Behavioral Auditory Research Tests (BART)

Revised: 02/2/2010

OVERVIEW

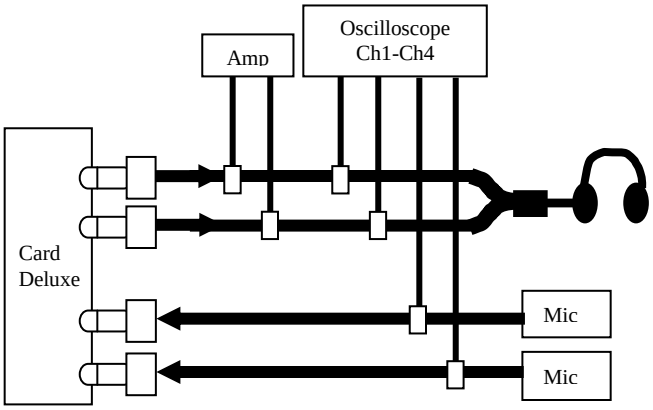
The purpose of Behavioral Auditory Tests (BART) is to simplify the process of designing and implementing experiments in auditory research. This comprehensive program is designed to include features that will facilitate data collection for both adults and children. BART incorporates the following: 1) a variety of signal processing and signal generation options, 2) standard or real-ear calibration, 3) an automated audiogram algorithm, 4) adaptive and fixed-level trials, 5) a variety of interactive games designed to reinforce responses, 6) an analysis of performance in real time, 7) frequency shaping of test stimuli for hearing-impaired subjects, and 8) a detailed summary of results.

Graphical interfaces have been provided to allow the user to monitor the trial-by-trial responses and the summarized results. The parameters controlling an experiment are listed in a definition (text) file. This choice allows for flexibility in the development and use of the program. In addition, a graphical interface assists users in managing the 50+ parameters that can be manipulated within the program. Because many of the parameters are interdependent, it is recommended that the user become familiarized with the experimental parameters prior to designing an experimental definition file.

SYSTEM REQUIREMENTS

Windows 2000 or XP
Sound card
Windows Drivers (MME)
Direct X
Personus Low-Level Headphone Amplifier (optional)
Wideband high-end earphones or loudspeakers (power amplifier may be needed for loudspeakers)

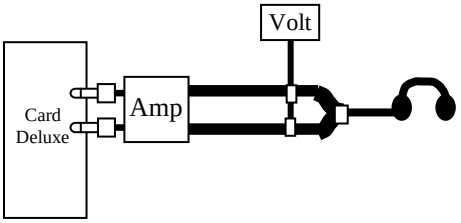
SET UP



1. Under the Control Panel, open the Sounds and Audio Devices Window.
2. Under the 'Sounds' tab and 'Sound Scheme' option, select 'No Sounds'. If this option is not selected, the subject will hear all computer-generated warning sounds through the earphones (or other transducer). If an amplifier is included in the circuit, these sounds may be uncomfortably loud.
3. Under the 'Audio' tab, select the desired sound card (e.g., Card Deluxe Analog) under the 'Sound Recording' and 'Sound Playback' options.
4. Although earphones may be connected directly to the output of the sound card, an amplifier and oscilloscope/voltmeter may be patched into the circuit. The amplifier is necessary only for studies requiring levels that exceed the maximum output of the soundcard (about 110 dB SPL).

CIRCUIT AMPLIFIER

In cases where hearing loss is the in moderately-severe to severe range, it may be necessary to include an amplifier in the circuit in order to ensure the audibility of test signals. It is *not* necessary to utilize an amplifier for thresholds measures. To insert the amplifier into the circuit, the right and left outputs from the Card Deluxe should be re-routed to the right and left channels at the rear of the HP-4 amplifier. Because each of the four outputs at the front of the amplifier are stereo outputs, the *same amplifier gain will be applied to both ears.* At the front of the amplifier, a stereo to mono connector should be used to split the signals into right and left channels. To calibrate the circuit, a T-connector should be used to load the circuit with the appropriate earphone while simultaneously monitoring the voltage on an rms voltmeter. Open BART and play the 1-kHz calibration tone. Adjust the volume control as shown in the table below. Repeat for the other earphone.



Voltage	Gain
350 mv	0 dB
700 mv	6 dB
1.4 v	12 dB
2.0 v	15 dB

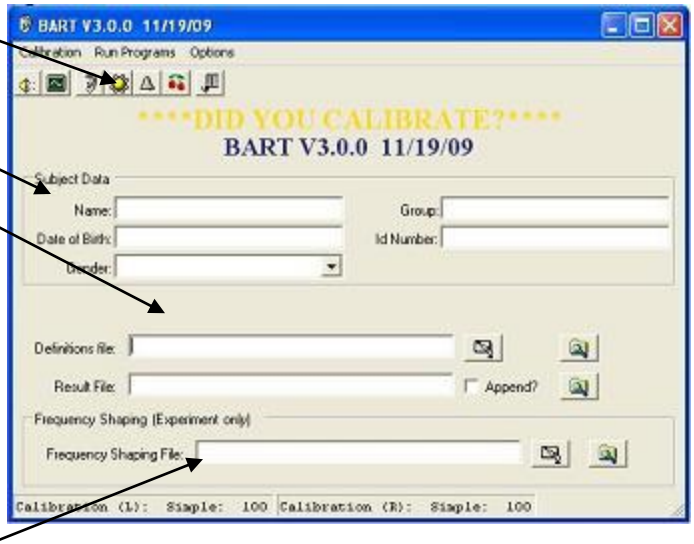
GETTING STARTED

Before using the program, the pictures associated with the game files used for feedback must be copied to a location on the network or C:\ drive that can be accessed during the experiment. The path to that location must also be added to each game .txt file. For example, the puzzle game contains two lines of code. The first specifies the source code for the game and the other gives the path to the picture to be revealed. The path to the picture must be included.

```
REVEAL;
R:\HARL\Common\kid\kid.bmp;
```

For further information regarding feedback and games, see page 26.

The initial startup screen includes 4 sections. The first section contains several buttons related to the functions currently available in the Test Suite: Simple Calibration, Chirp Calibration, Audiogram, Experiment, Categorical Loudness, Paired Comparison, Reset to Simple Calibration. These are defined in more detail below. The second section provides fields in which subject information can be entered. This information is printed as a header at the top of the Results file.



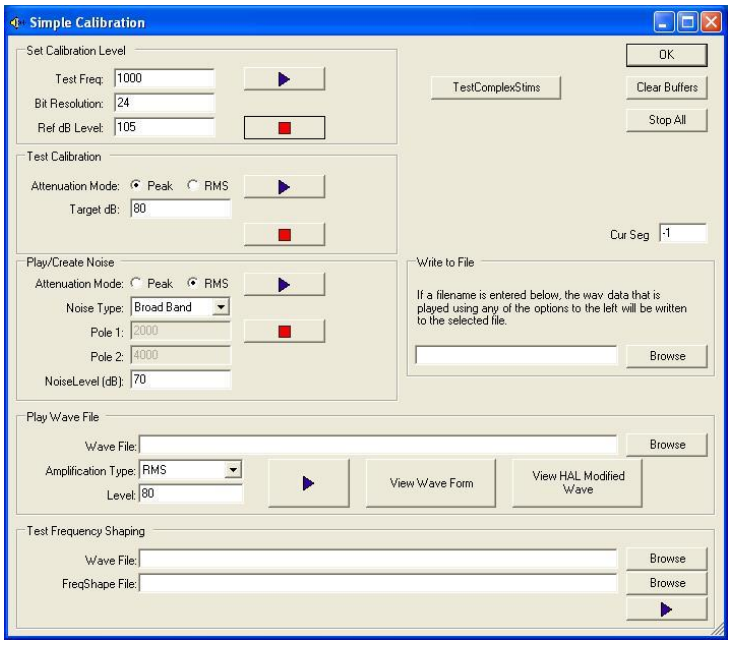
The third section defines the path to the definition file for the desired experiment and the destination of the results file. *Browse* buttons have been provided for both fields. In some cases the definition file may need to be altered prior to an experiment and this can be done by selecting the *Edit* button. The *Append* checkbox is used to append new data to an existing file. In the Audiogram Results files appended threshold data is integrated with original threshold data. In other BART applications, new data is appended at the end of the original data in the file.

For hearing-impaired subjects, the fourth section defines the path to a frequency shaping file. For further details regarding frequency shaping, see Appendix D. A Browse button has been provided for this field. In some cases the file may need to be altered prior to an experiment and this can be done by selecting the Edit button.

FUNCTIONS

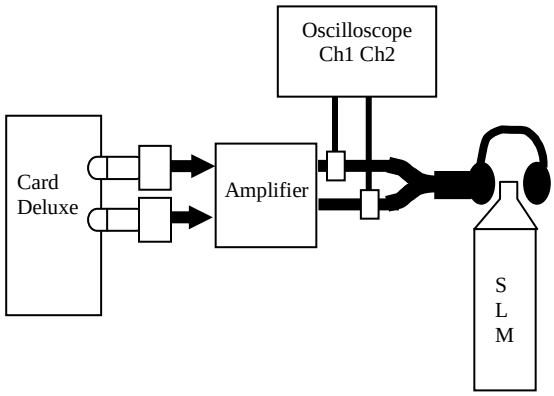
Six functions are currently defined within BART. More functions may be added at a later date. Although the main function in this program is experimental design, special utilities for the purpose of calibration, testing audiometric thresholds, and testing games within an experiment also have been developed. Each of these functions calls a unique user interface.

Simple Calibration



The calibration window can be used for two purposes: 1) to measure a calibration tone and 2) to play noise or wave files at designated levels. The first option is necessary for any testing where the level of the stimulus must be controlled. The second option allows the user to generate stimuli to verify the output of the signal processing features within the program.

The **Set Calibration Level** section illustrates this process. The *Play Cal Tone* button is used to play a high level pure tone at a selected frequency via the transducer to be used in the experiment. Signal level (in an appropriate coupler or in the sound field) should be measured by an independent system (e.g., sound level meter). The figure to the right shows an example of a circuit to be calibrated using a sound level meter. The level measured at the sound level meter should then be entered into the *Ref dB Level* field followed by *Stop*. All subsequent stimuli presented by the program will be referenced to this level until another calibration is performed.



NOTE: The OK button MUST be clicked after entering the calibration level. If not, the value will not be saved when exiting this screen.

The **Test Calibration** section is used to generate signals in order to verify that the program is calculating the output levels correctly. To play a pure tone (at the frequency specified in the Set Calibration Level section), type the desired level in the *Target dB* field and press the *Play Attenuated Tone* button. The output then can be measured using an oscilloscope, sound level meter, or voltmeter.

The **Play/Create Noise** section allows the user create low-pass, high-pass, or bandpass filtered noise. In all cases, the rejection rate is essentially infinite. The band-pass filtered noise will be one octave wide centered at the specified test frequency. The noise can be played using the *Start Noise* and *Stop Noise* buttons.

The **Play Wave File** section allows the user to play a stored wave file at the specified Target dB Level. If no level is specified, the wave file will be played at 80 dB SPL. The wave file can be selected either by typing in the path to the file or by using the Browse button. If the wave file contains a pure tone, the *Peak* option should be selected. If the wave file contains speech or noise, the *RMS* option should be selected.

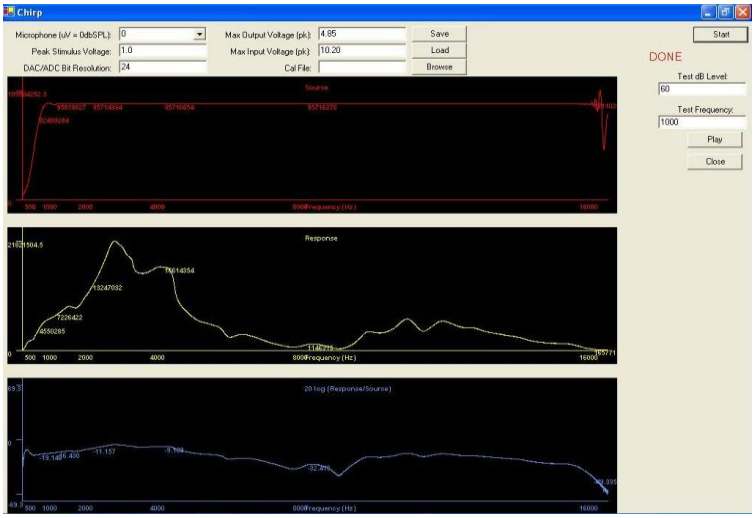
The **Test Frequency Shaping** section allows the user to play a stored wave file using a specific *FreqShape* file. Both the wave file and the frequency shaping file can be selected either by typing in the path to the file or by using the Browse button.

The files generated using any of the calibration utilities can be saved. In the **Write to File** section, enter a filename (.wav file). Use the *Browse Button* to select the appropriate file location. The file will be created automatically when the stimulus is played.

Chirp (Real-Ear) Calibration

This method allows the user to calibrate sound pressure in the ear canal of each individual research participant or in a designated cavity (e.g., 6-cm^3 coupler). A chirp is used as the calibration stimulus. The following example assumes a 24-bit sound card (e.g., Card Deluxe) and the use of an Etymotic Research microphone (ER7C or ER10C).

Microphone Sensitivity = Microphone cal level (uV corresponding to 0 dB SPL). Indicate if mic sensitivity is set to 0, 20, or 40 dB (default is 0 dB).
Peak Stimulus Voltage = (default is 0.25 v)
DAC/ADC Bit Resolution = (default is 24)



Max Output Voltage (pk) = Measured maximum output voltage of the soundcard (default = 4.85 v)
Max Input Voltage (pk) = Measured maximum input voltage of the soundcard (default = 10.2 v)

On the screen to the left, the top panel is the FFT of the output (card out), the middle panel is the FFT of the input (card in), and the lower panel is FFT (input) / FFT (output) and will be used for calibration. To begin calibration, click on the *Start* button. To check calibration or verify stimulus level for other signals, enter the desired level and frequency and click on *Play*.

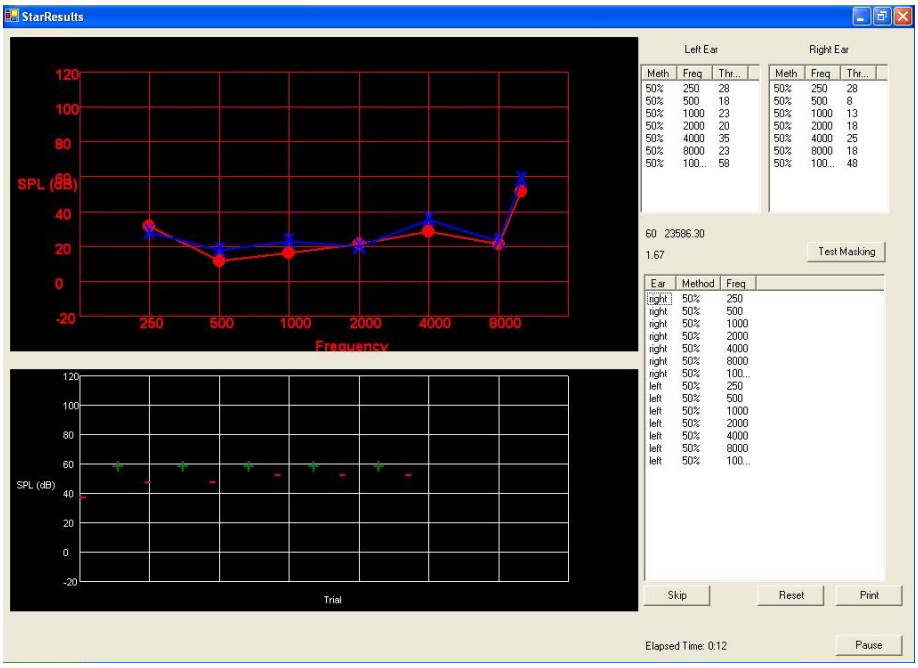
NOTE: The *chirp calibration method is only valid for pure tone stimuli. WAV files cannot be calibrated using a chirp.* In addition, the calibration is lost upon exiting the BART program. Upon re-entry, the calibration reverts to the most recent Standard Calibration value. To avoid errors, the current calibration method (standard vs. chirp) and value (e.g., default 105 dB SPL) are noted on the primary BART screen above the definition file field.

To save the chirp calibration values, use the following procedure:

- Click on the Browse button and navigate to the data folder where the data will be stored.
- Name the calibration file (e.g. *PGS_Rcal.cal).*
- Click on *Start* to play the Chirp. If the response looks as expected, click on *Save*. Chirp-RE differences (lower panel) will automatically be saved as the named .cal file. Currently, these same calibration values will be truncated and saved to the same folder as a .txt file with the same name.

Until the calibration is over-written or the user exits the program, all output signals from BART will be expressed in real-ear SPL. In cases where pure tone thresholds need to be repeated, the file **.cal* (for each ear)

will need to be loaded prior to testing. To re-load the **.cal*, in chirpcal, click on the *Load* button and browse for the appropriate file in data folder where it had been saved.



Experimental Controls

When designing experiments using either *Audiogram* or the *Experiment* function, various stimulus parameters must be specified in the form of a Definition File. The table on pages 11 and 12 contains a summary of these parameters. Not all parameters must be specified in each definition file nor do all parameters apply to both functions (e.g., Audiogram, Experiment). Table I lists the arguments required for each parameter and the functions to which they apply. If an inappropriate parameter is defined, the program will ignore it.

Audiogram

The audiogram function is an adaptive tracking procedure for which threshold is determined using one or more decision criteria based on Levitt (1970). One or two response-button formats may be used. In a 1-button format, the listener pushes the button when a tone is heard. In a 2-button format, a "light" (button changes color from gray to white) indicates the duration of each trial and the listener responds "yes" or "no" to indicate whether he/she heard a tone during the trial.

The figure below shows the results screen displayed during the testing. As each pure-tone threshold is obtained, the result is plotted on the audiogram. The trial-by-trial responses are displayed on the graph below. The parameters governing the audiogram (e.g., test frequencies, step size, response button format, ear) are defined in the Definition File.

Skip and *Reset* buttons have been provided for those subjects who have difficulty with a particular test condition. The *Skip* button skips the current condition and the *Reset* button starts the current condition over again. When the *Reset* button is used, all data will appear in the trial by trial log of the Results file, but only the latter set of values will be used in the summary statistics. When the *Skip* buttons is used, *999* will appear for that trial in the Results file.

Three stopping rules can be applied: standard error, number of reversals, both. The display on the right (see arrow) shows the current standard error, which is calculated after the secondary step size has begun. The standard error is calculated as the absolute difference in dB between each successive response. A standard error stopping rule is the default and is based on all reversals with the exception that it will not stop on a negative reversal. Alternatively, a stopping rule based on total number of reversals at the secondary step size can be

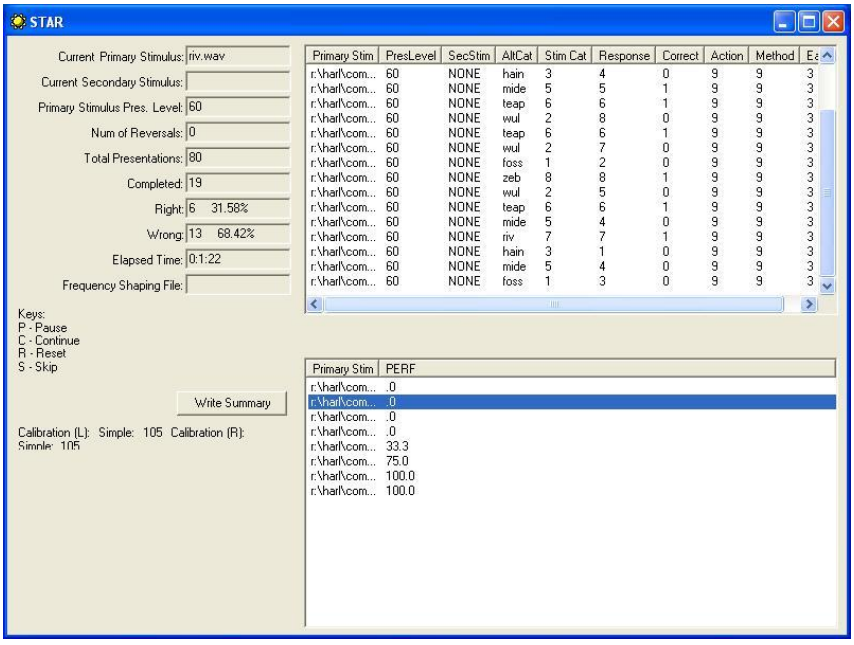
utilized. Finally, a combination of standard error and number of reversals can be specified. In this case, testing will terminate when one of the two rules is met.

One-third octave band noise is available to mask the non-test ear of subjects with asymmetric hearing losses. Contralateral masking will automatically be used when thresholds between ears differ by 40 dB or more at any frequency. **NOTE**: If an assymetrical hearing loss is suspected, it will be important to test the better ear first.

The data from the Audiogram program will not be saved in the Results file if the program is stopped prematurely. However, the program will always create a file (*results.dat*) in the same directory where final results are being saved. That file will be over-written each time Audiogram data is saved to the same folder. In order to preserve the data, the file should be renamed.

NOTE: In the Audiogram program, the starting level for stimulus presentation at a given frequency is always 40 dB SPL, unless otherwise specified (see Stimuli.Primary).

Experiments



The *Experiment* function is the primary feature in BART. The figure below shows the screen that is displayed during an experiment. The trial-by-trial data are displayed in the upper right window. Summarized results are updated after every trial and displayed in the lower window. The fields to the left display the current conditions of each experiment (e.g., current stimulus, stimulus level, total presentations).

For adaptive procedures, the *Skip*, *Reset*, and *StdErr* buttons and fields are activated.

The trial-by-trial data are written to the results file in real time, but the summary statistics are only written at the end of the experiment.

NOTE: If the experiment is stopped prematurely, the summarized data may be written to the results file using the '*Write Summary'* button, otherwise data will not be saved.

The *Skip*, *Reset*, *Continue* and *Pause* functions can be activated by single keystrokes via the keyboard. For keyboard activation:

c = Continue
r = Reset
s = Skip
p = Pause

Categorical Loudness

Frequency Shaping

The *FreqShape* function is a routine that applies frequency shaping to any input signal to compensate for degree and configuration of hearing loss on an individual basis. An Excel spreadsheet is used to calculate the necessary values to shape incoming signals. Once the FreqShape.txt file has been created, the path to the file is defined on the startup screen (see page 3). A complete description of the frequency shaping routine and required files can be found in Appendix D.

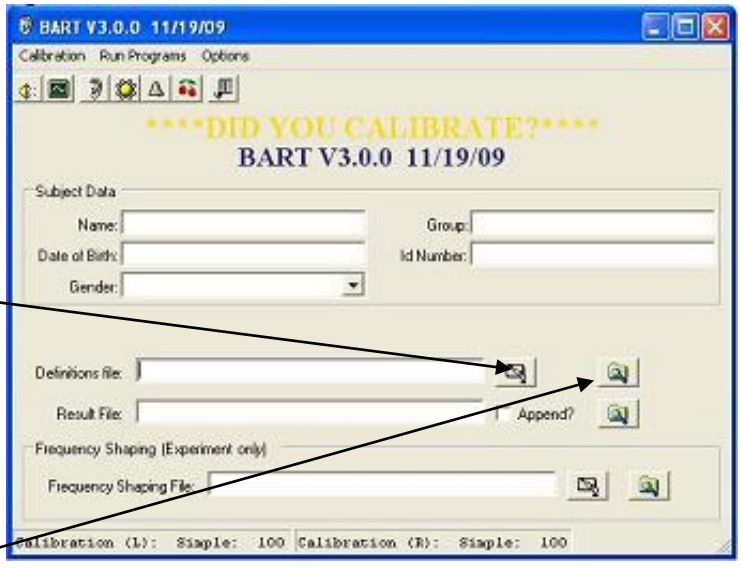
Options

The following features are available under *Options* on the Toolbar.

Test Game allows the experimenter to test the games without actually running an experiment. To test the games listed in a particular definition file, the filename must be entered into the *Definitions File* field. The graphical user interface for this function displays a single response button. All timers and stimuli in the definition file are disabled so that the user may quickly step through the games. See APPENDIX B for more information about the games used in this test suite.

Single Screen This option places both the experimenter and the subject screens on one screen. It can used during the creation and evaluation of an experimental protocol so that two screens, which are normally not in the same location, can be viewed.

EXPERIMENTAL PARAMETERS



The experimental parameters are listed in a definition (text) file. In addition, a graphical interface assists users in managing the parameters that can be manipulated within the program. A new definition file can be created using the graphical interface by clicking on the *Edit* button to open the Parameter tabs. All parameters will be set to their default values.

NOTE: If a particular parameter will not be used for the new experiment, be sure that its default value is set so that it will not interfere with other parameters.

To review or edit a current definition file, select the file using the *Browse* button. Then use the *Edit* button to access the Parameter screens for that definition file.

The following table contains a summary of these parameters as they are listed in the definition file as well as the arguments required for each parameter and the functions to which they apply. The names of the various parameters listed in the definition file may be slightly different from the name used in the BART graphical interface. Thus, the table also lists the page in this manual where the parameter is defined using the parameter name within BART. In the Edit section, the parameters are grouped according to the parameter tab within the program (Constants, Adaptive, Primary Stimuli, Secondary Stimuli, Paired Stimuli, Buttons, Feedback, Results, CL).

Parameter	Arguments	Default	Audiogram	Exp. Des. fixed	Exp. Design adaptive	Page in manual
Button	*xsize* ; *ysize* ; *xloc* ; *yloc* ; *fontsize* ; *category* ; *text* ; *picture file*		X	X	X	25
CatchTrial.BackToBack	(*0,1)*	0	X		X	12
CatchTrial.Enabled	(*0,1*)	0	X		X	12
CatchTrial.Frequency	*%*	5	X		X	12
CatchTrial.Length	*ms*	500	X		X	12
CLButtons	*#*				X	30
DirOfChange	(+,-)	+			X	18
Feedback.File	*filename*	required	X	X	X	28
GameDelay	*#*	1		X		13
GameDelayType	*(0,1)*	0	X	X	X	13
Interval.Buttons	*(0,1)*	0 (off)		X	X	14
Interval.TimeOn	*ms*			X	X	14
Interval.Cue	*(0,1)*	0		X	X	14
Interval.Interval	*ms*			X	X	14
ISI.Jitter	*ms*		X	X	X	13
ISI.Length	*ms*	1000	X	X	X	13
MaxLevel	*upper limit in dB*	120			X	14
MaxTrials	*#*		X	X	X	14
Method	(*1,2,3*)	1	X		X	17
Order.Primary	(*0,1*)	1	X	X	X	14
Order.Secondary	*Block, Random, RandomTrial*	Block		X	X	15
PrimaryAdapt	*None, Primary, Secondary, AcrossPrimary*				X	17
Range.StepSize	*dB*	0	X	X	X	14
Range.Width	*dB or Hz*	0		X		14
Repetitions	*#*	1		X	X	14
Results (adaptive)	*ear ; method ; responsecat ; altcat ; primarystim ; secondarystim*			X	X	29
Results (fixed-frequency)	*freq ; ear ; method ; responsecat ; altcat ; primarystim ; secondarystim ; "response"*			X	X	29
Results (fixed-performance)	*perf ; ear ; method ; responsecat ; altcat ; primarystim ; secondarystim*			X	X	29

StepSize.Initial	d*B*	10	X		X	17
StepSize.Final	*dB*	5	X	X	X	17
Stimuli.Masker	*Stim file* ; *level* ; *ear ; amplitude method*		X	X	X	31
Stimuli.Primary (nbn)	*nbn;centerfreq;bw;duration;rise;fall;ear*		X	X	X	19
Stimuli.Primary (nbp)	*nbp;centerfreq;bw;duration;interval;rise;fall;ear*		X	X	X	19
Stimuli.Primary (pulse)	*pulse* ; *freq* ; *duration*; *ISI*; *rise*; *fall*; *ear*		X	X	X	20

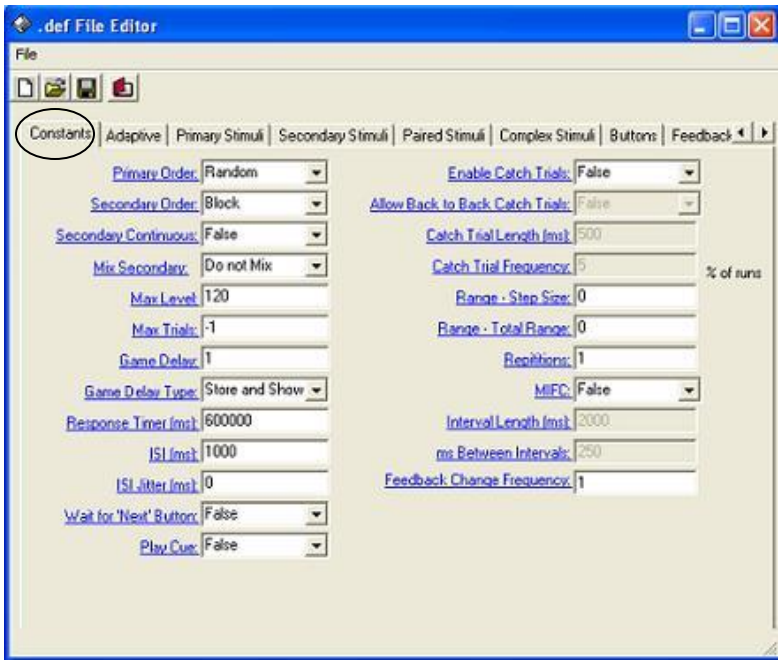
Parameter	**Arguments**	**Default**	**Audiogram**	**Exp. Des. fixed**	**Exp. Design adaptive**	**Page in manual**
Stimuli.Primary (tone)	*tone* ; *freq* ; *duration*; *rise* ; *fall*; *ear*		X	X	X	20
Stimuli.Primary (wave)	*wave* ; *filename* ; *category ; INFO* ; *level* ; *ear ; amplitude method*		X	X	X	21
Stimuli.Secondary (nbn)	*nbn;centerfreq;bw;duration;interval;rise;fall;ear*			X	X	22
Stimuli.Secondary (nbp)	*nbp;centerfreq;bw;duration;interval;rise;fall;ear*			X	X	23
Stimuli.Secondary (pulse)	*pulse* ; *freq* ; *duration*; *ISI*; *rise*; *fall*; *ear*			X	X	23
Stimuli.Secondary (tone)	*tone* ; *freq* ; *duration*; *rise* ; *fall*; *ear*	required		X	X	23
Stimuli.Secondary (wave)	*wave* ; *filename* ; *category ; INFO* ; *level* ; *ear ; amplitude method*			X	X	24
Stimuli.SecondaryContinuous	*0,1*	0		X	X	15
Stop.Mode	*0,1,2*	0			X	18
Stop.PrimaryReversals	*#*	5			X	17
Stop.SecondaryReversals	*#*	8			X	18
Stop.StandardError	*dB*	1.8	X		X	18
Timer	*ms*	3000	X	X	X	15
WaitForNextButton	*0,1*	0		X	X	16

PARAMETER SCREENS

In the following sections, all parameters are grouped according to their corresponding tab in the graphical interface. The name and arguments as listed in the definition file are included in brackets [] if different from the listing in BART. The default setting for each parameter is listed with that parameter.

CONSTANTS

The parameters under the *Constants* tab apply to both fixed and adaptive experiments. Fields for parameters that are not active will be gray.



Primary Order [Order.Primary] (Fixed [0], Random [1]) Specifies the order of presentation of the Primary stimuli. Options are *Fixed* and *Random*. When the *Fixed* is selected, the primary stimuli are played in the order specified in the definition file without replacement. If *Random* is selected, stimuli are randomized, and then played without replacement. If more than one presentation is specified in the *Random* mode, the program generates a list containing the correct number of stimulus repetitions. The default setting is *Random*.

Secondary Order [Order.Secondary] (Block, InOrder, Random, RandomTrial, RandomInterval) Specifies the order of presentation of the Secondary stimuli. The default setting is *Block*.

`Block` = fixed order (as listed). In an ADAPTIVE paradigm, the same secondary stimulus will be used until a threshold is found

`InOrder`= fixed order (as listed) but with a new secondary stimulus with each new trial.

`Random` = random order. In an adaptive situation, the same secondary will be used until a threshold is found.

`RandomTrial` = only enabled in a Multiple-Interval-Forced Choice (***MIFC*****)** situation, *RandomTrial* randomly selects a new secondary stimuli for each trial. It will use the same secondary stimuli for every interval in that trial.
`RandomInterval` = only enabled in an ***MIFC*** situation, *RandomInterval* selects new secondary stimuli for each interval (randomization without replacement).

Secondary Continuous [Stimuli.SecondaryContinuous] (True [1], False [0]) Set to *True* to play secondary stimuli continuously. The default setting is *False*.

```
Stimuli.SecondaryContinuous:  1
```

Mix Secondary [Mix.Secondary](Mix[0],Do Not Mix[1]) Mix secondary with primary. Default setting is Do Not Mix.

Max Level [MaxLevel] (dB) Defines the upper stimulus limit in an adaptive procedure. If a NO response occurs 3 times when the signal is at *MaxLevel*, the trial will terminate and record the response as *MaxLevel + 10 dB.* The default setting is *120 dB.*

Max Trials [NumberOfTrials] ( ) Sets a specific number of trials when using the *FIXED* paradigm. If this parameter is not set, the number of trials will be determined by *# of Primary stimuli x # of Secondary stimuli x Repetitions* or *# of Primary Stimuli x Repetitions* if there are no secondary stimuli. The default setting is *-1 (null).* At this setting, the trial will not stop based on number of trials.

In some cases, reinforcement for correct responses may be undesirable (to avoid learning effects in a post test condition). To maintain interest in the task, reinforcement can be provided on a schedule that is not linked to performance. The parameters, *Game Delay Type* and *Game Delay* allow the user to customize reinforcement.

Game Delay [GameDelay] (#) Delays the reinforcement by the number of trials specified. If trial-by-trial reinforcement is required set the parameter to 1. If delayed reinforcement is necessary, the argument can be set to any number up to the total number of trials in the experiment. For example, if GameDelay is set to 5, after every 5 trials the game will advance based on the selection for *GameDelayType*. The default setting is *1 trial*.

```
GameDelay: 1
```

Game Delay Type [GameDelayType] (Store and Show [0], One Step every X [1]) determines the number of steps the game will advance. The default setting is *Store and Show*.

```
GameDelayType: 1
```

If set to *Store and Show [0]*, the program will count the number of correct responses until after [*GameDelay*] number of trials, and then give the subject 1 step in the game for every correct response. This option is most appropriate for puzzles, dot-to-dot, and zap games. It does not work well with slideshow games.

If set to *One Step every X [1]*, the program will wait until [*GameDelay*] number of trials have occurred, and then give the subject 1 step in the game, regardless of the number of correct responses. This option is appropriate for any type of game.

EXAMPLE:

```
GameDelay: 3
GameDelayType: 1
```

The game would advance 1 step after every 3 trials, regardless of the number of correct responses.

```
GameDelay: 3
GameDelayType: 0
```

Assuming that over those 3 trials the subject had 2 correct responses, the game would advance 2 steps after the third trial.

Response Timer [Timer] (ms) Defines the time period the program will wait before a 'no response' is recorded. In the audiogram function, the lack of a response is considered to be a stimulus below threshold and the level is increased. In experiments in which a response is mandatory and likely to have considerable cognitive demands (e.g., repeating sentences at poor SNRs), this value may need to be increased substantially. Timer and signal are started at the same time. If the stimulus and the timer have the same duration, then the ISI will start immediately at the stimulus offset. Timer duration should be sufficiently long to allow subjects to respond. The default value in the audiogram function is *3000 ms*. The default setting for all other experiments is *600000 ms*. **NOTE:** buttons are locked and the user CANNOT respond during the ISI. See Appendix A for more details on stimulus timing.

ISI [ISI.Length] (ms) Defines the inter-stimulus interval between the response to the previous trial and the presentation of the next stimulus. The default setting is *1000*.

ISI Jitter [ISI.Jitter] (ms) Defines the magnitude of variation for the ISI in a *SINGLE INTERVAL* experiment. This function is designed to reduce the number of false positive responses by adding some random fraction of the defined value to the ISI. The default setting is *0 ms*.

```
ISI.Jitter: 50
```

Wait For 'Next' Button [WaitForNextButton] (True [1], False [0]) If *True [1]* is selected **AND** the *Response Category* of the button is set to *NextStim*, the program will wait at the end of each trial for the user to click the NextStim button, instead of automatically proceeding into the next trial. The default setting is *False*.

If is set to *True [1]*, a button **MUST** be declared similar to:

```
%BUTTON: XSIZE ; YSIZE ; XLOC ; YLOC ; FontSize ; Response Category ; TEXT
; Picture file (full path)

button: 80 ; 30 ; 40 ; 300 ; 12 ; NextStim ; Next ;
C:\gamepics\shannon.bmp
```

Play Cue [Interval.Cue] (True [1], False [0]) Typically used only in an ***MIFC*** situation. When *True*, the primary stimuli will be played before each trial, to cue the listener. The default setting is *False*.

NOTE: If ***Play Cue*** is set to *True* and the paradigm is MIFC, the delay between the cue and stimulus presentation will be equal to the value set for ***ms between intervals***. If ***Play Cue*** is set to *True* and the paradigm is NOT MIFC, there will be no delay between the cue and stimulus presentation.

Enable Catch Trials [CatchTrial.Enabled] (True [1], False [0]) Specifies whether catch trials should be included in the *ADAPTIVE* paradigms. The default settingis *False*.

The following parameter fields are active only when ***Enable Catch Trials*** is *True*.

Allow Back To Back Catch Trials [CatchTrial.BackToBack] (True [1], False [0]) When *True*, allows multiple catch trials to occur, one right after the other. When *False*, prevents two or more catch trials from occurring in sequence. The default setting is *False*.

Catch Trial Length [CatchTrial.Length] (ms) Defines the length of the catch trials. For pure tones and pulsed tones, the catch trial length should be similar to the duration of the generated tone. If using wave files, the duration must be specified (the length of the catch trial should be similar to the stimulus duration). The default setting is *500 ms*.

Catch Trial Frequency [CatchTrial.Frequency] (%) Specifies the frequency with which the catch trials will occur in every 100 trials. The default setting is *5%*.

For example:
`CatchTrial.Frequency: 10` = 10% of all trials will be catch trials.

Range – Step Size [Range.StepSize] (dB) Specifies the step size for the first two reversals in the adaptive tracking procedure. The default setting is *0 dB*.

Range - Total Range [Range.Width] (dB) For fixed paradigms, the *Range.Width* paradigm is used in conjunction with the parameter *Range.StepSize* to define the number of fixed intervals to be tested (the default setting is *0 dB*). For example, if the starting level of a stimulus is 75 dB and performance at 65, 55, and 45 is of interest, then the *Range.StepSize* parameter would be set to 10 dB and *Range.Width* to 30 as follows:

```
Range.StepSize: 10
Range.Width: 30
```

Repetitions (#) Defines the number of repetitions of the stimuli. In FIXED paradigms, the stimuli will be repeated the specified number of times. In ADAPTIVE paradigms, stimuli will be presented until performance specified in the parameter *Method* (e.g. 50%, 71%, 29%) is reached. Unless otherwise necessary, it is recommended that the *Repetitions* parameter be set to 1 for ADAPTIVE paradigms. If all of the stimuli have been presented and the performance level has not been achieved, the program will reset to the beginning of the stimuli again. The default setting is *1 repetition*.

MIFC [Interval.Buttons] (True [1], False [0]) Enables multiple interval forced choice trials. The default setting is *False*.

The following two parameter choices are active on the Constants screen when MIFC is *True*.

Interval Length [Interval.TimeOn] (ms) duration (in ms) that each button will be lit (color changes from gray to white). The default setting is *2000 ms*.

```
Interval.TimeOn: 2000
```

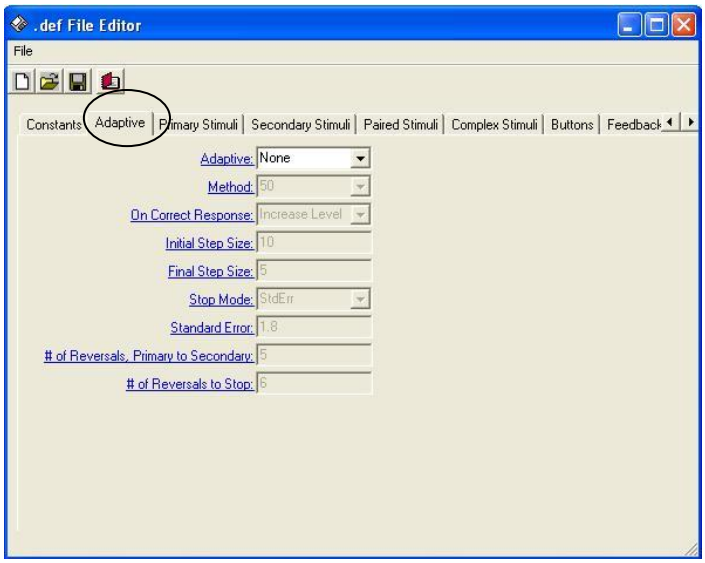
ms Between Intervals [Interval.Interval] (ms) duration (in ms) between intervals. The default setting is *250 ms*.

```
Interval.Interval: 200
```

Feedback Change Frequency: determines how often the feedback picture should be changed. Eg Value of 2 changes feedback picture after every 2 stims. Default setting is 1.

ADAPTIVE

This screen allows the user to select the parameters for an ADAPTIVE paradigm. If *Adaptive* is set to *None*, the paradigm is fixed and the remaining fields are not active.



Adaptive [PrimaryAdapt] (None, Primary, Secondary, Across Primary) selects settings for the adaptive paradigm. The default setting is *None*.

None is chosen when using a fixed paradigm.
Primary adapts the level of the primary stimuli.
Secondary holds the primary stimuli static and adapts the level of the secondary stimuli.

Across Primary selects a new primary stimulus each trial, and applies the adaptive track to it.

Method (50,70,30) Specifies the level of performance to be determined through the adaptive paradigm. The reader should refer to Levitt (1970) for a review of the trial-by-trial procedures used to determine these points. The default setting is *50%*.

50 = 0.5 (50%)
70 = 0.707 (71%)
30 = 0.293 (29%)

On Correct Response [DirOfChange] (Decrease Level [-], Increase Level [+]) Defines the direction of the level for ADAPTIVE stimuli. The default setting is *Increase Level*.

On Correct Response: Increase Level increases signal level of the adapting stimulus (primary or secondary) on correct responses.
On Correct Response: Decrease Level decreases signal level of the adapting stimulus (primary or secondary) on correct responses.

Initial Step Size [StepSize.Initial] (dB) Specifies the step size for the first two reversals in the adaptive tracking procedure. The default setting is *10 dB*.

Final Step Size [StepSize.Final] (dB) Specifies the step size for the remaining reversals in the adaptive tracking procedure. The default setting is *5 dB*.

Stop Mode [Stop.Mode] (StdErr [0], Reversals [1], Both[2]). When *StdErr*, adaptive procedures will be stopped based on the *Standard Error* criteria. When *Reversals*, adaptive procedures will be stopped based on the number of reversals indicated by the *Number of Reversals to Stop* parameter. When *Both*, will be stopped using either *Standard Error*, or *Number of Reversals to Stop*, whichever criterion is reached first. The default setting is *StdErr*..

Standard Error [Stop.StandardError] (dB) The decibel value defining the stopping rule for adaptive procedures. The choice of a standard error criterion will depend upon the task, the expected variability, and subject characteristics. This value may be any number, however, a standard error between 1 and 2 dB is commonly used. The default is *1.8 dB*. Lower values will increase the number of trials necessary to meet the criteria.

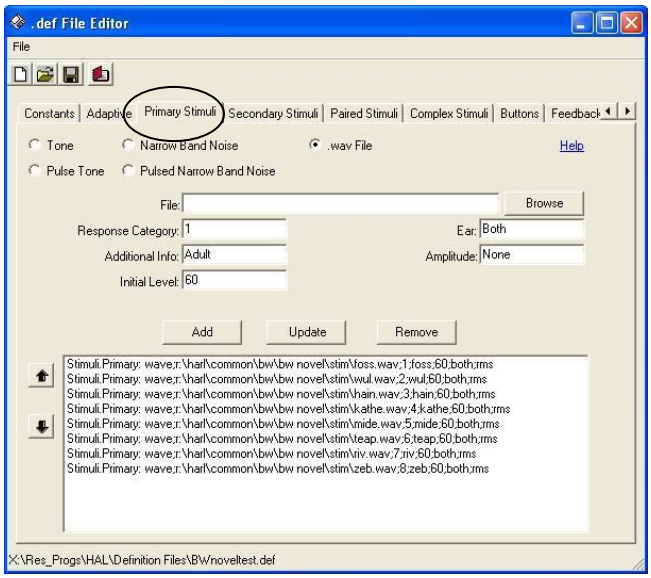
Number of Reversals, Primary to Secondary [Stop.PrimaryReversals] (#). In an ADAPTIVE task, the number of reversals changing from the primary to the secondary step size. The default is *5 reversals*.

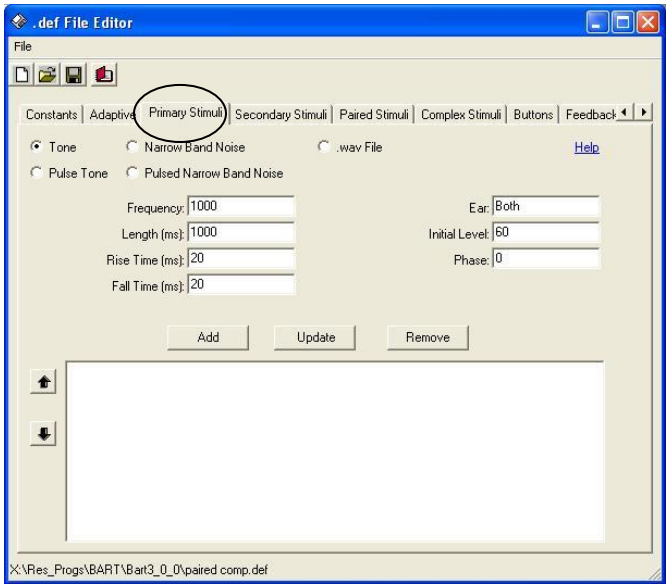
Number of Reversals to Stop [Stop.SecondaryReversals] (#). In an ADAPTIVE task, the number of reversals required to stop when the *Stop Mode* parameter is set to *Reversals [1]*. The default setting is *6 reversals*.

PRIMARY STIMULI

This screen allows the user to set the values for the primary stimuli. The active fields will vary depending on the type of stimulus chosen (tone, narrow band noise, .wav file, pulse tone, pulsed narrow band noise).

The panels below show the screens for *.wav file* and *Tone*.





Three buttons allow modification of the stimuli:
Add—Use this button to add a stimulus that includes the parameters in all of the active fields.
Update—Highlight a stimulus from the list and click on this button to change to the parameters in the active fields
Remove—Highlight a stimulus from the list and click on this button to remove it.

Narrow Band Noise [Stimuli.Primary:nbn] Several arguments are passed into the *Stimuli.Primary ( )* parameter in the following order:

```
stimuli = nbn (narrowband noise)
center frequency = specified in Hz
bandwidth = in fraction of an octave (e.g., 1=full; 3=1/3)
duration = specified in ms
rise time = specified in ms
fall time = specified in ms
ear = right, left, or both (presented binaurally)
level = specified in dB (rms) [for adaptive paradigms = initial
level; for fixed paradigms = fixed level]
```

Stimuli.Primary: nbn ; center freq ; bw(in fraction of octave) ; duration ms ; rise ms ; fall ms ; ear ; level (dB)

```
Stimuli.Primary: nbn ; 1000 ; 3 ; 1000 ; 20 ; 20 ; both ; 80
```

Pulsed Narrow Band Noise [Stimuli.Primary;nbp] Several arguments are passed into the *Stimuli.Primary ( )* parameter in the following order:

```
stimuli = nbp (narrowband pulsed)
center frequency = specified in Hz
bandwidth = in fraction of an octave (e.g., 1=full; 3=1/3)
duration of each pulse = specified in ms
interval between pulses = specified in ms
rise time = specified in ms
fall time = specified in ms
ear = right, left, or both (presented binaurally)
level = specified in dB (rms) [for adaptive paradigms = initial
level; for fixed paradigms = fixed level]
```

Stimuli.Primary: npb ; center freq ; bw(in fraction of octave) ; duration of each pulse; interval ; rise ; fall ; ear ; level (dB)

```
Stimuli.Primary: nbp ; 1000 ; 3 ; 300 ; 50 ; 20 ; 20 ; both ; 80
```

NOTE: There are always 3 pulses per segment.

Pulse Tone [Stimuli.Primary:pulse] Several arguments are passed into the *Stimuli.Primary ( )* parameter in the following order:

```
stimuli = pulse
frequency = specified in Hz
duration =  specified in ms
interstimulus interval = specified in ms
rise time = specified in ms
fall time = specified in ms
ear = right, left, or both (stimuli are presented binaurally)
level = specified in dB (peak) [for adaptive paradigms = initial
level; for fixed paradigms = fixed level]
```

Stimuli.Primary: pulse ; freq ; duration of each pulse ; interval ; rise ; fall ; ear ; level (dB)

```
Stimuli.Primary: pulse ; 1000 ; 300 ; 100 ; 20 ; 20 ; right ; 80
```

NOTE: There are always 3 pulses per segment.

Tone [Stimuli.Primary:tone] Several arguments are passed into *the Stimuli.Primary ( )* parameter in the following order:

```
stimuli = tone
frequency = specified in Hz
duration =  specified in ms
rise time = specified in ms
fall time = specified in ms
ear = right, left, or both (presented binaurally)
```

```
level = specified in dB (peak) [for adaptive paradigms = initial
level; for fixed paradigms = fixed level]
```

Stimuli.Primary: tone; freq; duration ms ; rise ms; fall ms; ear; level (dB)

```
Stimuli.Primary: tone ; 250 ; 1000 ; 20 ; 20 ; left ; 80
```

.wav file [Stimuli.Primary;wave] Several arguments are passed into the *Stimuli.Primary ( )* parameter in the following order:

```
stimuli = wave
filename = *.wav
response category = should correspond to Response Category in button
     line
altcat or INFO (additional info) = 2 functions for INFO: 1) info
     associated w/ stimuli or 2) when using an adaptive procedure
     based on filtering, the INFO parameter becomes the initial
     cutoff frequency
stimulus level = specified in dB (for adaptive paradigms = initial
level; for fixed paradigms = fixed level)
ear = right, left, or both (stimuli are presented binaurally)
amplitude method = peak, rms, or none
```

NOTE: For the argument *amplitude method*, if the wave file contains a pure tone, the *Peak* option should be selected. If the wave file contains speech or noise, the *RMS* option should be selected. *None* is selected with any externally calibrated wave file.

For adaptive paradigms, e.g., one in which wave files are low-pass filtered at several corner frequencies, the stimuli would be defined as follows:

Stimuli.Primary: wave ; *filename* ; *category ; INFO* ; *level* ; *ear ; amplitude method*

```
Stimuli.Primary: wave; xxx.wav ; 1 ; 6000 ; 65 ; right ; rms
Stimuli.Primary: wave; yyy.wav ; 2 ; 6000 ; 65 ; right ; rms
Stimuli.Primary: wave; zzz.wav ; 3 ; 6000 ; 65 ; right ; rms
```

For fixed paradigms, the corner frequency argument (INFO) will be ignored and may be used to note supplemental information (altcat) in the results file or it can be left blank as shown in the following examples:

Stimuli.Primary: wave ; *filename* ; *category ; altcat* ; *level* ; *ear ; amplitude method*

```
Stimuli.Primary: wave; xxx.wav; s ; female; 60 ; right; rms
Stimuli.Primary: wave; yyy.wav; f ; child; 60 ; right; rms
Stimuli.Primary: wave; zzz.wav; t ; male; 60 ; right; rms
```

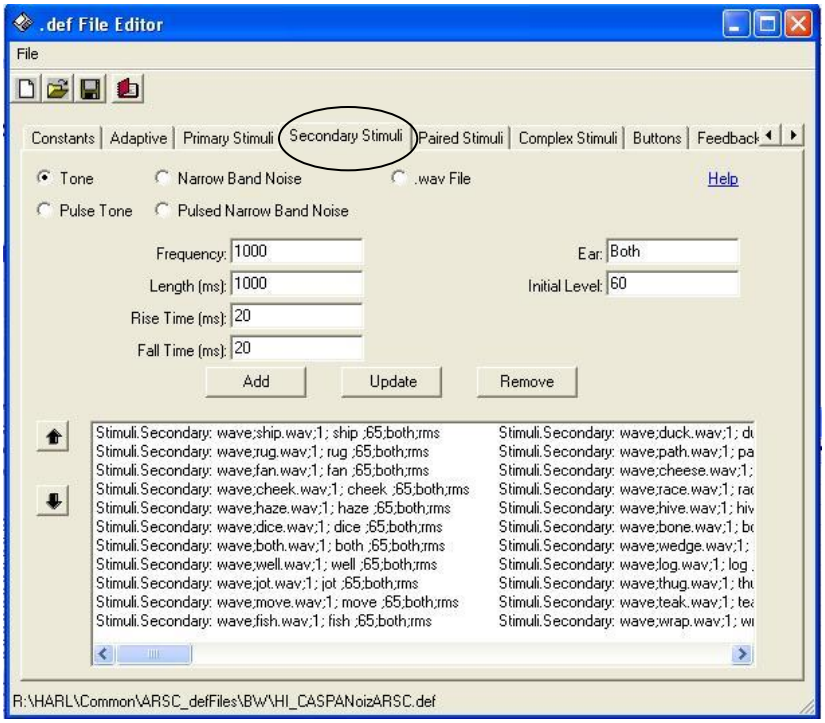
Or

```
Stimuli.Primary: wave; xxx.wav ; 1 ; ; 45 ; both ; rms
Stimuli.Primary: wave; yyy.wav ; 2 ; ; 45 ; both ; rms
Stimuli.Primary: wave; zzz.wav ; 3 ; ; 45 ; both ; rms
```

NOTE: For the argument *amplitude method*, if the wave file contains a pure tone, the *Peak* option should be selected. If the wave file contains speech or noise, the *RMS* option should be selected. *None* is selected with any externally calibrated wave file.

SECONDARY STIMULI

As shown below, the secondary stimuli declaration works exactly the same as a primary stimuli declaration.



Narrow Band Noise [Stimuli.Secondary:nbn] See page ___ for a description of the arguments passed into this parameter.

Stimuli.Secondary: nbn ; center freq ; bw(in fraction of octave) ; duration ms ; rise ms ; fall ms ; ear ; level (dB)

```
Stimuli.Secondary: nbn ; 1000 ; 3 ; 1000 ; 20 ; 20 ; both ; 80
```

Pulsed Narrow Band Noise [Stimuli.Secondary:nbp] See page ___ for a description of the arguments passed into this parameter.

Stimuli.Secondary: npb ; center freq ; bw(in fraction of octave) ; duration of each pulse; interval ; rise ; fall ; ear ; level (dB)

```
Stimuli.Secondary: nbp ; 1000 ; 3 ; 300 ; 50 ; 20 ; 20 ; both ; 80
```

Pulse Tone [Stimuli.Secondary:pulse] See page ___ for a description of the arguments passed into this parameter.

Stimuli.Secondary: pulse ; freq ; duration of each pulse ; interval ; rise ; fall ; ear ; level (dB)

```
Stimuli.Secondary:pulse ; 1000 ; 300 ; 100 ; 20 ; 20 ; right ; 80
```

Tone [Stimuli.Secondary:tone] See page ___ for a description of the arguments passed into this parameter.

Stimuli.Secondary: tone; freq; duration m ; rise ms; fall ms; ear; level (dB)

```
Stimuli.Secondary:  tone ; 250 ; 1000 ; 20 ; 20 ; left ; 80
```

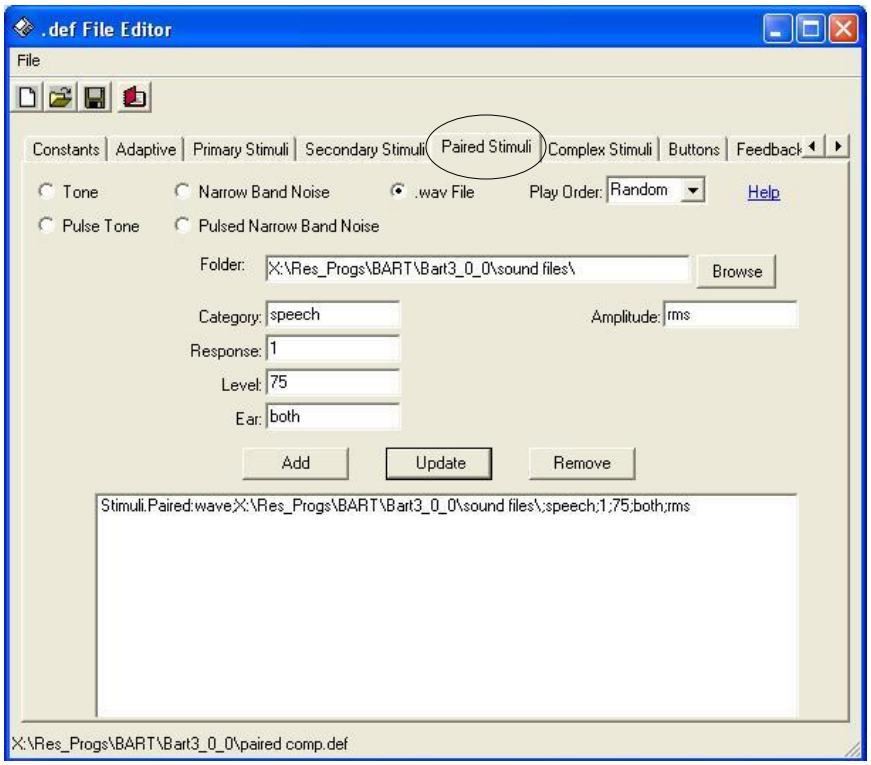
.wav file [Stimuli.Secondary;wave] See page ___ for a description of the arguments passed into this parameter.

Stimuli.Secondary: wave; filename; category; INFO; level; ear(s); amplification method

```
Stimuli.Secondary: wave ; test.wav ; s ; 1 ; 80 ; both ; none
```

NOTE: It is not possible to present different primary or secondary stimuli to each ear on the same trial. Also, stimuli from the left and right channels cannot be mixed digitally and presented to one ear. These options may become available in future versions.

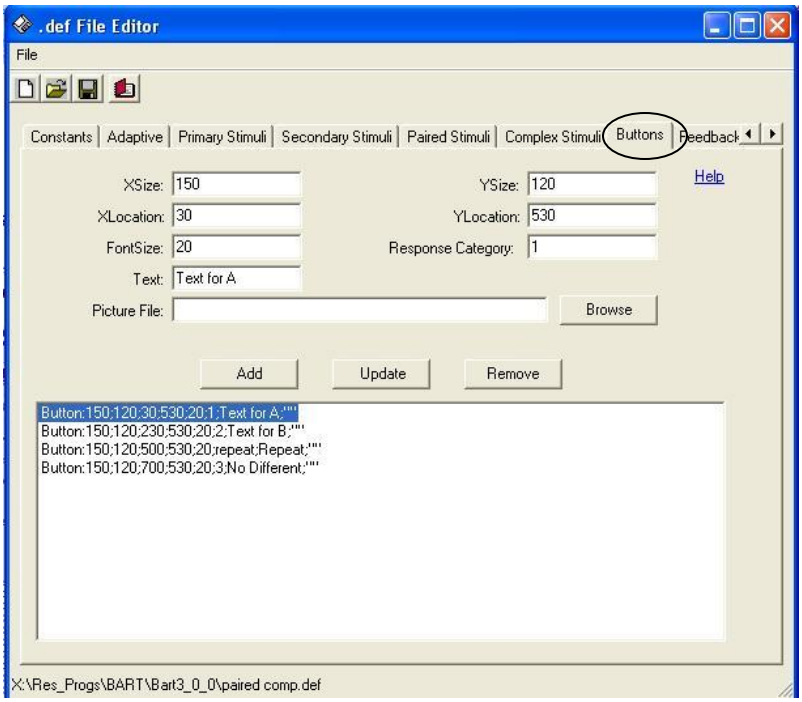
PAIRED STIMULI



See pages 29-31 for more details on Paired Comparison.

BUTTONS

This screen allows the user to select the size and location of response buttons as well as any text or pictures that will appear on the buttons.



Button ( ) Defines the parameters for response buttons. When multiple buttons are used in an experiment, the parameters for each button should be defined on a separate row.

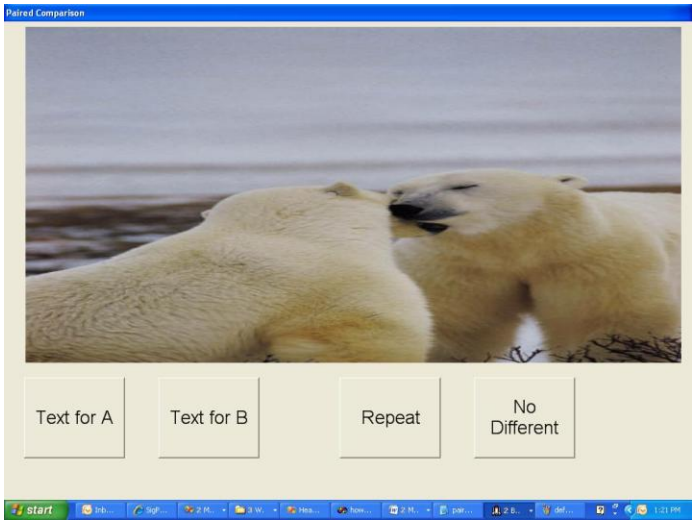
EXAMPLE:
Button: *XSIZE* ; *YSIZE* ; *XLOC* ; *YLOC* ; *FontSize* ; *Response Category* ; *Text* ; *PictureFile (full path)*

Button: 150 ; 120 ; 30 ;530 ; 20 ; 1 ; Text for A ; ""

Button: 150 ; 120 ; 230 ;530 ; 20 ; 2 ; Text for B ; ""

Button: 150 ; 120 ; 500 ;530 ; 20 ; repeat ; Repeat ; ""

Button:150 ; 120 ; 700 ;530 ; 20 ; 3 ; No Different ; ""



XSIZE and *YSIZE* define the size of the button in pixels.

XLOC and *YLOC* specify the location of the button relative to the upper left corner of the screen. A button

placed at the edge of the upper left corner will have *xloc* and *yloc* coordinates of 0 ; 0. Larger x-values move the button to the right and larger y-values move the button down.

FontSize denotes the size of the text assigned to that button in points. Values of 10 to 20 points are typical. *Response Category* should correspond to those listed under *Stimuli.Primary*. Buttons corresponding to response categories not defined under Stimuli can be assigned if necessary. For example, a button category of 'Other' can be specified and not associated with a stimulus file.

Text specifies the text assigned to the button.

Picture is the path to the .bmp file to be used for this button. The size of the picture must be equal to or less than the size of the button in pixels. This can be verified by copying the picture into Paint.

Repeat Button A special condition under this parameter has been provided to specify a repeat button. If provided, the subject may repeat the current stimulus as many times as desired. To program this button the *Response Category* argument must contain the word ' repeat' as in the example below. All other arguments remain the same.

Button: 150 ; 120 ; 500 ;530 ; 20 ; repeat ; Repeat ; ""

NOTE: With both the *Repeat* button and the *Respond* button it is imperative that the 'Category' field (6th entry) of the button declaration line be designated as either 'repeat' or 'respond'.

Once the intervals have been defined, the buttons must be defined. The number of buttons will determine the number of intervals to be included. For Example:

Interval Parameters are:

```
Interval.Buttons: 1
Interval.TimeOn: 2000
Interval.Interval 500
```

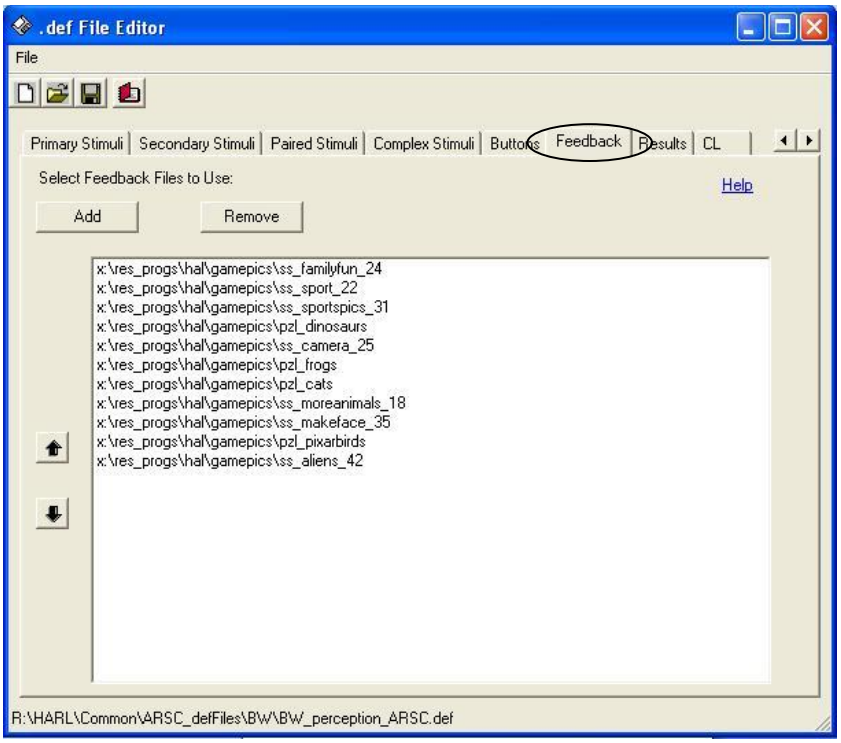
Button: XSIZE ; YSIZE ; XLOC ; YLOC ;FontSize; Response Category ; TEXT ; Picture file (full path)

```
button: 80 ; 30 ; 40 ; 50 ; 18; 1 ; 1 ; C:\gamepics\camera1.bmp
button: 80 ; 30 ; 40 ; 90 ; 18; 2 ; 2 ; C:\gamepics\camera2.bmp
button: 80 ; 30 ; 40 ; 130 ; 18; 2 ; 3 ; C:\gamepics\camera3.bmp
button: 80; 30; 40; 170; 18; respond; Respond; C:\gamepics\camera4.bmp
```

In this example, we will have three intervals, indicated 1, 2, and 3. Button 1 will light for 2000 ms, play the stimulus, and turn off. After a 500 ms delay (during which all buttons are 'off'), the next button will light and the next stimulus will play, etc. After all of the intervals, the 'Respond' button will light and the program will wait for the user to click button 1, 2, or 3. Clicking the 'Respond' button doesn't do anything. If there are only Primary Stimuli, then one interval will have the primary stimuli, and the others will be silent. If there are primary and secondary stimuli, then each interval will have the secondary stimuli, and only one will have the primary stimulus.

FEEDBACK

This screen allows the user to select the games that will be used as feedback during the experiment.

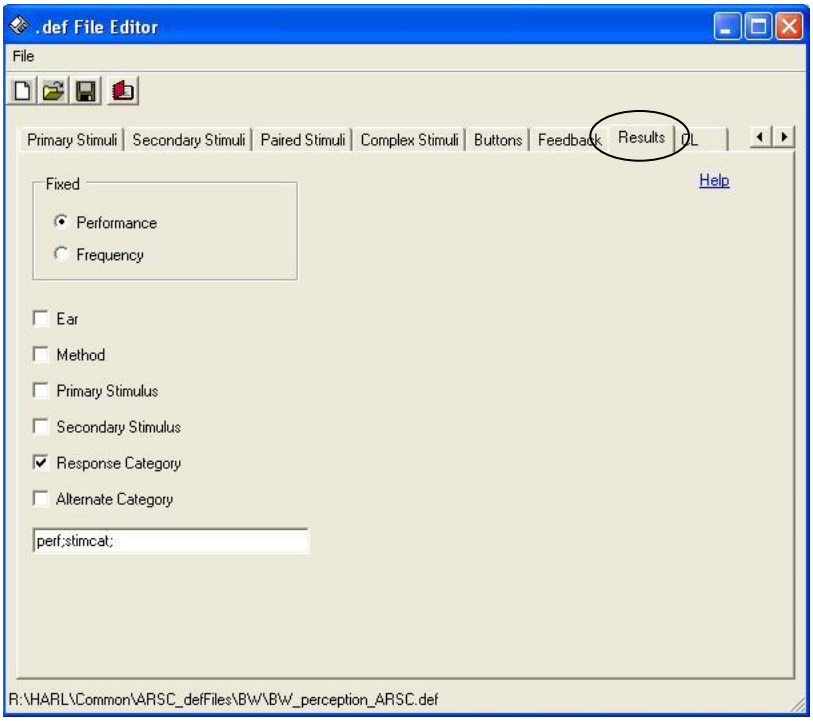


Feedback.File (complete path) lists the games to be used during testing. Currently there are a number of games available (see Appendix B).

```
FeedbackFile: x:\res_progs\hal\gamepics\ss_sport_22
```

RESULTS

This screen allows the user to select how results will be displayed in the Results file.



Results ( ) Results may be summarized in a number of ways. The *Experiment* results screen displays updated results with every trial. At the end of the program, the results are written to the Results file. If the program is ended prematurely, the results may be written to the file using the '*Write Summary*' button.

For *ADAPTIVE* paradigms, any order of the arguments in the *Stimuli.Primary* and/or *Stimuli.Secondary* parameters may be used:

ear
method
responsecat
altcat
primarystim
secondarystim

```
Results: ear ; responsecat
```

For *FIXED* paradigms, two types of analysis are available:

perf (performance in % correct)
freq (frequency # selected out of total presentations)

The type of analysis must be the first paradigm listed. Then, any order of the parameters may be used. When 'freq' is specified, the parameters must be followed by the term 'response'.

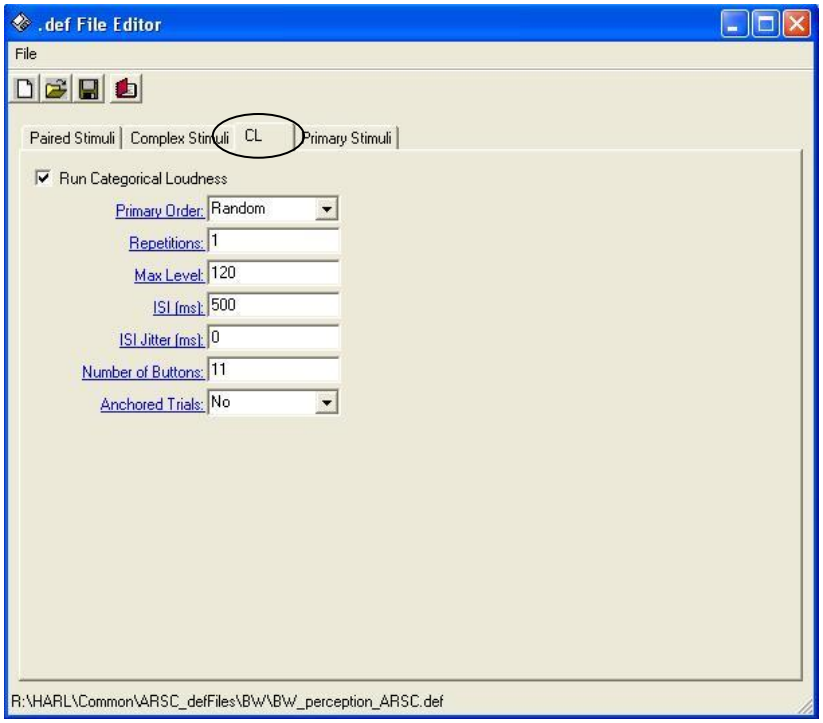
```
Results: perf ; ear ; primarystim

Results: freq ; responsecat ; response
```

The '*altcat'* parameter can be used to report results in terms of a second parameter. Such as a set of stimuli in which the target words are spoken by three different talkers (e.g., adult male, adult female, child). For the *Results* parameter, the *Response Category* would be the correct button for each word and the *altcat* would be the talker. For a fixed paradigm the following would provide percent correct based on talker:

```
Results: perf; altcat
```

CL (CATEGORICAL LOUDNESS)

When the Run Categorical Loudness box is checked, only four tabs will be available (Paired Stimuli, CL, Primary Stimuli, Complex Stimuli)



See the following pages for definitions of :
Primary Order (page ___)
Max Level (page ___)
ISI (page ___)
ISI jitter (page ___)

Number of Buttons [CLButtons] (#) determines how many buttons will appear on the subject screen.

ADDITIONAL PARAMETERS

Stimuli.Masker ( ) It is necessary at times to present masking noise to the non-test ear throughout the experiment. The parameter *Stimuli.Masker* provides for continuous masking from the first through the last trial. Three arguments are required:

`stimulus` = wave file containing masker (provided by user)
`level` = presentation level of the masker (in dB)
`ear` = right, left, both (presented binaurally)
`amp method` = rms or peak

Stimuli.Masker: filename; level; ear; amplitude

```
Stimuli.Masker: noise.wav ; 60 ; right ; peak
```

NOTE: If this parameter is needed, it must be typed directly into the definition file via Notepad. It cannot be added via the tabbed screens in BART.

PAIRED COMPARISON (PC):

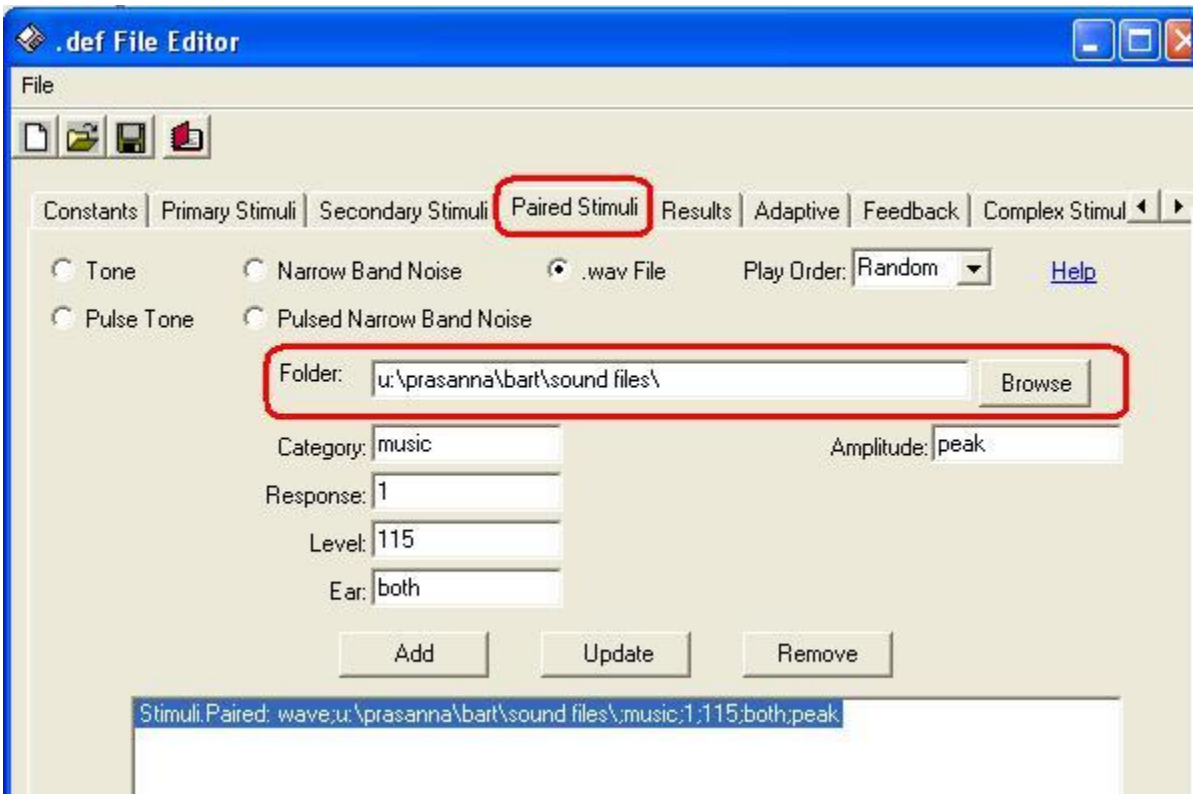
As of version 3.0, Paired Comparison module has been added to BART**.**

Follow these steps before using PC.

a) Change/Prepare Definitions file:
There is a separate tab for configuring PC module. Open the def editor.

Catch trial frequency is the % of catch trial runs based on total number of runs including catch trials.

Interval Length is the time between one pair of runs to the next.
ms Between Intervals is the time between two sounds in the same pair.
Feedback Change Frequency is the frequency at which feedback picture is presented. For example if this value is 2, then the user has to play two runs of paired comparison before a feedback picture is displayed.



Paired Stimuli Tab has options to configure paired comparison files.

Folder/Browse allows you to locate the folder where all the paired comparison sound files are located. All files should be put in a single folder without subfolders.
The program separates files into three groups – EBW, RBW and FC by looking at their filename.
The file naming convention is as : <filename>-<filetype>.wav …for example apple-EBW.wav is of type EBW, whereas ball-RBW.wav is if type RBW and cat-FC.wav is of type FC.

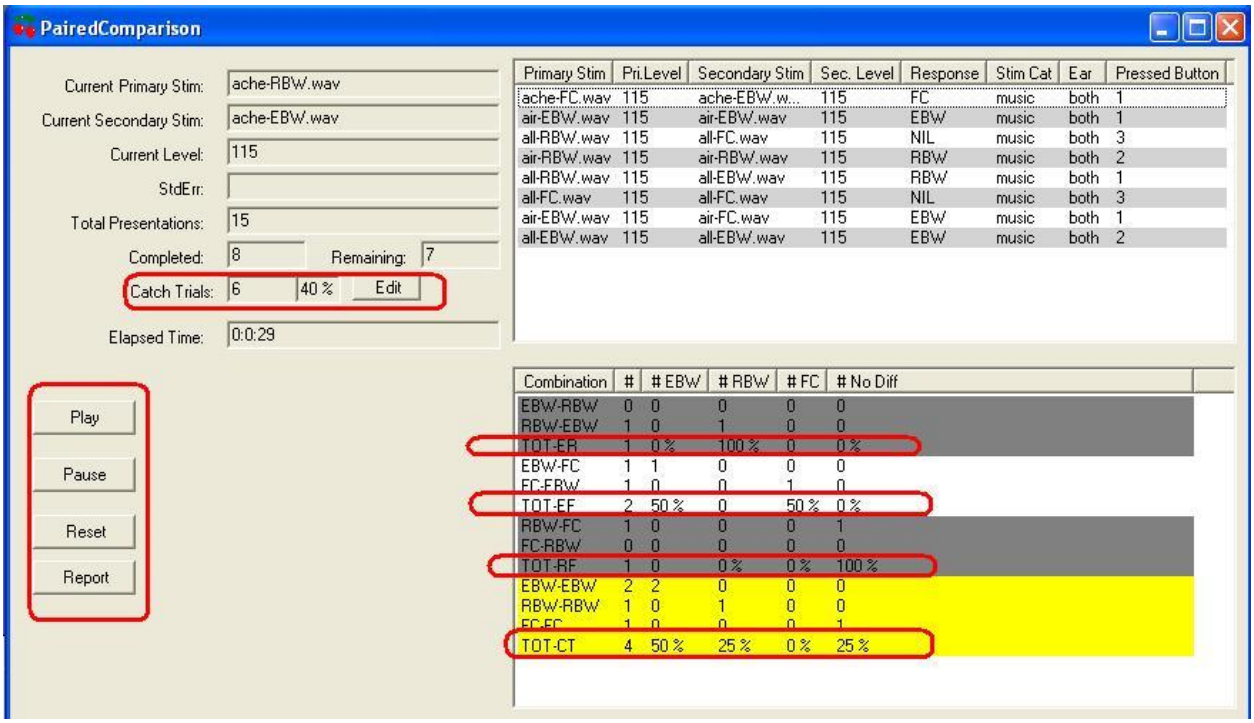
Don't forget to SAVE your work after you set everything.

b) Start Paired Comparison:

Go to Run Programs > Paired Comparison or click button to start Paired comparison.

c) Adjust Catch Trial numbers:

Since catch trial configuration set in definitions file editor works on percentages, you may not always get exact number of catch trials that you need. You may however set exact number of catch trials from Paired Comparison's main control.



Click "Edit" button on the "Catch Trials" row. Text box on the left brightens up and you may now enter the exact number of catch trials on this text box. Apply your changes.

d) Play Paired comparison.

Once the subject is ready, click Play. The paired comparison is now running.

You may Pause, Reset or Report at any time.

Pause pauses the paired comparison run. However, the program expects that response will be provided for the sound file pair that is already in progress.

Reset ignores any paired comparison run so far and brings the system to a point where you can give it a fresh start.

Report writes the results to the results file. However, at the end of game, results are automatically written to results file anyway.

APPENDIX A

STIMULUS TIMING

The following description outlines the various parameters associated with stimulus timing and how they interact.

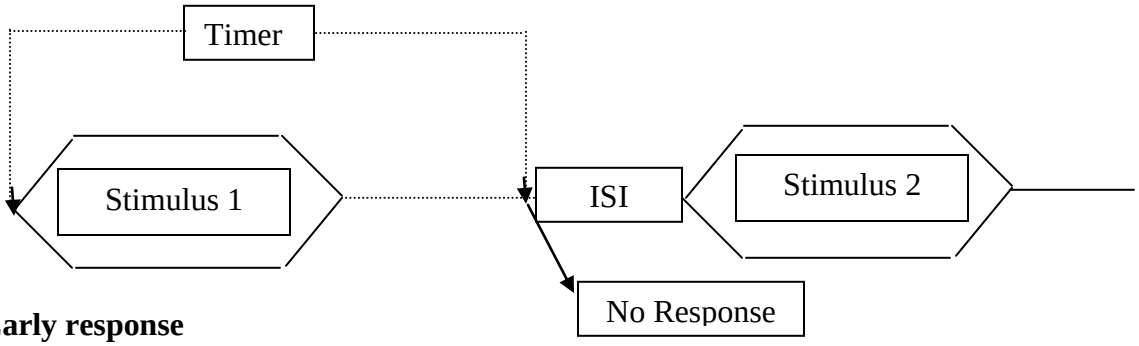
Stimulus Duration (ms): Defined by *Stimuli.Primary*

Timer (ms): Should be equal the Stimulus Duration + allotted Response Time

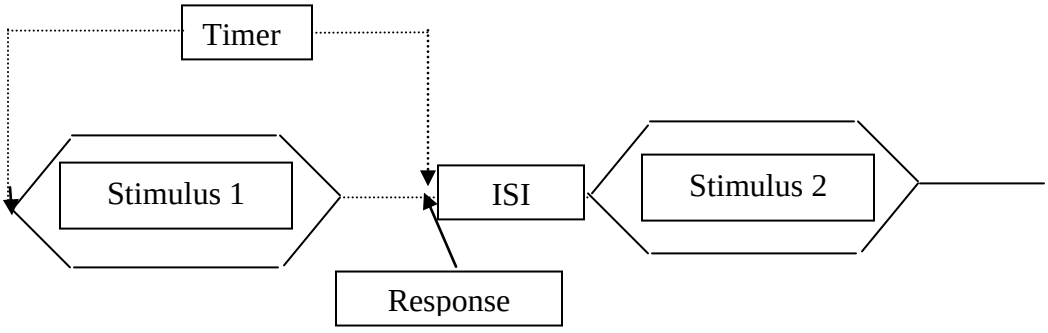
ISI (ms): Defines the interstimulus interval between the response to the previous trial and the presentation of the next stimulus. If the subject does not respond, the *ISI* will start when the *Timer* expires. If and when the subject does respond, the timer is stopped (and reset for the next trial) and the *ISI* starts immediately.

ISI.Jitter (ms): Defines the magnitude of variation for the ISI in a single interval experiment. This function is designed to reduce the number of false positive responses.

Example 1: **No Response**



Example 2: **Early response**



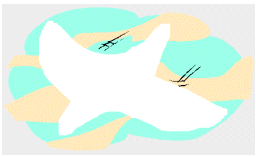
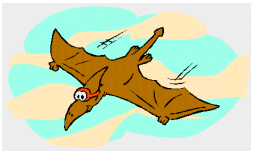
APPENDIX B

GAMES

There are currently four types of games: dot-to-dot, puzzle, zap, and slideshow.

Dot-to-dot

The dot-to-dot text files contain information regarding the source code for the game as well as coordinates for the numbered dots. As the game advances a red line is drawn from one point to the next in order. At the end of the game, the picture is revealed.

	Before	After
d2d_castle.txt		
d2d_ptero.txt		
d2d_sub.txt		
d2d_train.txt		

The first line of the file must be “DOT_TO_DOT”.

Example of the beginning text of a “dot-to-dot” game:

```
DOT_TO_DOT
%Beginning picture
X:\Res_Progs\HAL\gamepics\Dot2dot\blankcastle.bmp;
%ending picture
X:\Res_Progs\HAL\gamepics\Dot2dot\castle.bmp;
```

```
0.15589 0.18583; ... %1
0.14091 0.41163; ... %2
0.11968 0.40964; ... %3
```

Puzzle

The puzzle text files contain information regarding the source code for the game as well as the path to the picture to be revealed. Any bmp file may be used. The game begins with a white screen and progresses by revealing one square at a time.

pzl_sunglasses.txt

The first line of the file must be “REVEAL”.

Example of the text of a “puzzle” game:

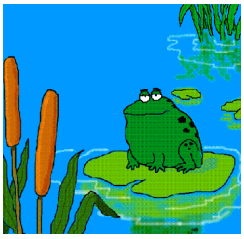
```
REVEAL;
X:\Res_Progs\HAL\gamepics\Puzzled\kid_sunglasses.bmp;
```

Zap

The zap text files contain information regarding the source code for the game as well as the coordinates of the objects being zapped. As the game advances, a red line is drawn from one point to one of the objects being zapped.

disasteroid.txt

catch_a_fly.txt



The first line of the file must be “ZAP”.

Example of the beginning text of a “zap” game:

```
ZAP;
%location of the background picture
X:\Res_Progs\HAL\gamepics\Catcher\toad.bmp;
black;
0.51899 0.51899;...%origin
0.66904 0.94477; ... %1
0.71899 0.88884; ... %2
```

Slide Show

Two types of slide shows have been created. The first type (ss) simply cycles through a series of photographs. The number at the end of the filename indicates the number of slides in that game. The second type (c) cycles through a series of pictures that show a picture being colored.

```
ss_scene_23.txt
```


```
c_sand.txt
```




The first line of the file must be "SLIDESHOW".

Example of the beginning text of a "slideshow" game:

```
SLIDESHOW;
X:\Res_Progs\HAL\gamepics\animals\bear1.bmp;
X:\Res_Progs\HAL\gamepics\animals\bird1.bmp;
X:\Res_Progs\HAL\gamepics\animals\bird2.bmp;
```

The .txt files that run the games as well as the corresponding folders with the appropriate .bmp files are located at:

```
X:\Res_Progs\HAL\gamepics\
```

The format of each txt file is determined by the type of game. As discussed above, the first line designates the game type (DOT_TO_DOT, SLIDESHOW, REVEAL (puzzle), or SLIDESHOW).

Within the definition file, only the txt file name is listed as shown below. The .txt extension should NOT be included. The *feedbackdirectory* must be the same for all of the .txt files used in a given experiment, but the .bmp files can be located in other directories.

```
feedbackfile: c_sand
```

```
feedbackfile: d2d_train
feedbackfile: d2d_castle
feedbackfile: c_play
feedbackfile: catch_a_fly


feedbackdirectory: X:\Res_Progs\HAL\gamepics\
```

APPENDIX C

EXAMPLE DEFINITION FILES

```
% EXAMPLE AUDIOGRAM DEFINITION FILE

%Random presentation order; initial step size 10 dB; secondary step size 5 dB
Order.Primary: 1
StepSize.Primary: 10
StepSize.Secondary: 5

%Catch trials are included at specified length and frequency of occurrence
CatchTrial.Enabled: 1
CatchTrial.Length: 2000
CatchTrial.Frequency: 10
StandardError: 1.6

%Level of performance for adaptive procedure = 50%
Method: 1

% Stimuli.Primary: pulse; freq; dur of pulse; interval; rise; fall; ear

Stimuli.Primary: pulse; 250 ; 200; 50; 20; 20; right
Stimuli.Primary: pulse; 500 ; 200; 50; 20; 20; right
Stimuli.Primary: pulse; 1000 ; 200; 50; 20; 20; right
Stimuli.Primary: pulse; 2000 ; 200; 50; 20; 20; right
Stimuli.Primary: pulse; 4000 ; 200; 50; 20; 20; right
Stimuli.Primary: pulse; 8000 ; 200; 50; 20; 20; right
Stimuli.Primary: pulse; 10000 ; 200; 50; 20; 20; right

Stimuli.Primary: pulse; 250 ; 200; 50; 20; 20; left
Stimuli.Primary: pulse; 500 ; 200; 50; 20; 20; left
Stimuli.Primary: pulse; 1000 ; 200; 50; 20; 20; left
Stimuli.Primary: pulse; 2000 ; 200; 50; 20; 20; left
Stimuli.Primary: pulse; 4000 ; 200; 50; 20; 20; left
Stimuli.Primary: pulse; 8000 ; 200; 50; 20; 20; left
Stimuli.Primary: pulse; 10000 ; 200; 50; 20; 20; left

Timer: 3000
ISI.Length: 2000

% Button: XSIZE; YSIZE; XLOC; YLOC; FontSize; Response Category; Text;

Button: 150 ; 150 ; 50 ; 450 ; 16 ; 1 ; BEEP BEEP BEEP ;

feedbackfile: c_fish_10
feedbackfile: c_girl_10
feedbackfile: c_play_14
feedbackfile: c_sand_11
feedbackfile: c_train_12


Feedback.Directory: X:\Res_Progs\HAL\gamepics\
```

% **EXAMPLE OF EXPERIMENT DEFINITION FILE**

```
Order.Primary: 1
Order.Secondary: Block
Stimuli.SecondaryContinuous: 0
MaxLevel: 120
NumberOfTrials: 120
GameDelay: 1
Timer: 15000
ISI.Length: 500
ISI.Jitter: 0
WaitForNextButton: 0
Interval.Cue: 0
CatchTrial.Enabled: 0
CatchTrial.BackToBack: 0
CatchTrial.Length: 500
CatchTrial.Frequency: 5
Range.StepSize: 0
Range.Width: 0
Repetitions: 10
Interval.Buttons: 0
Interval.TimeOn: 2000
Interval.Interval: 250
Method: 2
DirOfChange: -
StepSize.Primary: 10
StepSize.Secondary: 5
Stop.Mode: 1
Stop.StandardError: 1.8
Stop.PrimaryReversals: 5
Stop.SecondaryReversals: 6
```

% Stimuli.Primary: wave ; *filename* ; *category ; INFO* ; *level* ; *ear ; amplitude method*

```
Stimuli.Primary: wave;r:\harl\common\bw\bw novel\stim\foss.wav;1;foss;60;both;none
Stimuli.Primary: wave;r:\harl\common\bw\bw novel\stim\wul.wav;2;wul;60;both;none
Stimuli.Primary: wave;r:\harl\common\bw\bw novel\stim\hain.wav;3;hain;60;both;none
Stimuli.Primary: wave;r:\harl\common\bw\bw novel\stim\kathe.wav;4;kathe;60;both;none
Stimuli.Primary: wave;r:\harl\common\bw\bw novel\stim\mide.wav;5;mide;60;both;none
Stimuli.Primary: wave;r:\harl\common\bw\bw novel\stim\teap.wav;6;teap;60;both;none
Stimuli.Primary: wave;r:\harl\common\bw\bw novel\stim\riv.wav;7;riv;60;both;none
Stimuli.Primary: wave;r:\harl\common\bw\bw novel\stim\zeb.wav;8;zeb;60;both;none
```

% Button: XSIZE ; YSIZE ; XLOC ; YLOC ;FontSize; Response Category ; TEXT ; Picture file (full path)

```
Button:120;180;1;1;1;1;;"R:\HARL\Common\BW\BW Novel\foss.bmp"
Button:120;180;121;1;1;2;;"R:\HARL\Common\BW\BW Novel\wul.bmp"
Button:120;180;1;181;1;3;;"R:\HARL\Common\BW\BW Novel\hain.bmp"
Button:120;180;121;181;1;4;;"R:\HARL\Common\BW\BW Novel\kathe.bmp"
Button:120;180;1;361;1;5;;"R:\HARL\Common\BW\BW Novel\mide.bmp"
Button:120;180;121;361;1;6;;"R:\HARL\Common\BW\BW Novel\teap.bmp"
Button:120;180;1;541;1;7;;"R:\HARL\Common\BW\BW Novel\riv.bmp"
Button:120;180;121;541;1;8;;"R:\HARL\Common\BW\BW Novel\zeb.bmp"

FeedbackFile: x:\res_progs\hal\gamepics\ss_sport_22
FeedbackFile: x:\res_progs\hal\gamepics\ss_animals_32
FeedbackFile: x:\res_progs\hal\gamepics\ss_newgames_25
FeedbackFile: x:\res_progs\hal\gamepics\ss_camera_25
FeedbackFile: x:\res_progs\hal\gamepics\pzl_wateringhole
FeedbackFile: x:\res_progs\hal\gamepics\pzl_clown

Results: perf;primarystim;
```

% Example of Definition file for Paired Comparison

```
%%%HAL definition file
%%%Generated 2/3/2010

Order.Primary: 0
Order.Secondary: Block
Stimuli.SecondaryContinuous: 0
mix.secondary: do not mix
MaxLevel: 120
NumberOfTrials: -1
GameDelay: 1
GameDelayType: 1
Timer: 60000
ISI.Length: 1000
ISI.Jitter: 0
WaitForNextButton: 0
Interval.Cue: 0
CatchTrial.Enabled: 1
CatchTrial.BackToBack: 0
CatchTrial.Length: 500
CatchTrial.Frequency: 40
Range.StepSize: 0
Range.Width: 0
Repetitions: 1
Interval.Buttons: 1
Interval.TimeOn: 1000
Interval.Interval: 1000
ChangeFeedback: 2
Adapt: None
Method: 1
DirOfChange: -
StepSize.Primary: 10
StepSize.Secondary: 5
Stop.Mode: 0
Stop.StandardError: 1.8
Stop.PrimaryReversals: 5
Stop.SecondaryReversals: 6
Stimuli.Primary: wave;all-fc.wav;1;adult;60;both;none
Stimuli.Paired:wave;X:\Res_Progs\BART\Bart3_0_0\sound files\;speech;1;75;both;rms

Button:150;120;30;530;20;1;Text for A;""
Button:150;120;230;530;20;2;Text for B;""
Button:150;120;500;530;20;repeat;Repeat;""
Button:150;120;700;530;20;3;No Different;""

FeedbackFile: x:\res_progs\hal\gamepics\ss_sillyanimals.txt
Results: freq;responsecat;primarystim;secondarystim;

FreqShapeSecondary: 0
CLAnchor: 0
```

EXAMPLE RESULTS FILE

%**EXPERIMENT RESULTS FILE**

```
%SUBJECT DATA
%NAME
%DOB
%Female
%GROUP
%ID

%R:\HARL\Common\HAL\HIA_DR\IAM_left.def
%7/13/2004
%12:17 PM
%49;10

%STIMULI
%2000hz.wav
%3000hz.wav
%1500hz.wav
%4000hz.wav
%250hz.wav
%500hz.wav
%750hz.wav
%1000hz.wav
%6000hz.wav
%8000hz.wav

%REPETITIONS
%1
%ADAPTIVE
%Level

%RAW DATA
% Primary
%Stim,PresLevel,SecStim,CornerFreq,Responsecat,Response,Correct,Action,Method,
Ear
2000hz.wav,70,1000.wav,0,1,9,0,-,1,2
2000hz.wav,80,1000.wav,0,1,9,0,-,1,2
2000hz.wav,90,1000.wav,0,1,9,0,-,1,2
2000hz.wav,100,1000.wav,0,1,1,1,+,1,2
2000hz.wav,90,1000.wav,0,1,9,0,-,1,2
2000hz.wav,100,1000.wav,0,1,1,1,+,1,2
2000hz.wav,90,1000.wav,0,1,9,0,-,1,2
2000hz.wav,100,1000.wav,0,1,1,1,+,1,2
2000hz.wav,95,1000.wav,0,1,1,1,+,1,2
2000hz.wav,90,1000.wav,0,1,9,0,-,1,2
2000hz.wav,95,1000.wav,0,1,9,0,-,1,2
2000hz.wav,100,1000.wav,0,1,1,1,+,1,2
2000hz.wav,95,1000.wav,0,1,9,0,-,1,2
2000hz.wav,100,1000.wav,0,1,1,1,+,1,2
2000hz.wav,95,1000.wav,0,1,9,0,-,1,2
2000hz.wav,100,1000.wav,0,1,1,1,C,1,2
3000hz.wav,70,1000.wav,0,1,9,0,-,1,2
3000hz.wav,80,1000.wav,0,1,9,0,-,1,2
3000hz.wav,90,1000.wav,0,1,9,0,-,1,2
3000hz.wav,100,1000.wav,0,1,9,0,-,1,2
3000hz.wav,107,1000.wav,0,1,9,0,-,1,2
3000hz.wav,107,1000.wav,0,1,9,0,-,1,2
3000hz.wav,107,1000.wav,0,1,9,0,-,1,2
3000hz.wav,107,1000.wav,0,1,9,0,C,1,2
1500hz.wav,40,1000.wav,0,1,9,0,-,1,2
1500hz.wav,50,1000.wav,0,1,9,0,-,1,2
1500hz.wav,60,1000.wav,0,1,9,0,-,1,2
```

```
1500hz.wav,70,1000.wav,0,1,9,0,-,1,2
1500hz.wav,80,1000.wav,0,1,9,0,-,1,2
1500hz.wav,90,1000.wav,0,1,9,0,-,1,2
1500hz.wav,100,1000.wav,0,1,1,1,+,1,2
1500hz.wav,90,1000.wav,0,1,9,0,-,1,2
1500hz.wav,100,1000.wav,0,1,1,1,+,1,2
1500hz.wav,90,1000.wav,0,1,9,0,-,1,2
1500hz.wav,100,1000.wav,0,1,1,1,+,1,2
1500hz.wav,95,1000.wav,0,1,1,1,+,1,2
1500hz.wav,90,1000.wav,0,1,9,0,-,1,2
1500hz.wav,95,1000.wav,0,1,1,1,+,1,2
1500hz.wav,90,1000.wav,0,1,1,1,+,1,2
1500hz.wav,85,1000.wav,0,1,9,0,-,1,2
1500hz.wav,90,1000.wav,0,1,9,0,-,1,2
1500hz.wav,95,1000.wav,0,1,1,1,C,1,2
4000hz.wav,70,1000.wav,0,1,9,0,-,1,2
4000hz.wav,80,1000.wav,0,1,9,0,-,1,2
4000hz.wav,90,1000.wav,0,1,9,0,S,1,2
250hz.wav,40,1000.wav,0,1,9,0,-,1,2
250hz.wav,50,1000.wav,0,1,9,0,S,1,2
500hz.wav,40,1000.wav,0,1,9,0,-,1,2
500hz.wav,50,1000.wav,0,1,9,0,S,1,2
750hz.wav,40,1000.wav,0,1,9,0,-,1,2
750hz.wav,50,1000.wav,0,1,9,0,S,1,2
1000hz.wav,40,1000.wav,0,1,9,0,-,1,2
1000hz.wav,50,1000.wav,0,1,9,0,-,1,2
1000hz.wav,60,1000.wav,0,1,9,0,-,1,2
1000hz.wav,70,1000.wav,0,1,9,0,-,1,2
1000hz.wav,80,1000.wav,0,1,1,1,+,1,2
1000hz.wav,70,1000.wav,0,1,1,1,+,1,2
1000hz.wav,60,1000.wav,0,1,9,0,-,1,2
1000hz.wav,70,1000.wav,0,1,1,1,+,1,2
1000hz.wav,60,1000.wav,0,1,9,0,-,1,2
1000hz.wav,70,1000.wav,0,1,1,1,+,1,2
1000hz.wav,65,1000.wav,0,1,9,0,-,1,2
1000hz.wav,70,1000.wav,0,1,1,1,+,1,2
1000hz.wav,65,1000.wav,0,1,9,0,-,1,2
1000hz.wav,70,1000.wav,0,1,1,1,+,1,2
1000hz.wav,65,1000.wav,0,1,9,0,-,1,2
1000hz.wav,70,1000.wav,0,1,1,1,C,1,2


%,2000hz.wav,98
%,3000hz.wav,117
%,1500hz.wav,90
%,4000hz.wav,75
%,250hz.wav,40
%,500hz.wav,40
%,750hz.wav,40
%,1000hz.wav,68

%Elapsed Time: 0:5:43
```

APPENDIX D

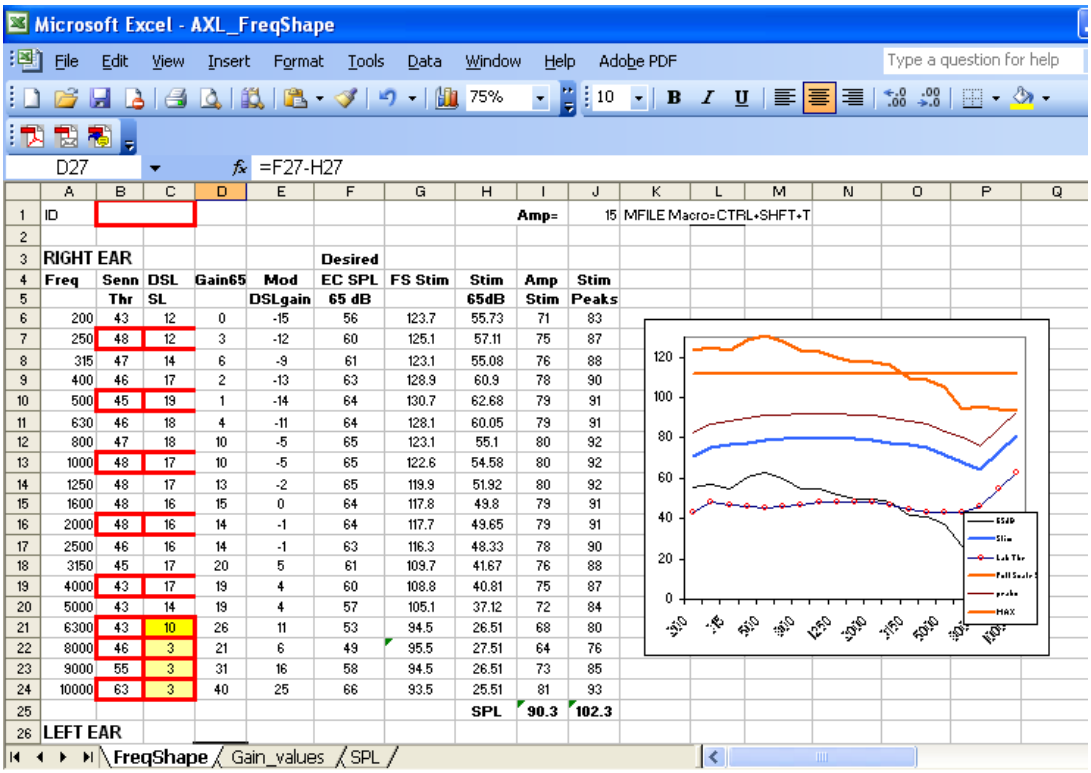
Frequency Shaping (FreqShape)

The FreqShape function is a routine that applies frequency shaping to any input signal to compensate for degree and configuration of hearing loss on an individual basis. As described below, frequency shaping is based on the Desired Sensation Level procedure (DSL i/o) using the long-term average spectrum of speech at any level specified. The example below assumes an average conversation (~65 dB SPL) as the input level.

Required Files:

Template_FreqShape.txt
FreqShape.xls
DSLmio.?

An excel spreadsheet (R:\HARL\common\calibration\freqshape\FreqShape.xls) is used to calculate the necessary attenuate values to shape incoming signals.

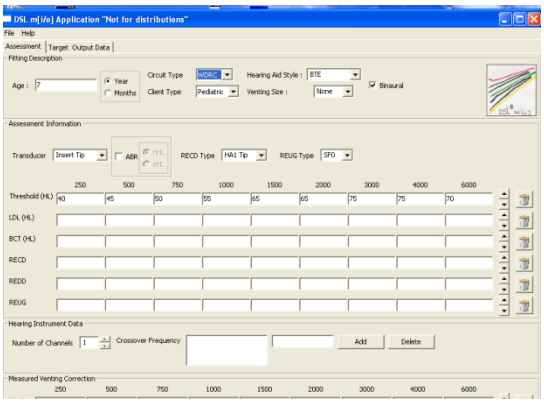


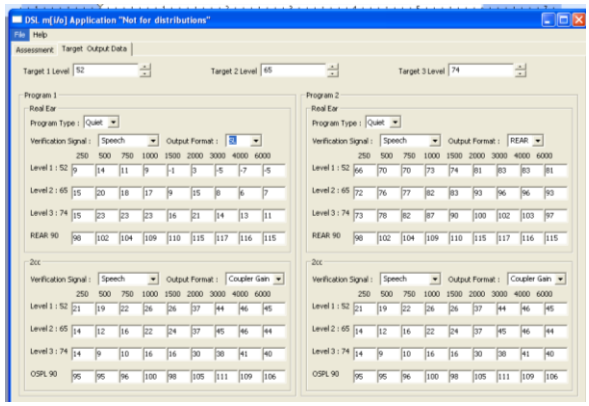
	A	B	C	D	E	F	G	H	I	J	K	L	M
1	ID								Amp=	15	MFILE Macro=CTRL+SHFT+T		
2													
3	RIGHT EAR					Desired							
4	Freq	Senn	DSL	Gain65	Mod	EC SPL	FS Stim	Stim	Amp	Stim			
5		Thr	SL		DSLgain	65 dB		65dB	Stim	Peaks			
6	200	43	12	0	-15	56	123.7	55.73	71	83			
7	250	48	12	3	-12	60	125.1	57.11	75	87			
8	315	47	14	6	-9	61	123.1	55.08	76	88			
9	400	46	17	2	-13	63	128.9	60.9	78	90			
10	500	45	19	1	-14	64	130.7	62.68	79	91			
11	630	46	18	4	-11	64	128.1	60.05	79	91			
12	800	47	18	10	-5	65	123.1	55.1	80	92			
13	1000	48	17	10	-5	65	122.6	54.58	80	92			
14	1250	48	17	13	-2	65	119.9	51.92	80	92			
15	1600	48	16	15	0	64	117.8	49.8	79	91			
16	2000	48	16	14	-1	64	117.7	49.65	79	91			
17	2500	46	16	14	-1	63	116.3	48.33	78	90			
18	3150	45	17	20	5	61	109.7	41.67	76	88			
19	4000	43	17	19	4	60	108.8	40.81	75	87			
20	5000	43	14	19	4	57	105.1	37.12	72	84			
21	6300	43	10	26	11	53	94.5	26.51	68	80			
22	8000	46	3	21	6	49	95.5	27.51	64	76			
23	9000	55	3	31	16	58	94.5	26.51	73	85			
24	10000	63	3	40	25	66	93.5	25.51	81	93			
25								SPL	90.3	102.3			
26	LEFT EAR												

Inputs to the spreadsheet include:

1. *Audiometric thresholds (in real-ear SPL)* obtained with the experimental transducer *using a standard calibration.* (To express all values in real-ear SPL, it is necessary to use this function - see Chirp Calibration documentation). These thresholds should be entered into *Column B.*

2. Open *DSLmio* from the desktop. Select *Real Ear SPL* from the pull-down menu under Transducer. Enter the subjects age (years) and the Lab Thresholds. Click on the *Target Output Data* tab and change the *Target2* level to 65. Change the *Output Format* to SL using the pull-down menu. Copy the *Level2* values to the Subject's datasheet. Repeat procedure for the other ear. For experiments with multiple or alternative input levels, it will be necessary to generate different target values for each level and adjust the levels in Column I to reflect a higher or lower level.

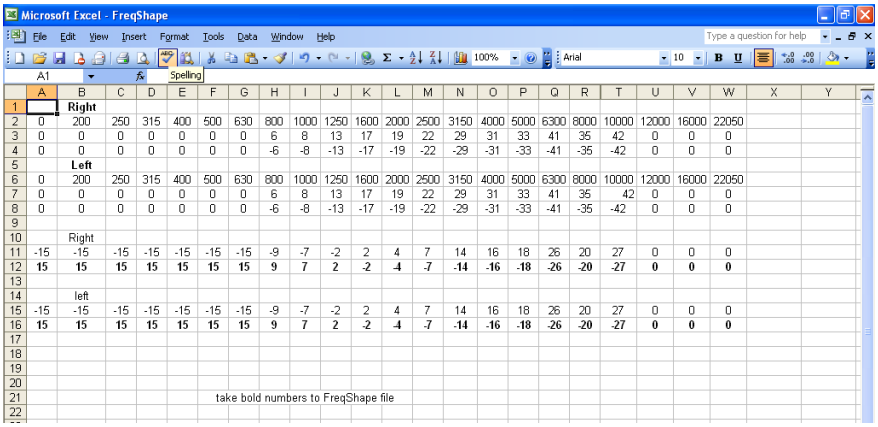




3. The graph on the right shows: a) thresholds, b) average unamplified LTASS (stim) c) full-scale LTASS, and d) amplified LTASS. Note that when there are regions of near normal hearing, the program will not allow frequency-shaped signal to fall below unamplified levels.

4. If the graph shows that audibility is insufficient in any frequency region, adjust cell H1. When the desired amplifier is determined add amplifier into the circuit and adjust the gain accordingly.

5. Run the macro (Ctrl+Shift+T) to generate gain values and copy them to the Gain Value sheet.



Microsoft Excel - FreqShape

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	T	U	V	W	X	Y
1		Right																						
2	0	200	250	315	400	500	630	800	1000	1250	1600	2000	2500	3150	4000	5000	6300	8000	10000	12000	16000	22050		
3	0	0	0	0	0	0	0	6	8	13	17	19	22	29	31	33	41	35	42	0	0	0		
4	0	0	0	0	0	0	0	-6	-8	-13	-17	-19	-22	-29	-31	-33	-41	-35	-42	0	0	0		
5		Left																						
6	0	200	250	315	400	500	630	800	1000	1250	1600	2000	2500	3150	4000	5000	6300	8000	10000	12000	16000	22050		
7	0	0	0	0	0	0	0	6	8	13	17	19	22	29	31	33	41	35	42	0	0	0		
8	0	0	0	0	0	0	0	-6	-8	-13	-17	-19	-22	-29	-31	-33	-41	-35	-42	0	0	0		
9																								
10		Right																						
11	-15	-15	-15	-15	-15	-15	-15	-9	-7	-2	2	4	7	14	16	18	26	20	27	0	0	0		
12	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**9**	**7**	**2**	**-2**	**-4**	**-7**	**-14**	**-16**	**-18**	**-26**	**-20**	**-27**	**0**	**0**	**0**		
13																								
14		left																						
15	-15	-15	-15	-15	-15	-15	-15	-9	-7	-2	2	4	7	14	16	18	26	20	27	0	0	0		
16	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**9**	**7**	**2**	**-2**	**-4**	**-7**	**-14**	**-16**	**-18**	**-26**	**-20**	**-27**	**0**	**0**	**0**		
17																								
18																								
19																								
20																								
21																								
22																								

take bold numbers to FreqShape file

6. To provide automatic frequency shaping from within BART, rows 12 & 16 from the Gain Values sheet should be copied into the [ ] in Template_Freqshape.txt.

7. To process files *off line* (as in a powerpoint show), use the MassFilter function.

NOTE: The following is additional documentation for FreqShape

Column:

A. Frequency

B. Threshold in dB SPL

C. Desired sensation level (Based on DSL, but other algorithms can be used)

D. Gain required to meet desired SPL (column F)

E. Gain + amplifier setting (cell J1)

F. Threshold + Desired SL (B + C)

G. Full Scale Stimulus – Assumes the Cox & Moore LTASS. Is based on max output of card + amplifier value (if used).

H. Cox * Moore LTASS at 65 dB overall level

I. Stimulus input + gain + amplifier value (H + D+ J1)

J. Peaks of speech (I + 12 dB)