**To build a new experiment sequence:**

1. Open SetupExperiment.m for editing
   1. Change nBands to the number of spectral bands that will be used in this experiment.
   2. Change bandsPerTrial to the number of bands that should be present in each trial of the experiment. I’ve found that a ratio of bandsPerTrial/nBands of roughly 0.4 gives about 50% performance in normal hearing subjects, and is generally above floor in cochlear implant subjects, although CI subjects tend to be much less predictable.
   3. Change nTrials to the total number of trials desired for this experiment. My designs have usually aimed for nTrials\*bandsPerTrial/nBands of around 70-75, although I recommend consulting the supplemental material in Bosen and Chatterjee (2016) to determine the number of trials that will give you the desired power.
   4. Change subjectID to the subject parameter file for this subject (example: ‘NH\_SII\_Bands’)
   5. Set preFilterStimuli to 1 if prefiltering is desired, or 0 if not. I highly recommend prefiltering, since it can take 6 or 7 seconds on our lab PC to perform filtering on the fly. If prefiltering is selected, files will be saved to ‘.\Filtered Sound Files\’ and must be present when running an experiment.
   6. Set vocoderType to ‘none’, ‘rectangular’ or ‘monopolar’, to select a synthesis filter. Details on each synthesis filter can be found in Bosen and Chatterjee (2016).
   7. Set trialsPerBlock to the number of trials per block in this experiment. I find that subjects are usually okay with block sizes of around 60 – 80 trials.
   8. Set experimentSentences and baselineSentences to index the desired IEEE sentence numbers. In my setup, pulled baseline sentences from the end of the list (sentences 681:720), but really as long as they don’t overlap and are consistent across subjects any selection should suffice.
   9. Change nRuns to the desired number of iterations of the trial matrix generation algorithm. More runs take longer, but give a better chance of finding trial sets that have roughly uniform probabilities of pairs of bands occurring together.
2. Run SetupExperiment.m in Matlab. This will take a while (roughly an hour with the default settings on our lab’s computer). If faster performance is needed, the function ChooseTargetTrials.m contains commented out lines that parallelize its main loop (look for poolobj and parfor), although I make no promises that this will work on your machine. Once the trials are selected a surface plot showing the frequency of bands co-occurring will be plotted, and shortly after the spectrogram of each filtered sentence will be showed as it is generated (if pre filtering). Experiment and baseline sequences will be saved to ‘.\Experiment Sequences\’

**To run an experiment sequence:**

1. Experimenter performs these steps in advance:
   1. Run RunExperiment.m in Matlab.
   2. Select Subject Parameters File (example: ‘.\Subject Parameters\NH\_SII\_Bands.dat’)
      1. Note: The subject parameters file needs to be set, but is only used for trials in the experiment sequence that have not been prefiltered (i.e. ‘prefiltered’ column in the experiment sequence file is set to zero). For audio files that have been prefiltered, the experiment software will simply play the file as is, without any processing at runtime.
   3. Select Experiment Sequence (example: ‘.\Experiment Sequences\NH\_SII\_Bands\_vocoder-rectangular\_Baseline.dat’).
   4. Press the ‘Start Experiment’ Button.
2. Subject performs these steps during the experiment:
   1. Press enter on the keyboard or the ‘Play Sound’ button to hear the sentence. It will play once, and cannot be repeated.
   2. Type whatever you thought was said in the ‘What did you hear?’ Text box.
   3. When you have finished entering your response, either press enter on the keyboard or the ‘OK’ button.
   4. Repeat until all trials have been completed
3. Experimenter performs these steps after the experiment has been completed:
   1. Subject responses are saved in the ‘.\Experiment Results\’ folder, with the same name as the experiment sequence file.
   2. Make a new folder for this subject (example: ‘.\Experiment Results\NHA87’), and move all subject response files to that folder.

**To score experiment results:**

1. Open ProcessExperimentResultsForScoring.m for editing
   1. Set subjectID, vocoderType, subjectParameters, and experimentBlockNames
   2. Choose whether you want to score results based solely on the five keywords in each sentence (scoreKeywordsOnly = 1) or based on every word in each sentence (scoreKeywordsOnly = 0). Convention in the field is generally to score based on keywords only.
2. Run ProcessExperimentResultsForScoring.m in Matlab
3. Open the processed results file (located in ‘.\Processed Results\[SUBJECT\_ID]\’) in Excel.
   1. Format the column widths and font size to make it easy to read
   2. For each trial in the processed results, the subject’s response is listed first, the correct response is listed second. Count the number of words that the subject got correct, enter that number in the ‘Words Correct’ column, in the first row corresponding to that trial (next to the number in the ‘Total Words’ column).
   3. Save the processed results as a .xlsx file with the same name.
   4. Repeat for each processed results file.

**To obtain band importance function from scored results:**

1. Open AnalyzeProcessedResults.R for editing
   1. Change parameterFileName to the subject parameter file used to run the experiment (example: ‘.\Subject Parameters\NH\_SII\_Bands.dat’)
   2. Change subjectID to match the folder name where the subject’s results are contained (example: ‘NHA87’).
   3. Change blocks to match the scored result file names, without the subject name prepended (example: c(“Baseline”, “Block 1”, “Block 2”) ).
   4. Change nBands (TODO) to the number of bands used in this experiment sequence (for NH listners this is set to 20, for cochlear implant subjects this depends on the device and the experiment sequence that was generated for them).
2. Run AnalyzeProcessedResults.R in R. Figures will be generated for this subject and saved in ‘.\Figures\’, and the R command line will display regression coefficients, significance, and some statistics related to subject performance.

**To build and use calibration signal:**

1. Run GenerateCalibrationSignal.m in Matlab.
2. Play ‘Calibration\_Signal.wav’ in your audio booth, adjust speaker volume to desired level. We used 65dB, C-weighting, fast settings.

This software tests how different spectral bands contribute to speech intelligibility in normal hearing and cochlear implant subjects. This is done using the method described in Bosen and Chatterjee (2016), in which speech materials are filtered to contain a pseudorandom subset of spectral bands. The software is currently tailored to use IEEE sentences, but it could be adapted for other stimuli as well. By regressing speech intelligibility (number of words correct in target stimuli) against the presence or absence of a given band across the experiment, we can determine that band’s average contribution to speech intelligibility.

The main program that runs this experiment is the script ‘RunExperiment.m’. It was created in Matlab’s GUIDE interface, then exported to be a standalone script. This script acts as a wrapper to load the subject parameter and experiment sequence files, and calls ‘ExecuteTest.m’ to run the actual experiment. In the experiment, subjects press the ‘Play Sound’ button or press enter to play a target sound (determined by the experiment sequence), then type in what they thought the heard and press the ‘OK’ button or press enter to submit their response.

Each subject needs to have a .dat file in ‘./Subject Parameters’ that defines their available channels and the acoustic range for each channel, which is selected when generating an experiment sequence and at the start of an experiment. These files contain three columns: Channel Number, Lower Bound, and Upper Bound (columns are tab delimited). This file is specified in the ‘Subject Parameters’ field in the main GUI, but is only used if the stimuli for the experiment aren’t prefiltered. I think I’ve set this up so channel order and ranges can be arbitrary, although I’d recommend testing it before doing anything too peculiar.

Each experiment block is defined by a .dat file in ‘.\Experiment Sequences’ that defines the file to be played for each trial, the acoustic bands to be included in the stimulus (if filtering during the experiment), and whether or not the sound file has been pre-filtered. Experiment sequences can be written manually, or can be generated by modifying and calling ‘SetupExperiment.m’. If prefilterStimuli is true, the sequence generator will create filtered sound files and save them in ‘.\Filtered Sound Files’, and save the corresponding file names in the experiment sequence. Additionally, stimuli can be vocoded (for use with NH listners) by setting the vocoder type to ‘rectangular’ or ‘monopolar’. Currently, there’s no option to vocode stimuli during an experiment, since it’s relatively time consuming.

Each trial in an experiment block is saved to a corresponding file in the ‘.\Experiment Results’ folder. Each row of this file contains the trial number, the sound file that was played, the number of active channels (which was only actually used if the sound files were not prefiltered), and the response the subject typed for that trial. These provide a raw record of subject performance, but are tedious to score. The script ‘ProcessExperimentResultsForScoring.m’ takes the raw data files and processes them a bit to be easier to score. The results file is defined in the resultsFile variable and the output file is defined by the fopen command for the outputFileHandle in this script, so both variables should be modified before running the script. This script takes the raw response for each trial, finds the corresponding correct response in the IEEE sentence list (located at ‘.\Sound Files\IEEE3\_FILE\_NAMES.TXT), and prints both to the output file together for easy comparison. To score results, open the processed result file and count the number of correct words in each subject response (subject’s response is the first line for each trial, correct response is second line), then put that number in the column to the left of the total number of words in the sentence.

If you need to rebuild the calibration signal, the script that generates it is ‘GenerateCalibrationSignal.m’. This script takes AzBio Sentence List 1, concatenates all 20 sound files, randomizes the phase of the resultant audio sequence, then writes the result to ‘Calibration\_Signal.wav’. This produces speech-shaped noise for calibration. Currently, we’ve been setting the gain so that the level is 65dB (C-weighting, fast settings) at the subject’s ear.

If you want to manually filter sound files, the function ‘BandFilterAuditorySignal.m’ is what I run all the sound files through. Hopefully, the comments in the file should be sufficient to figure out how to use the function.

**References**

Bosen AK, Chatterjee M (2016). Band importance functions of listeners with cochlear implants using clinical maps. *The Journal of the Acoustical Society of America*, 140(5)