Audio processing: lab sessions

Session 4: Multi-Channel Filtered-X Active Noise Cancellation

Alexander Bertrand, Jesper Brunnström and Julian Schott.[1]

December 2021

Introduction

In the previous lab session, a multi-channel Wiener filter (MWF) was applied for noise reduction. In this session, another technique is explored, which is referred to as active noise cancellation (ANC). As opposed to directly filtering the noisy signals in the MWF approach, in the ANC approach, noise reduction is achieved by adding an secondary signal that cancels out the unwanted primary noise signal. One of the most used algorithms for ANC is the so-called filtered-X least-mean squares (Fx-LMS), or its normalized version (Fx-NLMS).

In this lab, both a single-channel (one loudspeaker, one microphone) Fx-NLMS ANC as well as a multi-channel version (multiple loudspeakers, multiple microphones) will be implemented. A block diagram depicting a single-channel setup is shown in fig. 1.

The error microphone signal $e(n)$ at (sample) time n can be written as the sum of two components:

- The noise component $d(n)$, obtained by filtering the noise source signal $x(n)$ with the $[R \times 1]$ RIR $\mathbf{p}(n)$ between noise source and microphone
- The secondary signal component, obtained by filtering the loudspeaker signal $y(n)$ with the $[M \times 1]$ RIR $\mathbf{h}(n)$ between the loudspeaker and the microphone.[2]

The $[L \times 1]$ ANC filter $\mathbf{w}(n)$ is obtained by minimizing the mean of the square of the error signal, $e(n)$. However, since the loudspeaker signal $y(n)$ played by loudspeaker is also filtered with the RIR $\mathbf{h}(n)$ before reaching the error microphone (representing the ear), a standard N(LMS) architecture is not suitable as the information of the RIR $\mathbf{h}(n)$ must be included. In general $\mathbf{h}$ is not known and it must be estimated, hence we denote this estimate as $\hat{\mathbf{h}}(n)$. For the rest of this lab, however, we will use the actual RIR in place of its estimate i.e. $\hat{\mathbf{h}}(n) = \mathbf{h}(n)$.

The equation for the error signal is then the following:

$$\underset{[1\times 1]}{e(n)} = \underset{[1\times 1]}{d(n)} + \underset{[1\times M]}{\mathbf{h}^{\mathrm{T}}} \underset{[M\times 1]}{\mathbf{y}(n)}\,, \tag{1}$$

and the corresponding NLMS update is given by the following expression:

$$\underset{[L\times 1]}{\mathbf{w}(n)} = \underset{[L\times 1]}{\mathbf{w}(n-1)} - \frac{\mu}{||\bar{\mathbf{x}}(n)||^2 + \delta} \underset{[L\times 1]}{\bar{\mathbf{x}}(n)} \underset{[1\times 1]}{e(n)}, \tag{2}$$

[1]For questions and remarks: `jesper.brunnstroem@kuleuven.be` and `jschott@esat.kuleuven.be`.

[2]In practice, we would need a reference microphone to obtain some reference of the noise source, however in this lab we assume that we have a perfect reference.

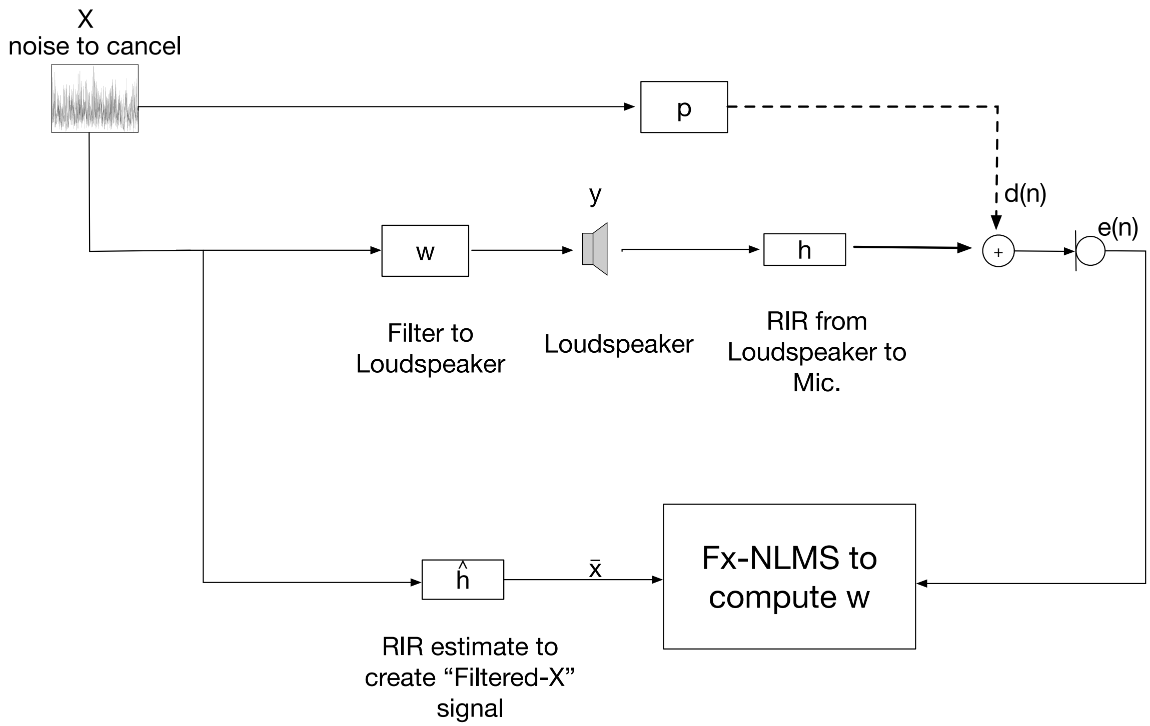

Figure 1: Block diagram depicting a singlee channel filtered-X LMS strategy for ANC.

where μ is the fixed stepsize (e.g. 5×10^{-1}), δ is a regularization factor (e.g. 5×10^{-5}), $\bar{\mathbf{x}}(n)$ is the result of the filtering operation between $x(n)$ and the estimated filter $\hat{\mathbf{h}}$ and $||\bar{\mathbf{x}}(n)||^2$ is the squared norm of $\bar{\mathbf{x}}(n)$.

A multi-channel extension of this ANC is shown in fig. 2. In this case, there are J loudspeakers $j = 1, 2, \ldots, J$ and 2 error microphones (representing the ears) $k = 1, 2$. This leads to a set of estimated RIRs $\hat{\mathbf{h}}_{jk}(n)$ of the true $[M \times 1]$ RIRs, $\mathbf{h}_{jk}(n)$ from each of the loudspeakers to each of the error microphones. Again, we will use the actual RIRs in place of their estimates i.e. $\hat{\mathbf{h}}_{jk}(n) = \mathbf{h}_{jk}(n)$. The $[L \times 1]$ ANC filters, $\mathbf{w}_j$ for $j = 1, 2, \ldots, J$ are computed by minimizing the sum of the mean of the squares of the error microphone signals:

$$C(n) = \sum_{k=1}^{2} \mathbb{E}\{e_k^2(n)\} \tag{3}$$

where $\mathbb{E}\{.\}$ is the expected value, $e_k(n)$ is the n^{th} sample of the k^{th} error microphone signal which is now the sum of the noise component $d_k(n)$, obtained by filtering the noise source signal with the RIR $\mathbf{p}_k$, dand the secondary signal components obtained by filtering the loudspeaker signals with the $\mathbf{h}_{jk}$, i.e.

$$\underset{[1\times 1]}{e_k(n)} = \underset{[1\times 1]}{d_k(n)} + \sum_{j=1}^{J} \underset{[1\times M]}{\mathbf{h}_{jk}^{\mathrm{T}}}\ \underset{[M\times 1]}{\mathbf{y}_j(n)}. \tag{4}$$

The NLMS update for the j^{th} filter $\mathbf{w}_j(n)$ is given by the following expression:

$$\underset{[L\times 1]}{\mathbf{w}_j(n)} = \underset{[L\times 1]}{\mathbf{w}_j(n-1)} - \sum_{k=1}^{2} \frac{\mu}{\sum_{i=1}^{J} ||\bar{\mathbf{x}}_{ik}(n)||_2^2 + \delta} \cdot \underset{[L\times 1]}{\bar{\mathbf{x}}_{jk}(n)} \underset{[1\times 1]}{e_k(n)} \tag{5}$$

or, in stacked form, by:

$$\mathbf{w}_j(n) = \mathbf{w}_j(n-1) - \frac{\mu}{||\bar{\mathbf{X}}_j(n)||_{\mathrm{F}}^2 + \delta} \bar{\mathbf{X}}_j(n)\mathbf{E}(n) \tag{6}$$

where $||.||_{\mathrm{F}}$ is the Frobenius norm,

$$\bar{\mathbf{X}}_j(n) = [\bar{\mathbf{x}}_{j1}(n)\ \bar{\mathbf{x}}_{j2}(n)] \tag{7}$$

$$\mathbf{E}(n) = [e_1(n)\ e_2(n)]^T, \tag{8}$$

and $\bar{\mathbf{x}}_{jk}(n)$ is the (filtered-X) vector including the results of the filtering operation between $x(n)$ and $\hat{\mathbf{h}}_{jk}$.

Exercise 4.1: Implementation of the Fx-NLMS

1. In the simulation environment, re-create a scenario similar to the one in previous lab sessions, i.e. a microphone array with two microphones and an inter-microphone distance of 15 cm, $J = 5$ sound sources placed freely around the microphone array, and make sure that all of them are at a similar distance from the microphone array (these will represent the five loudspeakers). Set $T = 0.5$ s and set the sampling rate to 8 kHz. Now place a noise source in the scenario, which will generate the noise signal that we intend to remove from the microphone signals. Is there any constraint to consider for the placement of the noise source? For the sake of a single-channel ANC implementation, in this first exercise you will only make use of one of the microphones (left ear), one of the loudspeakers (the first you placed), and the noise source. Store the resulting RIRs, and use them in the sequel. You can truncate your RIRs in the code to speed up the computations, making sure that you keep enough information. Work on the provided file `ANC_FxLMS_skeleton.m` to implement this exercise.

2. Create a variable `filt_noise` which contains the filtered version of the noise signal observed at the microphone (use the wav file `White_noise1.wav` for this).

3. Implement the Fx-NLMS. You can simplify the implementation by assuming that $M = L$.

4. Test the algorithm in the dummy case when no desired source signal is playing, i.e. $y(n)$ is just a filtered version of the reference noise $x(n)$. Start experimenting with the algorithm on short portions of your signal (e.g. 0.5 s) and once you think it is working use longer portions (e.g. 10 s). Try different stepsizes in order to find a good convergence speed.

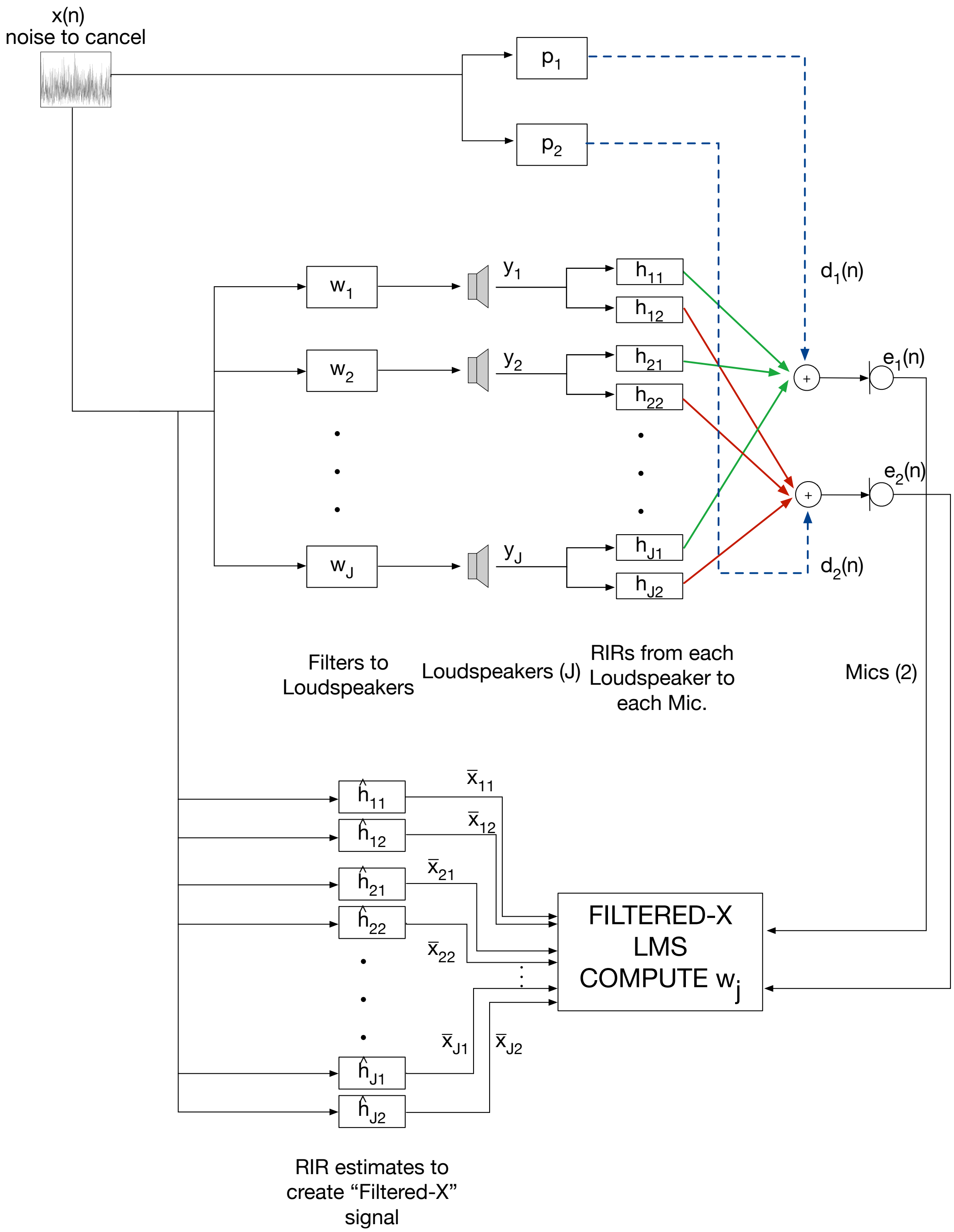

Figure 2: Block diagram depicting a setup using the multi-channel filtered-X LMS for ANC.

5. Overlay the plots of the variable `filt_noise` with the error signal $e(n)$ to show that the Fx-LMS is converging.

6. Calculate an estimate of the noise suppression as follows:

$$10 \log_{10} \frac{\mathbb{E}\{e_1^2(n)\}}{\mathbb{E}\{d_1^2(n)\}} \tag{9}$$

7. Does $e(n)$ go to zero, i.e. should we expect perfect cancellation upon convergence?

Exercise 4.2: Implementation of the MFx-NLMS

1. Use the same scenario as in the previous exercise. Here, however, you are going to use all the loudspeakers and microphones. Create a new file `ANC_MFxLMS_skeleton.m` where to implement this exercise. Store the resulting impulse responses, and use them in the sequel.

2. Create a 2-column matrix variable `filt_noise` which contains the filtered version of the noise signal observed by the two microphones.

3. Implement the MFx-LMS

4. Test the algorithm in the dummy case when no desired source signal is playing, i.e. $y(n)$ is just a filtered version of the reference noise $x(n)$. Start experimenting with the algorithm on short portions of your signal (e.g. 0.5 s) and once you think it is working use longer portions (e.g. 10 s). Try different stepsizes in order to find a good convergence speed.

5. Calculate an estimate of the noise suppression as before at the left ear.

6. Overlay the plots of $d_k(n)$ and $e_k(n)$ (k = 1,2).

Exercise 4.3: Implementation of the MFx-NLMS in a 3D audio scenario

1. Add the possibility to test the ANC in an actual 3D audio scenario. How does the scheme in fig. 2 need to be changed to achieve this?

2. Use the same scenario from Exercise 4.2 and adapt the MFx-NLMS implementation to allow for the possibility of adding 3D audio speech. Do this in the same script `ANC_MFxLMS_skeleton.m` where you will just add a boolean flag to decide whether to add or not the 3D audio speech.

3. Create a 2-column matrix variable `binaural_sig` which contains the 3D audio version of the speech signal observed by the two microphones, representing left and right ear of the listener.

4. Scale the signal `binaural_sig` such that the input SNR between the left channel of `binaural_sig` and the left channel `filt_noise` is of the order of 0 dB. The SNR should be calculated on, at least, a 5 s portion of the signal.

5. Overlay the plots of the left (right) channel of the variable `filt_noise` with the left (right) channel of the error signal $e_k(n)$.
6. How does the convergence of the algorithm compare to the noise only case? How do you expect this would be affected in practice?