# A MATLAB Platform for Precedence–based Source Separation: User Guide

Chris Hummersone*

August 23, 2011

## Contents



*christopher.hummersone@surrey.ac.uk

# 1. Introduction

This document describes a MATLAB program that separates out the constituent signals of binaural mixtures of sound by utilising spatial cues. The software contains implementations of numerous computational precedence models that aim to enhance spatial cues in reverberant environments. As well as providing a platform to assess the effectiveness of these models using a variety of metrics and in a range of acoustic and mixture conditions, the software is also fully expandable, allowing the user to easily add new precedence models or use new Binaural Room Impulse Responses (BRIRs). The source code is extensively commented if more detail is required. In addition, the software was developed for the author's thesis, where more technical details can be found (see Chapters 5 and 6 in particular):

Hummersone, C. (2011), *A Psychoacoustic Engineering Approach to Machine Sound Source Separation in Reverberant Environments*, Ph.D. thesis, University of Surrey.

# 2. Installation

The software and all of its required files are contained within its root folder. To use the software, simply save it to your hard drive and either: change the MATLAB current directory to this path, or add the path to your MATLAB search path. The software can then be started by typing `PrecSep` at the MATLAB prompt. The software automatically adds the required paths to the MATLAB search path and removes them once the Graphical User Interface (GUI) is closed.

## 2.1. Requirements

The software has the following requirements:

- MATLAB R2007a or later (earlier versions may work although they are not tested)
- Signal Processing Toolbox
- A compatible C compiler (all dependent C functions are automatically compiled). See:
  `http://www.mathworks.com/support/compilers/current_release/`
  `http://www.mathworks.com/support/compilers/previous_releases.html`

The software has been compiled and tested on Windows XP (compiled under Microsoft Visual C++), Mac OS X 10.5 & 10.6 (compiled under Xcode) & Ubuntu Linux 9.10 and 10.04 (compiled under GCC).

# 3. Using the Graphical User Interface

The GUI contains the following parameters:

`Target Sound File` Choose the target sound files. Multi-select (by ctrl-click or cmd-click) or click 'Add' to add multiple stimuli.

`Interferer Sound Files` Choose the interferer sound files.

`Rooms to test` Choose the rooms to test by using the arrow buttons to build a list of rooms. The labels correspond to the subfolder names of the BRIRs folder. In the default case they have been grouped according to $RT_{60}$.

`Target azimuth` Choose the target azimuth. This can be a single value or a MATLAB statement that evaluates to a vector. Furthermore, the values must have corresponding angles in the BRIR set (the default is -90:5:90).

`Interferer azimuth` Choose the interferer azimuth. As with the target azimuth, this can be a valid single value or vector. Furthermore, the number of target and interferer azimuths must be the same, since they are tested in pairs. The software assumes that the target is placed on the left[1].

[1] This means, for example, that the target could be specified as an interferer stimulus, provided that the target azimuths are to the right of the interferer azimuths.

**Table 1:** Permitted precedence model parameters

| **Precedence Model** | **Parameters** | **Permitted Values** |
|---|---|---|
| Baseline | 1. Inhibitory Gain $G$ | $G \geq 0$ |
| | 2. Inhibitory Time Constant $\alpha_p$ [ms] | $\alpha_p \geq 0$ |
| Martin | 1. Inhibitory Gain $G$ | $G \geq 0$ |
| | 2. Inhibitory Time Constant $\alpha_m$ [ms] | $\alpha_m \geq 0$ |
| Faller & Merimaa | 1. IC Threshold $\Theta_\chi$ | $0 \leq \Theta_\chi \leq 1$ |
| | 2. Exp Win Time Constant $\alpha_f$ [ms] | $\alpha_f \geq 0$ |
| Lindemann | 1. Inhibition Parameter $c_{\text{inh}}$ | $0 < c_{\text{inh}} \leq 1$ |
| | 2. Fade-off Time Constant $T_{\text{inh}}$ [ms] | $T_{\text{inh}} \geq 0$ |
| Macpherson | 1. Inhibitory Gain $G$ | $G \geq 0$ |
| | 2. Inhibitory Time Constant $\alpha_m$ [ms] | $\alpha_m \geq 0$ |

`Target-to-Interferer Ratio [dB]` Specify the ratio of the RMS levels of the target to interferer. This may be any single value or vector.

`Results Directory` Choose the results directory. The default is the `Results` folder. The filename can be manually modified if necessary.

`Listen to result?` Choose whether the software plays the resulting separation.

`Save result .wav file?` Choose whether the resulting separation wave file is saved. It will be saved in the same folder as the results. The filenames are tagged with all precedence and mixture parameters.

`Inner Hair Cell Model` Choose the IHC model.

`Precedence Model` Choose the precedence model. Note that changing the precedence model may, in some cases change the IHC model. Generally speaking, the models are interchangeable and indeed it may be interesting to test the effect of changing the model. However, for the Lindemann model, it is important that the IHC model is bypassed (`Bypass` is chosen), since the processing of the fine structure is carried out in the precedence function and is dependent the inhibition parameter; the function requires the unmodified output of the gammatone filterbank.

`Precedence Parameter 1` The precedence parameters depend upon the chosen precedence model. As with some other parameters, they can be a single value or vector. Furthermore, each model/parameter combination has some restrictions on these values, given in Table 1.

`Precedence Parameter 2` Another precedence model parameter.

`Anechoic Room` Choose the room tag corresponding to an anechoic set of anechoic BRIRs. These BRIRs are used to calculate an ideal ILD template.

`Low CF [Hz]` The lowest centre frequency of the gammatone filterbank. Must be a positive value.

`High CF [Hz]` The highest centre frequency of the gammatone filterbank. Must be a positive value, less than the nyquist limit (the sampling frequency is given in the top-right corner of the GUI[2]) and greater than the lowest centre frequency.

`No. of channels` The number of channels for the gammatone filterbank. Must be a positive integer.

`Max cross-correlation lag [ms]` The maximum cross-correlation lag. Cross-correlations will be tested between `-maxlag` and `maxlag`. Must be a positive value.

`Frame Length [ms]` The length of the analysis frame in milliseconds. Must be a positive value.

`No. of frame to overlap for localisation` Choose the number of frames over which to calculate the cross-correlograms. This value is not available for every model. It must be a positive integer.

The GUI will highlight, in yellow, any erroneous fields that do not meet these requirements.

The software performs one separation for each combination of the mixture variables and precedence model parameters, so that the performance for each value is compared to the performance for every other value. Therefore, the total number of iterations will be:

[2] The sampling frequency is determined by the sampling frequency of the BRIRs.

```
results
  . (modelname)
      . displayname
      . prec_par_1_name
      . ...
      . target
      . Room
      . ...
  . (modelname)
      . displayname
      . prec_par_1_name
      . ...
      . target
      . Room
      . ...
```

**Figure 1:** The `results` structure.

#Target stimuli × #Interfering stimuli × #Rooms × #TIRs × #Azimuth pairs × #First precedence model parameters × #Second precedence model parameters

The stimuli should be .wav files. They can be stereo or mono and any sampling frequency but note that they will be mixed to mono and resampled to match the sampling frequency of the software. Where necessary, stimuli will be zero-padded so that they are all the length of the longest file. For convenience, stimuli can be placed in the `Stimuli` folder; the GUI opens this folder by default.

The target, interferer and mixture can be previewed using their corresponding buttons on the GUI[3]. The interferer preview will use the currently highlighted interferer. The mixture preview will use the currently highlighted room and interferer, and the first element of the azimuths and TIRs. The mixture is remembered until any of these mixture parameters change.

Once all of the GUI parameters have valid values, the "GO!" button will be enabled and the calculations can be carried out. If the results for the chosen model already exist in the specified results file, the user will be warned and asked whether they wish to overwrite or append them (or cancel). During calculations a progress bar will be displayed (which increases only with each iteration) and the current iteration, estimated azimuths, IBMR and processing times will be displayed after each iteration.

## 4. The Results File

The results are saved to a MATLAB structure and a substructure is created for each precedence model that is tested. The structure of the `results` structure is given in Figure 1. Each substructure has two types of fields: some are global for the model under test, others record data for each iteration. The global fields are:

`revision_profile` A snapshot of version information for each of the toolbox functions. The data can be useful in maintaining an audit trail, or establishing the version used to generate a particular set of results.
`prec_par_1_name` Name of the first precedence model parameter (see Table 1).
`prec_par_2_name` Name of the second precedence model parameter.
`displayname` The display name of the model, i.e. one suitable for use on plots.

The data fields are:

`target` The target stimulus name (derived from the filename minus the extension).

[3]Preview functions use the native MATLAB `sound` function, which can be terminated by pressing ctrl+c in the MATLAB command window.

`interferer` The interfering stimulus name.
`Room` The tag for the room (as it appears on the GUI).
`TIR` The Target-to-Interferer Ratio.
`azi_t` The user–specified azimuth of the target.
`azi_i` The user–specified azimuth of the interferer.
`prec_par_1` The first precedence parameter value.
`prec_par_2` The second precedence parameter value.
`SINR_L` The Signal-to-Ideal-Noise Ratio for the left ear signal, i.e. the Signal-to-Noise Ratio calculated with the target resynthesised from the Ideal Binary Mask (IBM) as the reference.
`SINR_R` As above for the right ear signal.
`SNR_L` The Signal-to-Noise Ratio for the left ear signal.
`SNR_R` As above for the right ear signal.
`IBMR` The Ideal Binary Mask Ratio.
`SNR_L_IBM` The Signal-to-Noise Ratio achieved by the IBM for the left ear signal.
`SNR_R_IBM` As above for the right ear signal.
`prec_time` The time taken to run the precedence model.
`overall_time` The time taken to perform the overall separation.

These fields record the value of each parameter for every iteration of the algorithm. Broader trends in the data can be identified by calculating means over variables. For example, the performance for a given parameter value of the Baseline model in a given room could be calculated in the following way:

```
% Load the results file:
load results.mat
% Choose the model:
modelname = 'Baseline';
% Find the data:
IDs = results.(modelname).prec_par_1==0.5 ...
    & results.(modelname).prec_par_2==5 ...
    & strcmp(results.(modelname).Room,'0_00s');
% Take the mean:
performance = mean(results.(modelname).IBMR(IDs));
```

## 5. Expanding the Software

As well as providing the means to test each of the implemented precedence models, the software also offers expansion possibilities. Specifically, mechanisms are provided to change the BRIRs used by the software and to include additional precedence and IHC models. Some other areas that could be upgraded are also suggested. More details are given below.

### 5.1. Binaural Room Impulse Responses

Any BRIRs can be used in the software and there are only a few minor requirements:

- The BRIRs must be placed in subfolders of the `BRIRs` folder.
- Each folder is equivalent to a room: one BRIR set per room/folder.
- The filenames within a folder must be identical apart from containing the azimuth (the azimuth part must be enclosed in non-numeric characters).
- They should be 2-channel wave files.
- They should all have the same sampling frequency.

The software will automatically ‘detect’ the folders contained within the `BRIRs` folder each time it is started. The subfolder names will be used throughout the software and in the GUI as a unique tag. The tag will appear in the ‘Room’ field of the results structure. The sampling frequency of the BRIRs determines the global sampling frequency for the software. Any other stimuli that are loaded in will be resampled to this sampling frequency.

## 5.2. Precedence Models

The addition of precedence models requires two operations: 1. creation of the precedence function(s) and 2. modify the configuration file. These are discussed below.

The precedence model function has two requirements:

1. Since the software uses function handles to refer to files, the argument list is always the same. Hence, the input argument list at the top of the new function must match the call from the `separate` function. This does of course mean that some variables are likely to be unassigned, which is unfortunate, but unavoidable. Users with MATLAB R2009a or later can un-assign these inputs by replacing their entry with a tilde (~) (this has obviously not been implemented here for compatibility reasons).
2. The function must return `ccg`: the cross-correlogram for every frame of the input. `ccg` should have dimensions [lag,channel,frame] with lengths [`2*maxlag+1`,`numchans`,`frameCount`].

Much of the precedence–related operation of the software is determined by the configuration file `PrecSep.config`, which is in fact just a CSV file. The file contains a table of values that control parameters relating to each precedence model. The field names are given on the first row (**do not adjust this row**) and the fields are:

`displayname` A unique value for the results field 'displayname' (see Section 4) and the name shown in the precedence model GUI pop-up menu.

`modelname` A unique identifier for the given model. It will be used to name the substructure of results and warpers (see Section 5.4) structures.

`mex_file` If the precedence function calls or is a mex file, write the filename (incl. extension) here. The software will check that it is compiled at runtime. This field should be left empty if no mex file is used.

`mex_options` Options to pass to the mex compiler, if any.

`prec_fhandle` The handle for the precedence model function. For example, the baseline model uses the function `prec_baseline.m` and so its handle is `@` followed by the filename (no extension): `@prec_baseline`

`prec_par_1_label` A nice label for the first precedence model parameter. This is used on the GUI and in the `prec_par_1_name` field of `results.(modelname)`.

`prec_par_1` A default value for the first precedence model parameter.

`prec_par_2_label` A nice label for the second precedence model parameter.

`prec_par_2` A default value for the second precedence model parameter.

`noverlap_enable` Control whether the GUI text box "No. of frames to overlap for localisation" is enabled, since not all precedence models use this parameter.

`noverlap` Default value for the above parameter. Setting the value reserves `noverlap` space at the end of the signal and reduces the frame count by `noverlap` frames. If it is not used then set it to 1, or any other value that is sufficiently large to reserve enough frames for the cross-correlations to be calculated without exceeding the limits of the signal.

`IHC_model` The default IHC model for the chosen precedence model. This is specified as an integer relating to the value of the IHC model pop-up menu: the top model in the box has a value of 1; the value increases down the menu.

This configuration file is automatically loaded every time the software starts.

There is a further optional step that can be taken when including new precedence models: the GUI validation of the precedence model parameters. This is dependent upon the chosen precedence model and defined in `PrecSep.m`. To add validation for new precedence models, simply adjust the appropriate switch/case clauses in:
`txtPrecedenceParameter1_Callback`
`txtPrecedenceParameter2_Callback`

An additional function is included in the package: `ch_xcorr.m`. The function is a flexible precedence model, allowing easy control over cross-correlation parameters including overlap, normalisation, interaural coherence threshold, and inhibition. The function is not utilised in the default models, but is provided as an additional tool. Details of the algorithm are provided in ch_xcorr_doc.pdf.

### 5.3. Inner Hair Cell Models

The call to the IHC function works in a similar way to the precedence function call, i.e. the software uses a handle to the function. Hence, as with precedence functions, the input argument list must match the call from `separate.m`. In order to add new IHC models, three operations must be performed:

1. Create the IHC function.
2. Append an appropriate name to the string property of `cboIHCmodel` in `PrecSep.m`.
3. Use this name in the switch/case operation in "Define operation for chosen Inner Hair Cell model" (in `separate.m`) and assign the function handle to `ihc_fhandle`.

### 5.4. Other Areas for Work

One key area for work is the enhancement of the function `get_warp_func` that creates functions to warp Interaural Time Difference (ITD) to azimuth. The functions take the form of a matrix, one row per frequency channel, one column per degree for `-maxdeg:maxdeg` where `maxdeg` is the maximum angle in degrees. Each element relates this angle to its corresponding ITD value. Currently, the two are related by a simple geometric function that is not related to the BRIRs (as it should be). It would be relatively straightforward to adapt this function and make calculations based on the anechoic impulse responses, esp. as the function is already provided with all the necessary data.

## 6. Acknowledgements

I would like to thank the following people who, directly and indirectly, have provided code that is used in this software:

- Dr. Richard O. Duda from San Jose State University, California, for his assistance with implementing the Lindemann model.
- Guy Brown from Sheffield University, UK, for his freely available MATLAB Gammatone, ERB and cross-correlogram code, available from: `http://www.casabook.org`
- Malcolm Slaney for his MATLAB "Auditory toolbox" (the Meddis Hair Cell function in particular), available from `http://cobweb.ecn.purdue.edu/~malcolm/interval/1998-010/`

This work was supported by the EPSRC.