application's Matlab folder (default: C:\Program Files (x86)\ExpSuite\AMTatARI\MATLAB). All files in the scheme: AA_SOFAsave[any convention name].m are displayed. Users can prepare these files adapted to their measurement system. It is recommended to have a look at the files already existing, create a copy, and adapt the containing parameters to the actual measurement system. The files included in the AMTatARI installation are prepared for measurements at the Acoustics Research Institute in Vienna.

Post Processing Toolbox

The GUI for the Post Processing Toolbox can be launched by executing the function "Post Processing Toolbox" right at the bottom of the main window, in the Results menu.

For details about the signal processing steps please see the following chapter Post Processing.



Data matrix:
Record channels: 2
Channels/Elevations: 91
Number of Vectors: 451
Length: 256 samples 5.333ms

Modify data matrix:

Scripts:
☑ Add to script

Time-Of-Arrival:

Loaded P:\Resources\Experimental_Data\Main Experiments\NH\MMihocic\HRTFs\hrtfLAS_20210118\hrtfLAS_M_dtf LAS.sofa
stimPar updated: WorkDir, ID, Application parameter

Figure 7: Post Processing Toolbox

The following sub-chapters describe the functions available on the Post Processing Toolbox window (for screenshot see Figure 7: Post Processing Toolbox). In general the window consists of the following sections: Left there is a summary of the current data matrix (Matlab workspace); in the center there are buttons performing operations to this data matrix; on the right there is the possibility to operate script-based in order to make similar measurements and their post processing easier. On the bottom a (scrollable) history of previously performed functions is displayed.

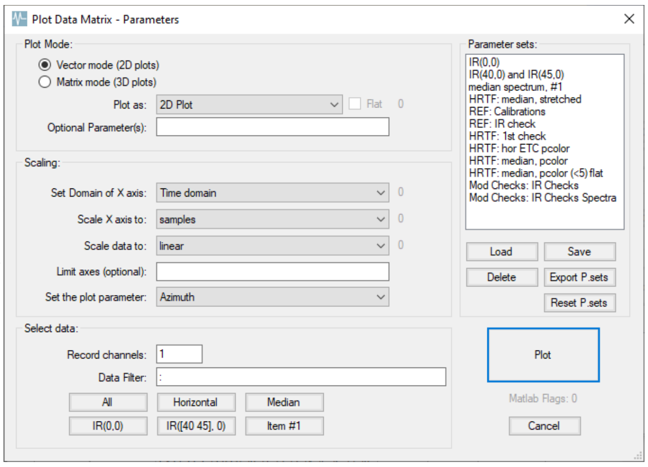
When describing functions, `hM` is used as a synonym for the data matrix, as it is used internally in AMTatARI and Matlab. The data matrix can be stored as SOFA object.

Update

Button can be used to updated the displayed parameters of the data matrix (record channels, number of channels/elevations, number of vectors, length). Can be useful if Matlab workspace was modified somewhere else.

Plot Data

This buttons opens another window (see Figure 8: Plot Data Matrix window). In the left side of the window several plotting parameters can be selected. There is also the possibility to save and load some parameter sets (right, top of the window), in order to use same plotting parameters for similar data sets. The list of parameter sets is saved locally on every computer in the ProgramData folder (where the options file is also stored: %ProgramData%\Expsuite\AMTatARI), as a .csv file. It can be exported to the working directory by using the corresponding button.



Plot Mode:
● Vector mode (2D plots)
○ Matrix mode (3D plots)
Plot as: 2D Plot ☐ Flat 0
Optional Parameter(s):

Scaling:
Set Domain of X axis: Time domain 0
Scale X axis to: samples 0
Scale data to: linear 0
Limit axes (optional):
Set the plot parameter: Azimuth

Select data:
Record channels: 1
Data Filter: :

Parameter sets:
IR(0,0)
IR(40,0) and IR(45,0)
median spectrum, #1
HRTF: median, stretched
REF: Calibrations
REF: IR check
HRTF: 1st check
HRTF: hor ETC pcolor
HRTF: median, pcolor
HRTF: median, pcolor (<5) flat
Mod Checks: IR Checks
Mod Checks: IR Checks Spectra

Matlab Flags: 0

Figure 8: Plot Data Matrix window

Most of the buttons and fields on this form should be self-explaining. In general, these parameters are used as input values for the Matlab script file FW_ShowStimulus.m.

Reshape to *LIN

hM (SOFA object) is converted to linear matrix hLIN (see chapter hLIN). idxLIN is re-calculated.

Load from SOFA

Load a SOFA object from work directory. The available files have to be named according to the scheme: [ID]_M_[name].sofa (ID: defined in Settings → Description, see part Description in chapter Settings). They have to fulfill the requirements of the SOFA conventions format.

Save as SOFA

This function saves the current data matrix in the SOFA conventions format. In the following dialogue windows the file name can be entered, and the SOFA convention can be selected. Files are saved according to the scheme: [ID]_M_[name].sofa (ID: defined in Settings → Description, see part Description in chapter Settings).

The list of SOFA conventions being displayed depends on the files in the application's Matlab folder (default: C:\Program Files (x86)\ExpSuite\AMTatARI\MATLAB). All files in the scheme: AA_SOFAsave[any convention name].m are displayed. Users can prepare these files adapted to their measurement system. It is recommended to have a look at the files already existing, create a copy, and adapt the containing parameters to the actual measurement system. The files included in the AMTatARI installation are prepared for measurements at the Acoustics Research Institute in Vienna.

Load from ARI

This function loads files according to the (outdated) ARI format, stored as .mat files, according to the scheme: [ID]_M_[name].mat (ID: defined in Settings → Description, see part Description in chapter Settings).

- Input: `postfix`
- Loading file `ID_M_postfix.mat`
- File should contain `hM`, `meta` and `stimPar`. When loading a .mat file in the even older ARI format (`IRnrLIN`, `itemnrLIN`, `latLIN`, `posLIN`, `ampLIN`), it is automatically converted to new structure.

Save as ARI

This function saves files according to the (outdated) ARI format, stored as .mat files, according to the scheme: [ID]_M_[name].mat (ID: defined in Settings → Description, see part Description in chapter Settings).

- Input: `postfix`
- saving `hM`, `meta` and `stimPar` to file `ID_M_postfix`.m (structured meta data)

Window

Window the current data matrix according the following parameters:

- Begin of window (offset) [ms]
- End of window (offset + length) [ms]

- Fade in [ms]
- Fade out [ms]

These parameters can be entered in four subsequent dialogues. The used Matlab script is the file "AA_Window.m" in AMTatARI's Matlab directory.

Smooth Cepstral

Smooth the spectrum with a cepstral filter according the parameter: Low pass quefrency of the filter[ms]. Matlab script "AA_SmoothCepstral.m" is used to perform cepstral smoothing. For signal processing details see sub-chapter Reference Measurement in chapter Post Processing, especially: Figure 11: Cepstrally-smoothed reference data (low-pass lifter with a cut-off quefrency of 3.8 ms). Left panel: normalized IRs (in dB). Right panel: normalized amplitude spectra (in dB). The different colors represent different audio channels.

Min. Phase

Calculates the minimum phase signals. Matlab script "AA_MinimalPhase.m" is used to perform the calculation. For signal processing details see sub-chapter Reference Measurement in chapter Post Processing, especially: Figure 12: Post-processed reference data. Left panel: normalized IRs (in dB). Right panel: amplitude spectra (in dB). The different colors show different loudspeakers.

Augmentation

Augments the current data matrix (add energy to low frequency region), using the Matlab script "AA_Augment.m", according the following input paramters:

- Low frequency [Hz]
- High frequency [Hz]
- Augmentation amplitude of lower frequencies [dB]

Augmentation can produce some problems in HRTFs. The lateral error increases in localization tasks.

Average Spectral

Spectral averages are calculated using the Matlab file "AA_AverageSpecstral.m", according to the input items of the filters for channels #1 and #2.

Average Cepstral

Cepstral averages are calculated using the Matlab file "AA_AverageCepstral.m". For signal processing details see sub-chapter Reference Measurement in chapter Directional Transfer Functions (DTFs).

Merge Items

Merge items in data matrix. It can be used to merge two items, where each of them contains two recorded channels but only one interesting channel each. Merging happens using Matlab script „AA_MergeItems.m", according the following parameters:

- item with record channel 1
- item with record channel 2

Merge Channels

Merge items to one record channel (e.g. preemphasis filters), or merge items to one play channel (e.g. headphones analysis). Merging happens using Matlab script „AA_MergeChannels.m“.

- `AA_MergeChannels`: `hM` is “diagonalized” for all `REC` “: a new `hM` is created, containing one `ele` but all `REC`s, every `REC` can be copied from a different `ele`.
  Example: `hM` contains `ele`s 1, 3 and 14 with 2 `REC`s. The new `hM` should use `REC=1` from `ele=`1 and `REC=`2 from `ele=`14. The map vector is: `[1 14]`.
- MergeChannels is used for headphones equalization where more than 2 `ele`s on 2 different `REC`s the left and right channels are measured, and a filter matrix should be created. `posLIN = [NaN NaN NaN]`.
- MergeChannels can also be used to filter a single `ele` from a larger `hM`. Warning: This step is irreversible.

Equalize

A target is defined. A filter is loaded and de-convoluted with the target, so that the filter being filtered with the inverse filter results in the target again. `hM` is convoluted with the inverse filter, and also equalized. The corresponding equalization function is used from column `channel` in `hM`, and from the filter. If a channel cannot be found, or the filter contains the channel NaN then the normalization uses this filter (see Merge Channels). Equalization happens using the Matlab script „AA_Equalize.m“. For details see sub-chapter Head Related Transfer Functions (HRTF) in chapter Post Processing. The data matrix is equalized to the following input parameters:

- target amplitude [dB FS]
- start frequency of target [Hz]
- end frequency of target [Hz]
- length of resulting impulse response [ms]
- postfix of filter, which filter file being used for the equalization: one or more `postfix` for files `ID_FILT_postfix.m`

Filter

Filter data matrix, using the Matlab script “AA_Filter.m“, according to the following input parameters:

- length of impulse response [ms]
- postfix of filter, which filter file being used for the equalization: one or more `postfix` for files `ID_FILT_postfix.m`

Fix Data Channels

Fix data channels if a singleton dimension was truncated by Matlab. This happens sometimes if a three-dimensional matrix is reduced to singletons in one dimension. AMTatARI checks if this happened in relevant operations. If everything works well this function does not need to be performed manually.

Save Filter

This function is used to sage filter files, using the Matlab file “AA_CalcSaveFILT.m”.

- `AA_CalcSaveFILT`: `hM` is saved as a filter matrix.
- Input for result: `postfix`
- Result: `hFILT` and `chFILT` in file `ID_FILT_postfix.m`. `hFILT` is 3-dimensional (length of filter, number of channels, `REC`).

Save Emphasis

- `AA_CalcSaveEMP`: Target is defined; an inverse filter is calculated from `hM`, so `hM` filtered with the inverse filter results in the target. Just one `REC` channel of `hM` is used.
- Input for target: Amplitude, start and end frequency
- Input for hM: channel REC, filter length, FadeIn, FadeOut, Offset
- Input for result: `postfix`
- Result: `hEMP` and `chEMP` in file `ID_EMP_postfix.m`

Post Processing

Chapter imported from file *Post Processing Toolbox.odt* (Majdak, 2007, revised 2011). In the following sub-chapters you can find the signal processing details from the post processing toolbox.

Abstract

Presenting sounds in virtual environments requires filtering of free field signals with head related transfer functions (HRTF). The HRTFs describe the filtering effects of the incoming sound by the pinna, head, and torso, measured in the ear canal. Individual HRTFs are required for a high-quality reproduction of spatialized sounds. The measurement of HRTFs is a system identification procedure for many spatial positions and results in a set of raw measured data. The measured data are disturbed by the effect of the equipment and the room, both of which have to be reduced to obtain proper HRTFs. Furthermore, the post-processed HRTFs can be used to calculate directional transfer functions (DTF), which emphasize the direction-dependent portions of the HRTFs. These calculations are done in a post-processing procedure described in this document.

General

For each position in space a binaural pair of impulse responses (IRs), the head related impulse responses (HRIRs) is measured. The Fourier transform of an HRIR results in an HRTF. Each HRTF is measured using the multiple exponential sweep method (Majdak, Balazs and Laback, 2007). The measured HRTFs contain not only the spatial information about the signal but also artifacts. One kind of artifacts is caused by the audio equipment, which consists of loudspeakers, power amplifiers, in-ear-microphones, and mic-preamplifiers. The equipment artifacts are different for the left and right microphones, and for the different loudspeakers, or in general, the different audio channels[1]. Another kind of artifacts is caused by the room or sound chamber where the measurements are performed. The room reflections interfere with the measured HRIRs and depend on the positions of the speakers and microphones. Thus, the room effect differs for each measured position.
The goal of the post processing is to reduce the differences in the measurement performed via different audio channels and to reduce the effects of the room and equipment. A reference measurement captures the effects of the equipment and the room helps to reduce their effect in the HRTFs. Additionally, the direction-dependent part of HRTFs is emphasized by calculating the directional transfer functions (DTF). Finally, the HRIRs are windowed to shorter IRs for fast filtering with free-field signals.

Reference Measurement

A reference measurement is performed at the beginning of the HRTF measurement to capture the effects of the room and the equipment. In the reference measurement the microphones are placed in the center of the loudspeaker arc, which is assumed to be the center of the interaural axis. The effect of the loudspeaker includes the corresponding effects of the digital-analog converter and the power amplifier. The effect of the microphone includes the corresponding effects of the pre-amplifier and

1 In our setup, each audio channel represents one vertical position.

the analog-digital converter. All the effects can be further represented by the equipment effect for one audio channel. Thus, the raw reference data describe (in the frequency domain):

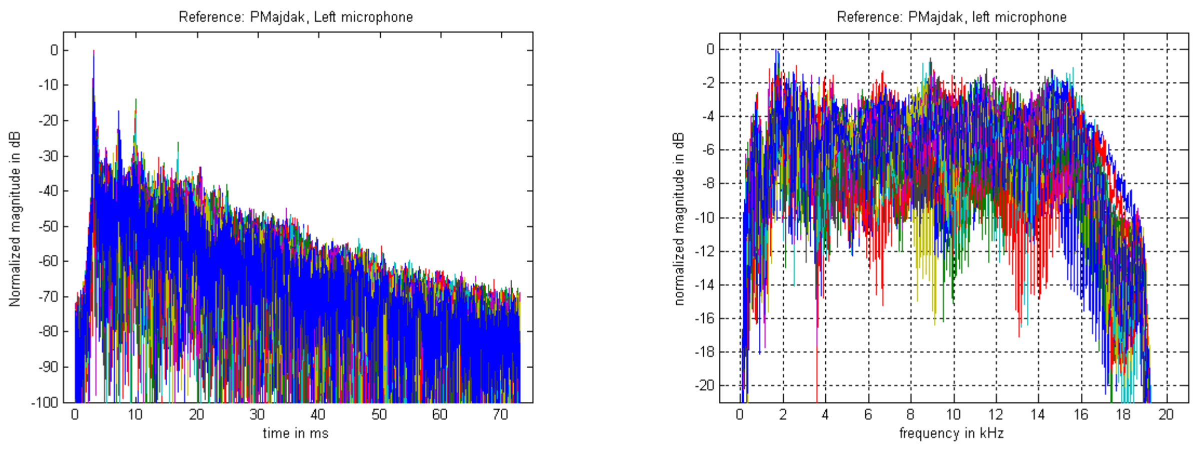


Figure 9: Raw reference data from the reference measurement for the left microphone and 22 audio channels. Left panel: normalized IRs (in dB). Right panel: normalized amplitude spectra (in dB). The different colors show the 22 different audio channels.

$$R_{raw} = L \cdot C_{ref} \cdot M = E \cdot C_{ref}, \qquad (1)$$

where R_{raw} corresponds to the raw recorded reference data, L is the transfer function of the loudspeaker, C_{ref} is the transfer function of the room (chamber), M is the transfer function of the microphone, and E is the transfer function of the equipment. Figure 9 shows examples of the raw reference data for 22 audio channels[2].

Now, one could attempt to remove the equipment effect by equalizing the raw HRTFs with the reference data and then reduce the effect of the room by smoothing the spectra. This method would work well if the room effect were equal in both the reference data and raw HRTFs. Unfortunately, the microphone position of the reference measurement is in the center of the loudspeaker arc, while in the HRTF measurements, the microphone position varies depending on the direction of each particular HRTF. Thus, equalization of the HRTFs with the reference data does not remove the room effect, but combines the different effects of the room measured at the various positions and may increase the fluctuations in the amplitude spectra. Furthermore, a subsequent spectral smoothing procedure may fail – yielding in either removing too many interesting parts of HRTFs (too much smoothing) or not sufficiently reducing the fluctuations (too little smoothing).

One possible solution could be to measure the reference data for every spatial position. However, since the position of the microphones varies for each subject and each HRTF position, it would be necessary to adapt the reference measurements to the particular subject and repeat them for all positions. This approach would result in much more data and much more complex post-processing, making the measurement of the HRTFs much more difficult.

Thus, another way of removing the room effect is proposed, basing on the following assumptions:

- The delay between the direct sound and the most prominent reflections is assumed to be very long. This yields in a comb filter effect, where the amplitude spectrum has very fast fluctuations.
- Multiple reflections are assumed, causing the amplitude spectrum fluctuations to be very stochastic.
- The amplitude spectrum of the equipment is assumed to be very smooth, considering a high-

2 The raw reference data are saved in “hrtf_M_ref1.mat”. Use “hrtf_CalcREF” to perform all calculations in this chapter at once.

quality loudspeakers, microphones, and the corresponding amplifiers.

With these assumptions, the equipment amplitude spectrum are supposed to be smoother than the room amplitude spectrum, showing a structural difference between the room effect and the equipment effect, and providing a basis for separation the two effects. Transforming the raw reference data to cepstrum (Bogart et al., 1963), the smooth spectral components of the equipment appear around the quefrency of zero. The stronger fluctuating spectral components of the room caused by reflections appear as smeared peaks. We define the cut-off quefrency, c_l, which separates the equipment effect from the room effect in cepstrum. With c_l, the spectrum of the raw reference data can be smoothed reducing the room effect but not strongly affecting the equipment transfer functions.

In order to determine c_l, the transfer function of each equipment audio channel was measured without the room effect. This was realized by placing the loudspeaker and the microphone in the room in such a way that the room reflections occur after the main fade-out of the direct signal. Then, for each audio channel, the system identification procedure was applied and the cepstrum was calculated. The results for two typical channels are shown in Fig. 10. The first peak in the cepstrum, representing the first room reflection, was found to be at the quefrency of 5.25 ms for all channels. The most cepstra decayed within 4 ms. The stagnation of the decay, indicating no further

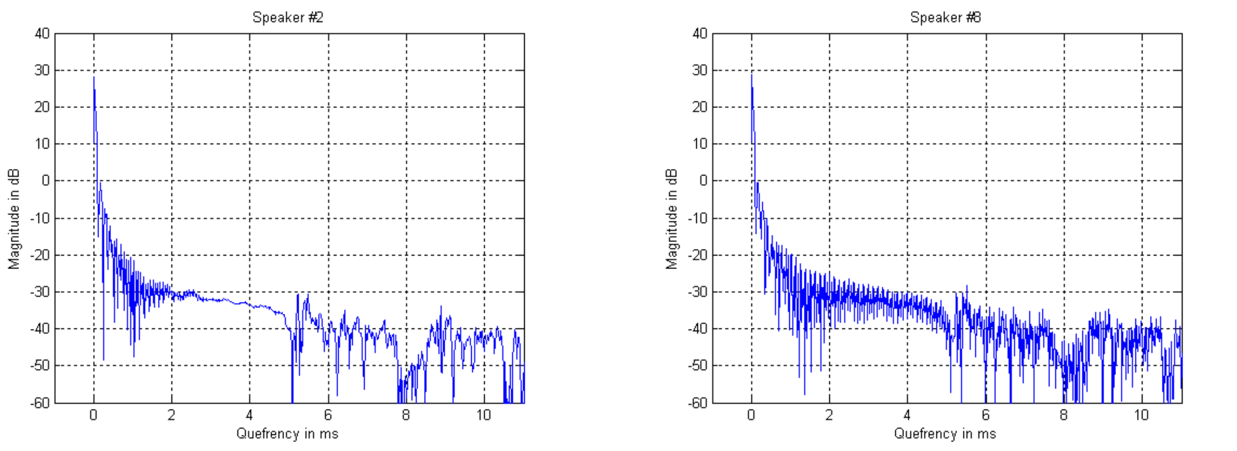


Figure 10: Cepstrum (in dB) for two typical audio channels without the reduced room effect (see text).

contribution of the equipment, begins between 3 and 4 ms, depending on the channel. For further calculations, we will use c_l=3.8 ms. Note that the estimation of c_l is required only at the very first build-up of the equipment and is not required at the HRTF measurements. The estimation of c_l should, however, be validated, when equipment components like loudspeakers change.

Before the HRTF measurement, the reference measurement is performed. With c_l, the raw reference data are cepstrally smoothed[1] using a low-pass lifter configured to c_l. This results in IRs, which decay quickly (within 7 ms down to -60 dB, see Fig. 11, left panel) compared to the raw reference data (decay within 50 ms, see Fig. 9, left panel). Also, the amplitude spectra are more smooth now (see Fig. 11, right panel). A Tukey window is applied to crop the region from 1.5 to 7 ms of the IRs[2]. The Tukey window is configured to a fade in of 0.5 ms and fade out of 1.5 ms. The resulting IRs have the length of 5.5 ms.

In each IR, the room effect has been drastically reduced. However, each IR represents a mixed-phase system, consisting of both the minimal and maximal phase components. For a further equalization, only minimal phase filters are of interest because equalizing a signal with filters which contain maximal phase components yields in unstable or uncausal results (Oppenheim and Schafer,

1 Use “AA_SmoothCepstral” to perform cepstral smoothing.
2 Use “AA_Window” to perform windowing and fading.

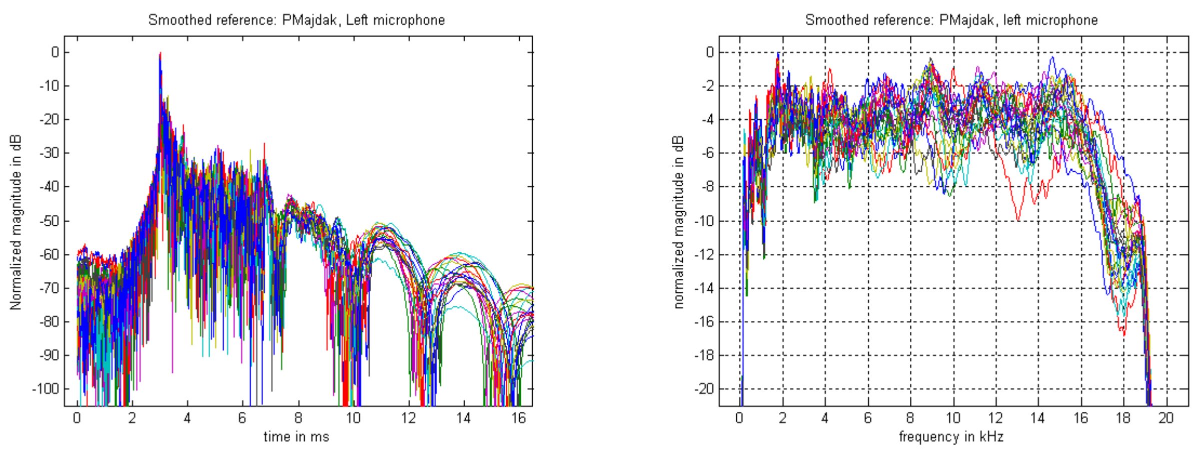


Figure 11: Cepstrally-smoothed reference data (low-pass lifter with a cut-off quefrency of 3.8 ms). Left panel: normalized IRs (in dB). Right panel: normalized amplitude spectra (in dB). The different colors represent different audio channels.

1998, pp. 288). Thus, in each IR, the phase spectrum is replaced by the minimal phase spectrum.[1] The minimal phase spectrum is calculated using the Hilbert transform (Oppenheim and Schafer, 1998, pp. 788).

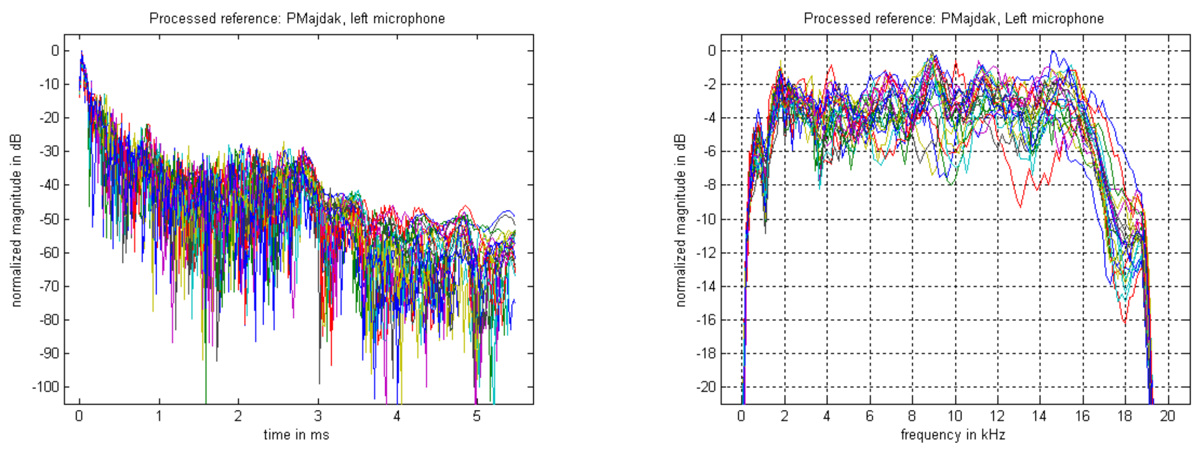


Figure 12: Post-processed reference data. Left panel: normalized IRs (in dB). Right panel: amplitude spectra (in dB). The different colors show different loudspeakers.

These post-processed reference data, R_{post} are shown in Fig. 12.[2] They show reduced room effect, contain the minimum-phase information about the equipment, and have the length of 5.333 ms. We assume that they represent the equipment effect:

$$R_{post} \approx E \tag{2}$$

and thus treat these data as equalization filters for further processing.[3]

Head Related Transfer Functions (HRTF)

The raw HRTFs can be represented in a similar way as the raw reference data:

$$H_{raw} = E \cdot C_{hrtf} \cdot H , \tag{3}$$

1 Use “AA_MinimalPhase” to replace the phase by the minimal phase.
2 The post-processed reference data are saved in “hrtf_M_ref1 processed.mat”.
3 The post-processed reference data are saved as equalization filters in “hrtf_FILT_ref1 processed.mat”. Generally use “AA_CalcSaveFILT” to save IRs as filters.

where H_{raw} corresponds to the recorded raw HRTFs, E is the transfer function of the equipment, C_{hrtf} is the transfer function of the room (chamber), and H is the actually desired HRTF. Figure 13 shows examples for raw HRTFs for the positions front (0°), left (90°), back (180°), and right (270°) in the horizontal plane.[1]
In order to remove both the room and the equipment effects, Eq. 3 is reformulated:

$$H=\frac{H_{raw}}{E}\cdot\frac{1}{C_{hrtf}}=\frac{H_{equ}}{C_{hrtf}} \tag{4}$$

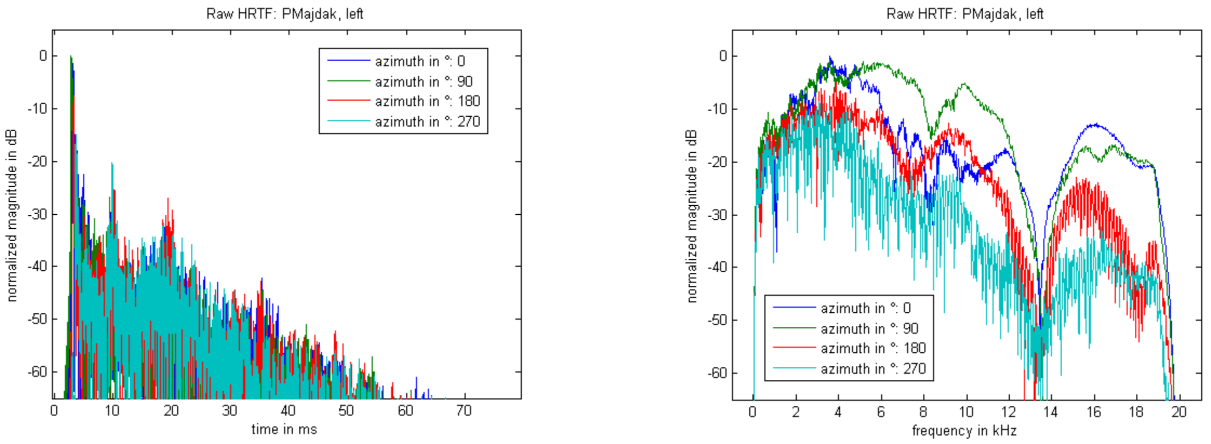


Figure 13: Raw HRTFs for the left ear. Left panel: normalized IRs. Right panel: normalized amplitude spectra. Different colors represent the HRTFs for position 0°, 90°, 180°, and 270° in the horizontal plane.

where H_{equ} represents the equipment-equalized HRTF.

In a first step, the equipment effect is reduced. We call this process equalization and H_{equ} is calculated from H_{raw} with the equalization filters from the reference measurement:

$$H_{equ}=\frac{H_{raw}}{E}\ . \tag{5}$$

Note that the post-processed reference data, R_{post} are assumed to represent the equalization filters ($R_{post}\approx E$).

The equalization is performed by dividing the complex spectrum of the raw HRIRs by the complex spectrum of equalization filters separately for each audio channel (Fielder, 2003).[2] In order to increase the frequency resolution, the raw HRIRs are zero-padded to 100 ms before Fourier transformation. The complex division is applied only within the frequency range where the equalization filters have much energy, which is from 300 Hz to 18 kHz (see Fig. 12). This avoids a regularization proposed by Fielder (2003). The amplitudes for frequencies below 300 Hz and above 18 kHz are set to zero. After division and inverse Fourier transformation, windowing to a length of 9 ms is applied. Figure 14 shows the resulting equalized HRTFs, H_{equ}.[3]

In the next step, in order to obtain H, the room effect, C_{hrtf} is reduced in H_{equ}. This is done by applying the cepstral smoothing procedure from the reference measurement to H_{equ}: First, H_{equ} is zero-padded to the length of 50 ms; then, it is cepstrally smoothed using a low-pass lifter configured to the cut-off quefrency c_l=3.8 ms. This yields in 50-ms long HRIRs, H.[4]

1 The raw HRTFs are saved in "hrtf_M_raw.mat". Use "hrtf_CalcREF" to perform all calculations in this chapter at once.
2 Use "AA_Equalize" to perform an equalization.
3 The equalized HRTFs are saved in "hrtf_M_equ.mat".
4 The long final HRTFs are saved in "hrtf_M_hrtf.mat".

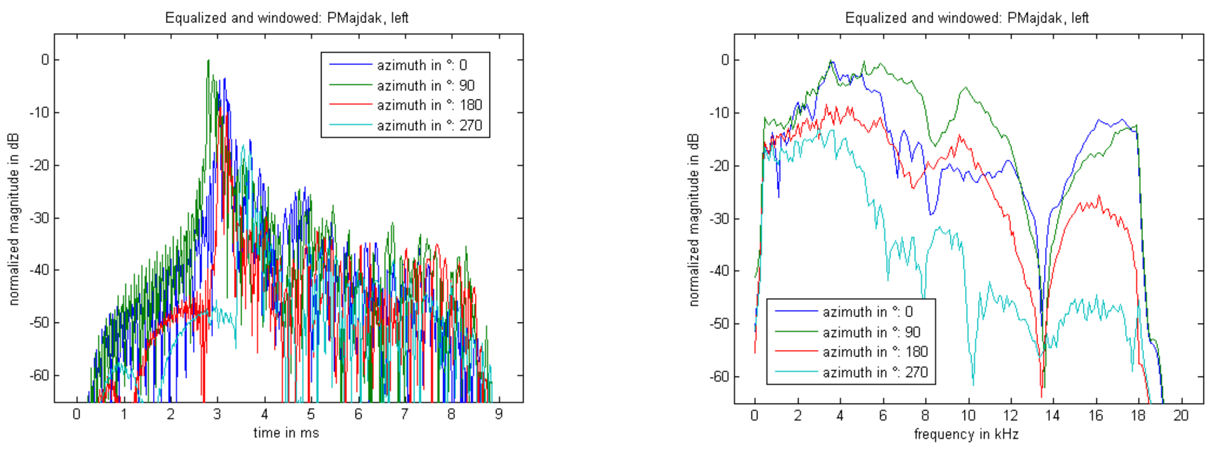


Figure 14: Equalized HRTFs for the left microphone. Left panel: normalized IRs. Right panel: normalized amplitude spectra. Different colors represent the HRTFs for position 0°, 90°, 180°, and 270° in the horizontal plane.

These HRIRs are further windowed to the length of 5.333 ms, which corresponds to 256 samples at the sampling frequency of 48 kHz. Figure 15 shows an example of four short HRTFs.[1]

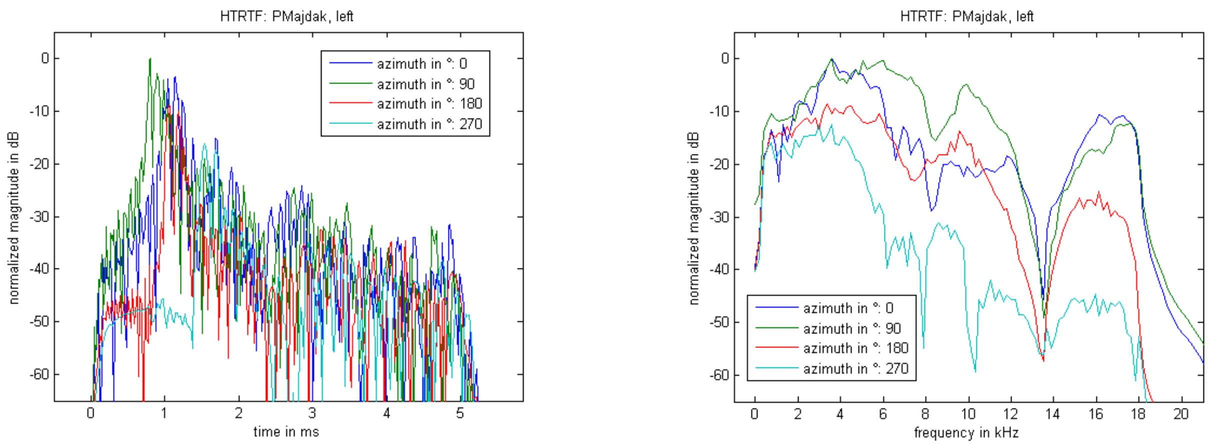


Figure 15: Post-processed HRTFs for the left ear. Left panel: normalized IRs. Right panel: normalized amplitude spectra. Different colors represent the HRTFs for position 0°, 90°, 180°, and 270° in the horizontal plane.

Now, in the post-processed HRTFs, the room and equipment effects are substantially reduced and the filters contain mostly the effects of the head, pinna, and torso. The HRIRs have the length of 5.333 ms and frequency range from 300 Hz to 18 kHz. Examples of the post-processed HRTFs for the horizontal and median plane is shown in the Figure 16.[2]

1 The short final HRTFs are saved in "hrtf_M_hrtf 256.mat".
2 Load "hrtf_M_hrtf 256.mat" and start "hrtf_ShowData" to get some nice plots of the HRTFs.

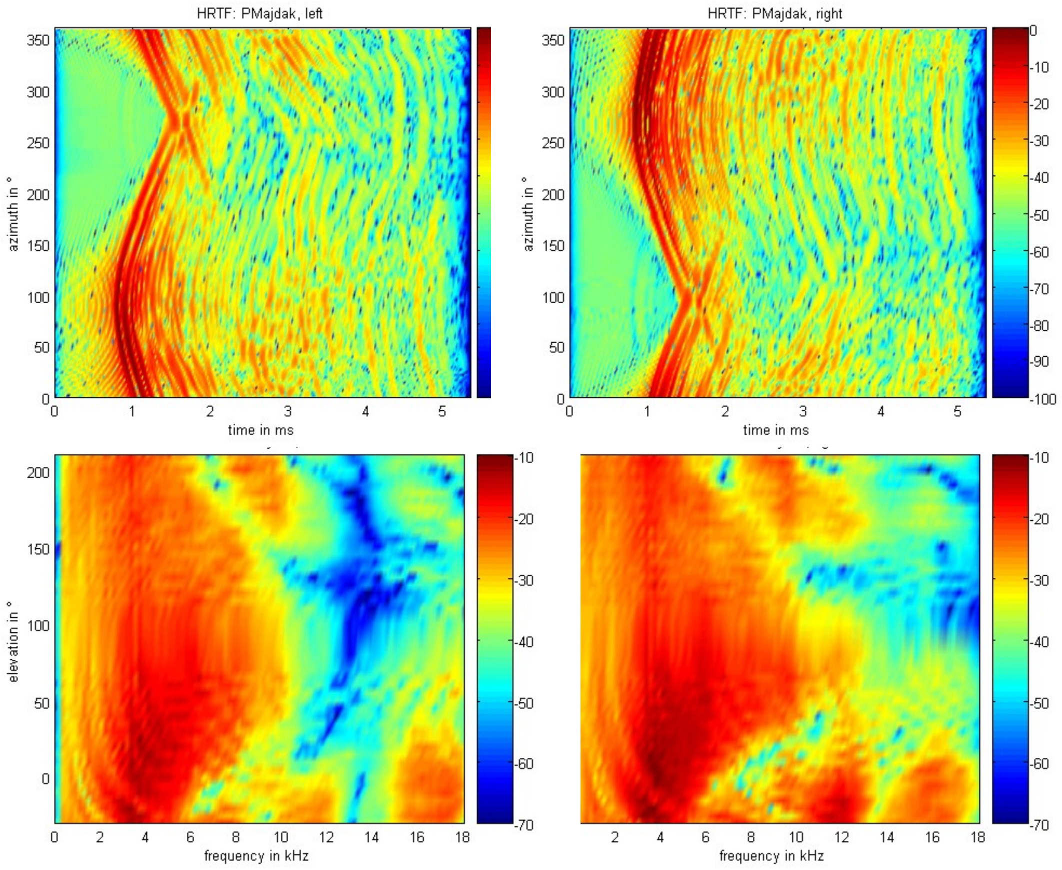


Figure 16: Post-processed HRTFs. Upper panels: normalized IRs in the horizontal plane. Lower panels: normalized amplitude spectra in the median plane. The colors correspond to the magnitude in dB. Left and right panels correspond to the left and right ear, respectively.

Directional Transfer Functions (DTFs)

Each HRTF can be considered as a result of the convolution of two transfer functions, the directional transfer function (DTF) and the common transfer function (CTF, Middlebrooks and Green, 1990). The CTF contains the direction-independent portion of the HRTF such as effects of the ear canal. Removing the CTF from the HRTFs reduces the detailed acoustic effects associated with the precise location of the in-ear-microphones and emphasizes the directional effects in the HRTFs.

In order to calculate the CTF, its amplitude spectrum is approximated by averaging the real cepstra of HRTFs for all available positions for one ear.[1] The phase spectrum of the CTF is considered to be a minimal phase. We use the equalized, not the final HRTFs to calculate the CTF.[2] Each HRTF is transformed to a real cepstrum. All cepstra are averaged and the average cepstrum is transformed to the frequency domain. The minimal phase for the amplitude spectrum is calculated using the Hilbert transform. The amplitude and phase spectra are combined to build the raw CTF. The raw CTF still contains the position-dependent components of the room effect. Thus, the raw CTF is cepstrally smoothed using the procedure from the reference measurement. Finally, the smoothed CTF is windowed to a length of 5.333 ms.[3] Figure 17 shows the CTFs for both ears. Note the low-pass-filter characteristic which is assumed to be an effect of the head, which absorbs more energy in the higher frequency region than in the lower frequency region.

1 Use “AA_AverageCepstral” to calculate cepstral averages.
2 The calculations in this chapter are a part of “hrtf_CalcALL” from the previous chapter.
3 The CTFs are saved in “hrtf_M_ctf.mat”.

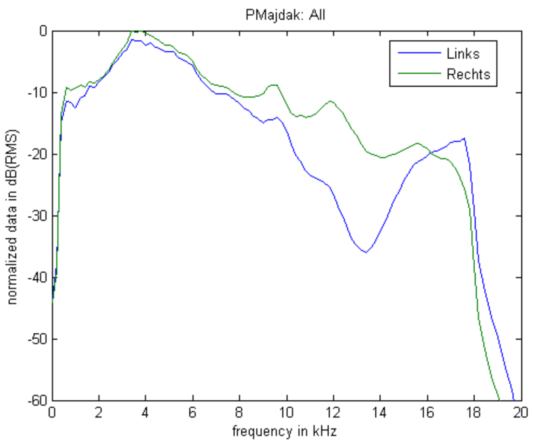


Figure 17: Amplitude spectra of the common transfer functions for the left and right ear

The calculation of DTFs is performed by removing the CTF from each HRTF for one ear. This is achieved by dividing the equalized, not the final HRTF by the corresponding CTF within the frequency range of 300 Hz to 18 kHz. The amplitudes outside of that range are set to zero. The result is the raw DTF, which is then cepstrally smoothed using the procedure from the reference measurement and yields the final DTFs[1]. As for the HRTFs, the DTFs are additionally windowed to a length of 5.333 ms to obtain filters for convenient filtering of free field sounds.[2]

Because of the low-pass-filter characteristic of the CTF, the calculation of DTFs corresponds to a whitening process: Compared to the HRTFs, the amplitudes in the high frequency region are elevated, see Figures 28 and 19.[3]

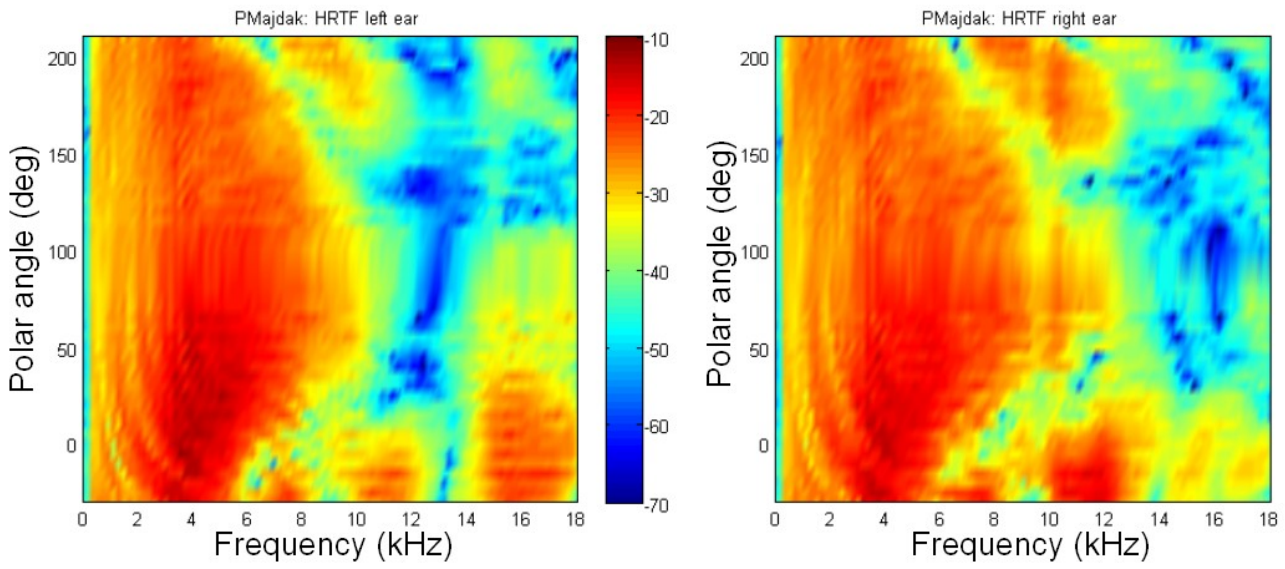


Figure 18: Comparison of the HRTFs left and right. Both panels show the amplitude spectra in the median plane.

1 The long DTFs are saved in “hrtf_M_dtf.mat”.
2 The short DTFs are saved in “hrtf_M_dtf 256.mat”.
3 Load “hrtf_M_dtf 256.mat” and start “hrtf_ShowData” to get some nice plots of the DTFs.

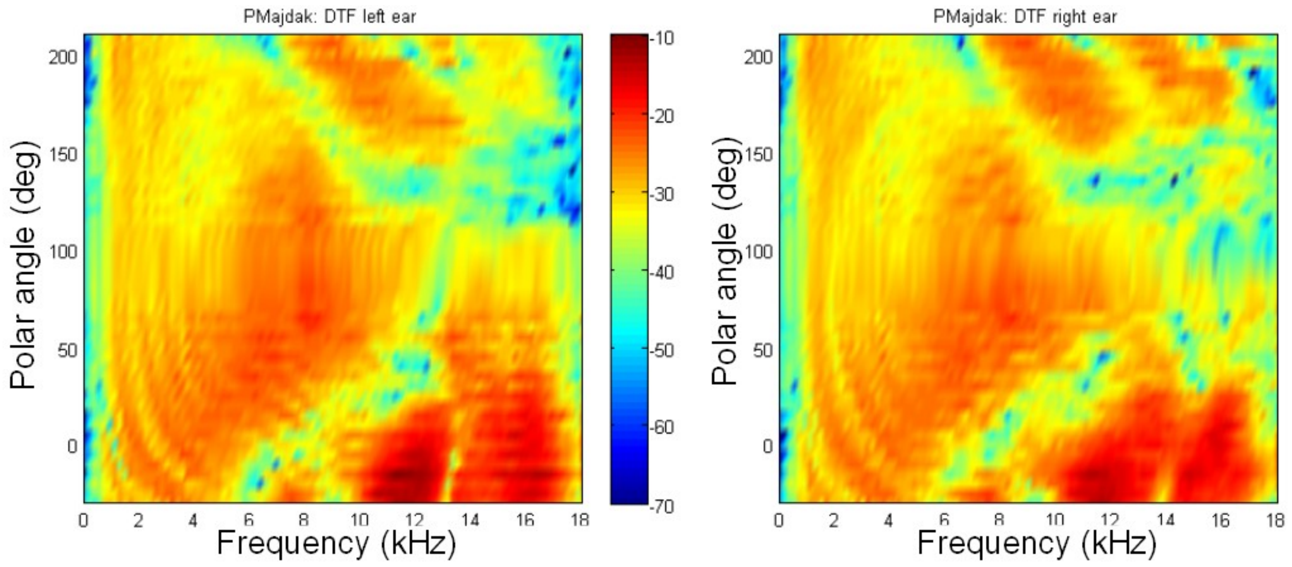


Figure 19: Comparison of DTFs left and right. Both panels show the amplitude spectra in the median plane.

References Post Processing

Bogert, B. P., Healy, M. J. R., and Tukey, J. W. **(1963)**. "The quefrency alanysis of time series for echoes: cepstrum, pseudo-autocovariance, cross-cepstrum, and saphe cracking," in *Proceedings of the Symposium on Time Series Analysis* edited by M. Rosenblatt (Wiley, New York).

Fielder L. D. **(2003)**. "Analysis of Traditional and Reverberation-Reducing Methods of Room Equalization," J. Audio Eng. Soc. **51**, 3-26.

Majdak, P., Balazs, P., and Laback, B. **(2007)**. "Multiple exponential sweep method for fast measurement of head-related transfer functions," J Audio Eng Soc **55**, 623-637.

Middlebrooks, J. C., and Green, D. M. **(1990)**. ''Directional dependence of interaural envelope delays,'' J. Acoust. Soc. Am. **87**, 2149–2162.

Oppenheim, A. V. and Schafer, R. W. **(1998)**. *Discrete Time-Signal Processing*, 2nd ed. (Prentice Hall, New Jersey).

MESM Parameters (Multiple Exponential Sweep)

The GUI to calculate MESM parameters can be launched by executing the function "Calculate MESM parameters" right at the bottom of the main window, in the Results menu.

This is a shortened and simplified introduction how to optimize MESM parameters for a MESM/HRTF measurement in AMTatARI. For further details please read the source publication: "*Multiple Exponential Sweep Method for Fast Measurement of Head-Related Transfer Functions*", Majdak et al. 2007, download: http://piotr.majdak.com/publications/2007_JAES_MESM_paper.pdf

The described process is not linear; if some of the later parameters are changed it might influence the previous ones. So the optimization process includes possibly some iterations.

1. find amplitude that does not cause clipping of your signal, when playing one sweep (Variables → Amplitude [dBFS], or directly in item list → Amp)
2. find out minimum signal duration to fulfill your SNR requirements (Constants → Stimulus Length); a longer duration is not a problem but will increase your measurement time
3. analyze the impulse response of your recorded sweep in a figure:
    1. Impulse Response Toolbox: calculate *C. *LIN, convert to SOFA object (structured meta

data, hM)

2. Post Processing Toolbox: Plot impulse response (time, amplitude)
3. the signal and plotting parameters (durations) should cover enough time to analyze the impulse response, in a similar view like in Figure 20.
4. You will need some of these parameters in the next step(s).

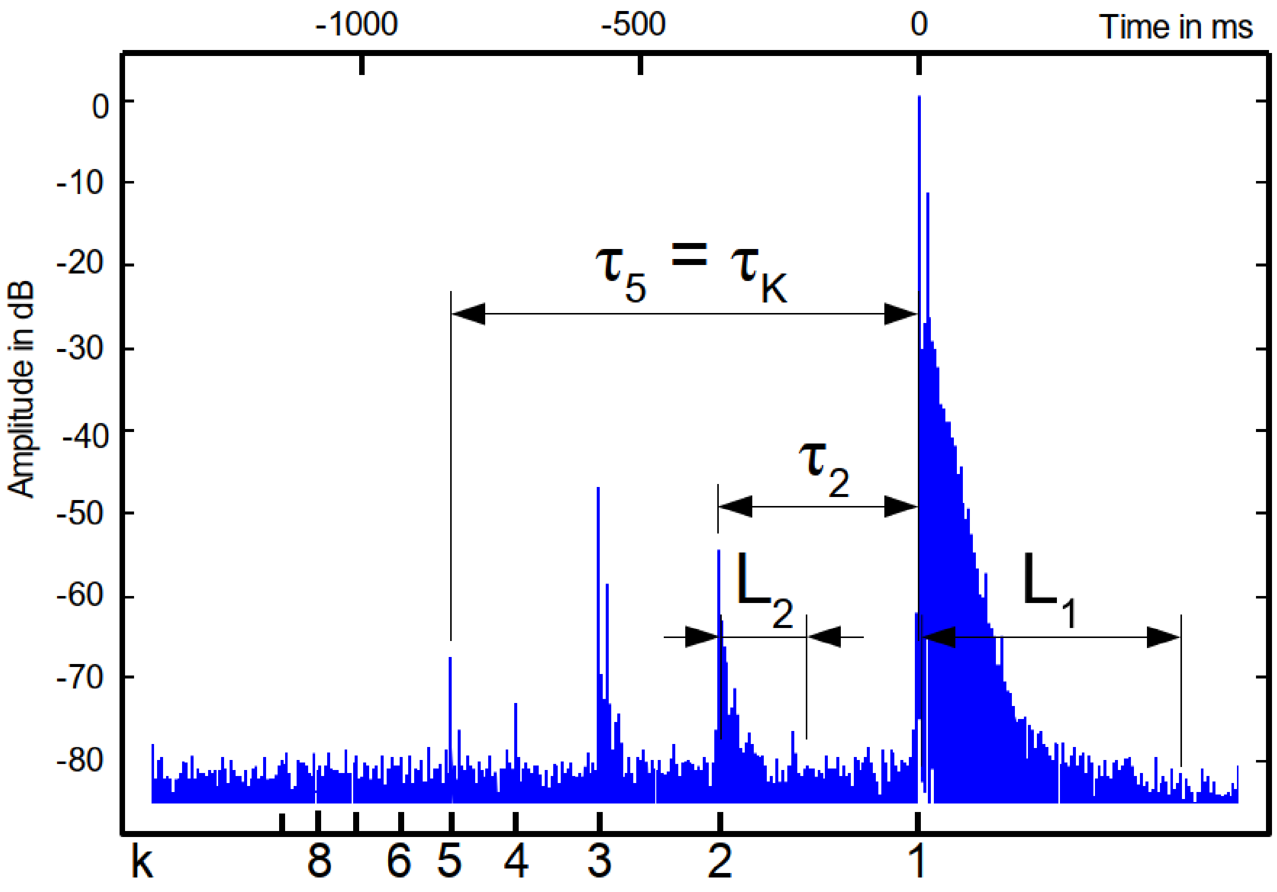
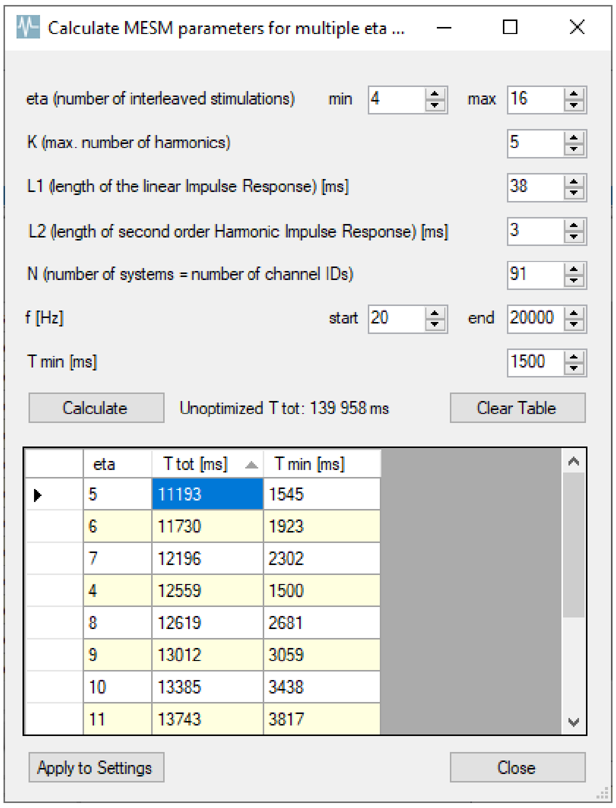


Figure 20: Impulse response, with its harmonics

4. Run “Calculate MESM parameters” from the results menu. A new window (see Figure 21) pops up and asks for some parameters:

- eta: number of interleaved sweeps, define a range of how many sweeps you might want to interleave; the value should not exceed your number of Play Channels
- K: max number of harmonics, having a look at Figure 20 you can see that there are 5 harmonics visible → in our case K=5.
- L1: Length of impulse response: time duration of our signal duration, see Figure 20 for a graphical explanation
- L2: Length of second order harmonic impulse response: see Figure 20 for a graphical explanation
- N: Number of systems = number of Play Channels (defined in Settings → Signal)
- f: frequency range of exponential sweeps [Hz] (defined in Settings → Constants)
- T min: minimum sweep duration according to step 2 above (defined in Settings → Constants)
- Press Calculate
- The result is a table at the bottom of the form: It shows how many sweeps you can interleave (eta) to reach a total measurement time of T tot. The duration of the sweep has to be adapted
- By pressing “Apply to Settings” your Settings are adapted to the calculated parameters.



eta (number of interleaved stimulations)	min	4	max	16
K (max. number of harmonics)				5
L1 (length of the linear Impulse Response) [ms]				38
L2 (length of second order Harmonic Impulse Response) [ms]				3
N (number of systems = number of channel IDs)				91
f [Hz]	start	20	end	20000
T min [ms]				1500

Unoptimized T tot: 139 958 ms

eta	T tot [ms]	T min [ms]
5	11193	1545
6	11730	1923
7	12196	2302
4	12559	1500
8	12619	2681
9	13012	3059
10	13385	3438
11	13743	3817

Figure 21: Calculate MESM parameters

As you can see, your signal duration changed. You might want to repeat some steps (check for clipping,…), review your results, and maybe repeat some of the calculations.

Do not forget to save your Settings file once you have determined the optimized parameters for your system!

Questions? Feedback?

This documentation should contain an overview to a very detailed description of AMTatARI and its functionality. In case there are still open questions, or you have some feedback, feel free to contact michael.mihocic@oeaw.ac.at or open a ticket on Sourceforge: https://sourceforge.net/projects/expsuite/.