# ALaRI: Digital Signal Processing

## Exercise Session 1: Audio playback, recording, and analysis

Alexander Bertrand, Deepaknath Tandur, and Toon van Waterschoot[1]

May 2013

**Goal:** to create m-files and a Simulink model for playing back and recording audio signals, and performing a time-frequency analysis of these signals.

**Requirements:** Matlab/Simulink in Windows, a sound card with line in/out, a loudspeaker and a microphone (possibly in a headset).

**Outcome:** 2 m-files (initparams.m, playrec.m), and 1 Simulink model (audio_io.mdl).

**Important remark:** This exercise requires the use of Matlab/Simulink in Windows. You can either work on your personal laptop, if it has Matlab/Simulink installed under Windows, or you can run Matlab/Simulink in a Windows Virtual Machine on one of the Linux PC's in the ALaRI computer lab. To create the Windows Virtual Machine you should execute the following steps: (1) run `virtualbox` from the Linux command line, (2) create a new virtual machine (with an arbitrary name, e.g., "Win7") in the Virtualbox GUI, (3) specify the desired memory allocation (recommended: 4 GB), (4) when queried by Virtualbox about the creation of a new virtual image, select to "use an already existing image" and point to the path `/usr/local/virtualbox/`, (5) start the virtual machine from the Virtualbox GUI.

In all experiments, use a sampling frequency $f_s = 16000$ Hz, unless specified otherwise. Work with m-files, and save your work regularly, in order to be able to repeat your experiments later on. If possible, save the recorded signals or the resulting spectrograms in a mat-file for later reference. All magnitude plots should be visualized on a dB-scale (for the y-axis). If you are using a stereo loudspeaker set, make sure that one of the two loudspeakers is switched off. Try to optimize the dynamic range of loudspeakers and microphone, but **scale your signals to avoid clipping!**

## Exercise 1-1: Audio playback and recording

In this exercise, you will set up a Simulink model that allows for simultaneous audio recording and playback. The audio signal is taken from a Matlab workspace variable and saved in another workspace variable. This simple Simulink scheme will be the backbone for the Matlab algorithms that you will develop during the upcoming exercise sessions.

[1]For questions and remarks: toon.vanwaterschoot@esat.kuleuven.be

1. In Simulink, adjust the Configuration Parameters in the Simulation menu. In the upcoming exercises, you should always use a discrete solver with variable step, and set the stepsize to "auto".

2. Create a Simulink model "audio_io.mdl" as shown in Fig. 1. Notice that the loudspeaker block and microphone block can have a different name, depending on the Matlab version[2].




Figure 1: Audio playback and recording

3. Create an m-file "initparams.m" in which a user can set the necessary parameters for all the blocks in the scheme (including simulation time, sampling frequency, etc.). This file should also generate a sine signal of 440 Hz with the same length as the specified simulation time, and put it in the workspace variable that is used by the Simulink model.

4. Set the sampling frequency to 16000 Hz. Run initparams.m and start the Simulink model. Make sure a microphone is connected.

5. Create an m-file "playrec.m", that plays the signal recorded in the Simulink model.

## Exercise 1-2: Time-frequency analysis of recorded signals

1. Generate a sine signal of 100 Hz (with respect to the sampling frequency $f_s$). Take a look at the power spectral density (PSD) and the power spectrogram (=PSD as a function of time). Is this what you expected? Are there other frequency components besides the 100 Hz component present in the spectrogram? Why (not)?

[2]The specific wave blocks from Fig. 1 can be found in the (hidden) library 'dspwin32' (type `dspwin32` in the Matlab workspace). However, these wave blocks will disappear in future versions of Matlab.

2. What is the influence of the discrete Fourier transform (DFT) size when calculating the spectrogram? Try to answer first and then verify experimentally.

3. Generate a single signal containing a sum of sines of 50, 100, 200, 500, 1000, 2000, 4000 and 6000 Hz. Take a look at the PSD and the power spectrogram. Is this what you expected? Can you locate the original frequency components in the spectrogram? Now play back and record the signal using the Simulink model that you implemented in Exercise 1-1. Take a look at the (mean) PSD and the power spectrogram of the recorded signal. Is there a difference with the (mean) PSD of the transmitted signal? Can you explain this?

4. Did you properly scale the signal in the previous experiment to avoid clipping? If you did: nice! What do you observe in the spectrum of the recorded signal if clipping occurs? Try it out! Is this harmful for the acoustic modem?

5. *Optional*: Redo 3 with a white noise signal. Does the spectrogram of the recorded white noise signal change significantly in time? Why is this question relevant for the acoustic modem? Move the microphone around while recording. What do you observe now in the spectrogram?

6. *Optional*: Generate a sine signal of 4000 Hz with a length of 15 s. Check the phase spectrogram of this signal. Make sure that the time dimension shifts with a multiple of the period of the sine (why?). Now record the signal and again check the phase spectrogram, using the same time shift. Due to delay, the phase will be different. What else do you observe? Can you explain this?