FACULTY OF ENGINEERING AND ARCHITECTURE

DEPARTMENT OF INFORMATION TECHNOLOGY
RESEARCH GROUP WAVES

Prof. Dr. Verhulst
s.verhulst@ugent.be
waves.intec.ugent.be/hearing-technology

Lab Exercise 1: Auditory Model Flavours, Properties and Use

Make a new conda-environment that operates Python 3.6:
```
conda create -n name_environment python=3.6 anaconda
```

Choose your own environment name and adapt name_environment for convenience.
Activate this environment:
```
source activate name_environment
```

Next, install the needed packages for these exercices:
```
conda install -c conda-forge keras
pip install sounddevice --user
pip install soundfile --user
```

Go to the AudCompLab1 folder in your terminal (`cd..`) and open a jupyter notebook.
```
jupyter notebook
```

If you are a Matlab user getting into python, you can use the following cheatsheets:
https://cheatsheets.quantecon.org

You can get more python basics from the following book (available via UGent library):
A Primer on Scientific Programming with Python (5th ed., Springer)

Part 1: Features for Speech processing

Automatic Speech Recognition Systems or Machine Learning often uses energetic features of in short-time windows (i.e. Magnitude of the short-time Fourier Transform) to perform their tasks rather than applying a real-time calculation based on the auditory band-pass filtered time-waveform. The `CompAudLab1_Speech` file investigates several approaches (Short-Time Fourier Transform, Mel, MFCC and Gammatone filterbank) which could yield speech features for machine-listening applications. Run the first and second cell of the code with the given parameters to compare the different speech images.

1.1 Make a new code-cell in which you create a figure which compares the center frequencies of the computed bands in the mel/mfcc and gammatone features. If you're able to run the CoNNear model, a neural network representation of a non-linear transmission-line model, you should add this to your comparison as well. If not, comment the parts from the CoNNear model out (you should however be able to run this with the correct packages installed). You might require to write out the following parameters to generate your figure: `mel_out['cf']`, `gt_out['cf']`. The CF's for the CoNNear model can be found in the cf.txt file.

What are the differences between the mel and gammatone frequency-per-band organization? Are both scales logarithmically spaced?

1.2 Machine-hearing applications (e.g. speech enhancement) often use an energetic representation of the signal in fixed filter-bands and time frames (and often with overlapping time windows). The benefit of this method is that it reduces the speech features considerably, which might be beneficial when using them as inputs to machine-learning algorithms, and when using gammatone or STFT features, their inverse transforms can be implemented to reobtain a modified speech-signal in the time domain. However, the approach is very dependent on the frame duration, and removes the original signals' phase information. In the next code cell, you will evaluate the influence of this energetic procedure onto the signal waveform in a specific frequency band. Run the written code-cell, compare the plots and listen to the sound outputs. Now vary the `framelength_t` and `frameshift_t` parameters in a proportional way in the top cell of the code, and evaluate their influence on the energetic algorithm and sound. Describe your observations.

```
framelength_t = 0.025 # framelength in time = 25ms
frameshift_t = 0.01 # frameshift in time = 10ms
```

Part 2: Auditory Model Properties

Here you will compare the outputs of a linear gammatone (GT) filterbank and a neural network replacement (CoNNear) of a coupled nonlinear transmission-line (TL) model of the cochlea. Both models mimic how the human cochlea processes sound and compute filtered outputs over a range of frequency bands which span the human audible frequency range. In the GT filter bank, the frequency spacing follows the perceptual equivalent-rectangular bandwidth (ERB) scale whereas the CoNNear model picks a large enough number of channels necessary to yield a stable numerical solution of the coupled oscillator equations (here 1000 are computed, and 201 are stored, see Altoè et al., 2014). To explore the properties of the two types of models we can explore their responses to two types of common stimuli: the pure tone and a click signal. The pure-tone can be used to estimate the *cochlear excitation pattern* as a function of frequency and level, whereas the click signal is ideal to assess the impulse response of the model and associated *filter tuning*.

2.1 Open the `CompAudLab_Tuning` file and insert your code to generate a 1-kHz pure tone signal of duration 250 ms. You will require the following functions for this: `np.arange`, `np.sin`. You can plot your generate pure tone using the example code in the next cell. The sampling frequency of the CoNNear model is 20 kHz and that of the GT model is 48 kHz. Take note of the comments about the addition of context for the CoNNear model.

2.2 Pure-tones of high-frequencies have a very sharply rising instantaneous amplitude, and for that reason, they are often windowed with a certain onset/offset ramp before their use in psychoacoustic experiments. Sharply rising instantaneous amplitudes often create an "onset" response of the auditory system which is heard as a click at the start of stimulus. Generate a custom window with the same length of your pure tone, which has amplitude 1, except for the first and lest 10-ms which should follow the rising and decaying slope of the waveform from a Hanning window, respectively. You will need the following functions to generate this window: `np.hanning, np.ones, np.concatenate`

Plot the time-domain waveform of the window as well as of the windowed pure-tone stimulus (i.e., point-by-point multiplication of your window with the pure-tone stimulus). Listen to your pure-tone before and after applying a hanning window on it using the following command: `sd.play(signal, fs)`

2.3 To assess the properties of a cochlear filters' output, it is worthwhile to evaluate the response during the stimulus as well as for some time after the stimulus offset. Zero-pad your windowed pure tone with 50 ms of silence. Use the functions `np.zeros` and `np.concatenate` for this purpose and plot your signal.

2.4 Execute the code in the next two cells, which runs the stimuli through the GT and CoNNear model, and which generates output for the same number of time samples as were present in the stimulus over the number of defined frequency channels. The second cell plots a cochleagram (i.e. energies in short timeframes (bins) vs simulated frequency bands/auditory filters) as well as the filtered time-domain signal from the 1-kHz auditory filter in the model. Do you notice a difference between the two models?

2.5 Next, create an excitation pattern for the 70-dB 1-kHz pure tone, which is a figure which plots the root-mean-square energy of each time-domain signal for every simulated cochlear frequency band. Calculate excitation patterns for the GT and CoNNear model and you may need to following functions for this operation: `np.sqrt`, `np.mean`. Plot both excitation patterns on the same plot vs the center frequency of the simulated cochlear filter band. You can get you can extract the variables as follows: `gt_out['cf']` for the GT model, and to get the CFs of the CoNNear model, use the cf.txt file. Plot the results in a normalized way, i.e. divide the rms values by the maximum rms of each model so that the maxima of both excitation patterns are the same and = 1. Which differences do you observe between the two excitation patterns, and what do you think the origin of the differences are?

2.6 In the next step, we will investigate how the cochlear filters change as a function of frequency and level in both models. Filters can be characterized by their impulse response, hence we can stimulate the model with a short transient click signal and investigate how the impulse responses change. Make a transient stimulus which has 1 ms of zeros, 100 µs of ones and 50 ms of zeros. Use the `np.zeros, np.ones and np.concatenate` functions for this purpose and plot the waveforms. Note that the GT and CoNNear model have different sampling frequencies. Execute the next code cell to run a loop that simulates the models for 10 different sound levels. For each sound level, it will calculate the time-domain output for the different simulated channels as well as the magnitude spectrum. Make a figure which plots the time-domain waveforms as well as the magnitude spectra for the different simulated levels and for a the 1 kHz channel. Do the filters vary as a function of level? Compare between the GT and CoNNear model implementations.

Part 3: Sound Localization Models

The Jeffres type of binaural models use a cross-correlation technique within different auditory filter bands to compute the delay between the ears, from which the interaural time-delay (ITD) can be derived. In this last exercise, you will use a binaural cross-correlation model to determine the laterization (location inside the head) of a virtual sound source. First you will

convolve a monaural sound stimulus to achieve with a specific HRTF (head-related transfer function), after which you can use a binaural model to calculate the ITD across the different frequency channels. At the end of the exercise, you are also asked to compute the interaural level difference (ILD) across the different frequency bands in the L and R gammatone filterbank. Does the ILD provide you similar information to the ITD and is it consistent in saying where the sound source comes from? Are there robust and consistent ITD and ILD cues across all frequencies? The `CompAudLab_Binaural` file has step-by-step instructions to complete this exercise.