Acoustic modem project

Session 7 (+8): OFDM transmission over the acoustic channel with adaptive channel equalization in decision-directed mode and OFDM transmitter-side beamforming

Alexander Bertrand, Paschalis Tsiaflakis, Hanne Deprez, Amin Hassani, Niccoló Antonello[1]

December 2017

Goal: Using the OFDM modem to transmit an image over the acoustic channel with adaptive channel equalization in decision-directed mode. Using a stereo loudspeaker set-up to improve the BER by transmitter-side beamforming.

Requirements: Matlab/Simulink in **Windows**, a sound card, a loudspeaker and a microphone.

Required files from DSP-CIS website: *recplay.mdl* (see session 1), *imagetobitstream.m*, *bitstreamtoimage.m*, *image.bmp* (see session 4).

Required files from previous sessions: *initparams.m*, *alignIO.m*, *qam_mod.m*, *qam_demod.m*, *ofdm_mod.m*, *ofdm_demod.m*, *ber.m*, *main.m*, *visualize_demod.m*

Outcome: 8 m-files: *DDequalization.m*, *transmit_pic.m*, *visualize_demod.m*, *initparams_stereo.m*, *ofdm_mod_stereo.m*, *ofdm_demod_stereo.m*, *transmit_pic_stereo_a.m*, *transmit_pic_stereo_b.m*

Deliverables: 2 m-files: *milestone4.m*, *transmit_pic_stereo_b.m*

In sessions 5 and 6, we have explored two methods to estimate the acoustic channel response in the frequency domain, i.e., packet-based training or pilot tones. The channel estimates could then be used to equalize the channel so as to obtain reliable OFDM transmission. However, block-training and pilot tones based schemes require channel resources to transmit training data. Consequently, these channel resources cannot be used for the transmission of actual data. In this session we will focus on an alternative method to adaptively estimate and equalize the channel in the frequency domain without sacrificing channel resources. For this, in the first part of this exercise session, we will use adaptive filters for channel equalization in decision-directed (DD) mode (see also chapter 10).

In session 4, we have explored two methods to improve the bit error rate: ON-OFF bit loading and adaptive bit loading. These two methods result in longer transmission times as not all frequency bins are used to the fullest extent. Another method to improve the BER is called beamforming. Transmitter-side beamforming is based on a multichannel transmission scheme, where we use two transmitters (= two loudspeakers) that steer towards the receiver (= the

[1]For questions and remarks: `jeroen.verdyck@esat.kuleuven.be`

microphone). In the second part of this session, we will implement a (fixed) transmitter-side beamformer.

1 PART I: Exercise 7-1: Implementation of an LMS adaptive filter for frequency domain adaptive channel equalization in DD mode

In this exercise we focus on one individual tone (i.e., frequency bin) for which the OFDM transmission can be modelled by a simple model as follows:

$$Y_k = H_k X_k + N_k, \tag{1}$$

with Y_k the received QAM symbol on tone k, H_k the (complex) channel response for tone k, X_k the transmitted QAM symbol on tone k, and N_k the received noise on tone k.

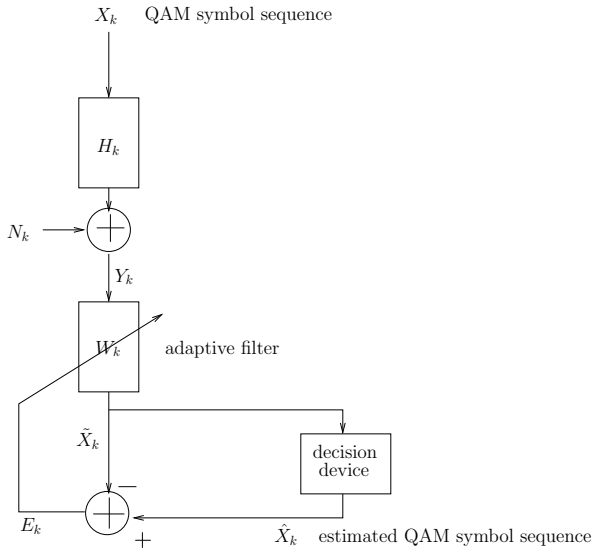


Figure 1: Adaptive filter for channel equalization in DD mode

In Figure 1 the adaptive filter for channel equalization in DD mode is shown. The decision device maps the input to the closest QAM constellation point. Explain the operation of this adaptive filter. What are the input signal and desired output signal of the adaptive filter? Which is the signal that we are interested in? How does the filter coefficient W_k relate to H_k?

Exercise: create an m-file *DDequalization.m* in which the following steps are implemented:

1. Construct a QAM symbol sequence of length 1000. This corresponds to the transmitted QAM symbol sequence X_k at a single tone k. Choose a complex value for the H_k and generate the received symbol sequence Y_k based on (1).

2. Implement the adaptive filter as shown in Figure 1 using an LMS or NLMS updating scheme (you can omit the noise). Note that the adaptive filter in our case has only one (complex) tap. Choose an initial value for $W_k = \frac{1}{H_k^*} + \Delta$, where Δ corresponds to a small deviation such that the decision device still makes correct decisions (the conjugation of H_k is explained later).

   Hint: It is noted that the (N)LMS formula for complex numbers is similar to the real case, but the transpose operator (T) should be replaced by a conjugate transpose operator (H), e.g., for NLMS (using the notation of the lecture slides):

   $$\mathbf{w}(L+1) = \mathbf{w}(L) + \frac{\overline{\mu}}{\alpha + \mathbf{u}_{L+1}^H \mathbf{u}_{L+1}} \mathbf{u}_{L+1} \left(d_{L+1} - \mathbf{w}(L)^H \mathbf{u}_{L+1} \right)^*$$

   In this project, we use a single-tap filter such that the conjugate transpose (H) becomes merely a conjugation (*). Note that there is also a conjugation in the filtering operation, i.e., the filtering in Fig. 1 is implemented as $W_k^* Y_k$.

3. Does the adaptive filter converge? Experiment with different values for the stepsize parameter.

4. Generate a figure that plots the error signal (y-axis), i.e. the difference between the adaptive filter coefficient W_k^* and the inverse channel coefficient $\frac{1}{H_k}$, over the iterations (x-axis), for different stepsizes.

5. What happens when the channel changes (slow and fast)? What does this mean for the OFDM transmission?

Only if the adaptive filter converges correctly, you can proceed to the next exercise!

2 PART I: Exercise 7-2: OFDM transmission over the acoustic channel with adaptive (frequency domain) channel equalization in DD mode

In this exercise, we will transmit data packets over the acoustic channel using the adaptive channel equalization scheme of Exercise 7-1 for each tone. To

obtain good initial values for the frequency response, we will add a packet of training frames at the beginning of the QAM stream.

1. Create an m-file *transmit_pic.m* in which the following steps are implemented (reuse your code from *transmit_pic.m* of session 6).

   - Create a vector `trainblock` containing the QAM training symbol sequence that will be used (see exercise 5-1). Furthermore, create a QAM symbol sequence `qamStream` representing the data obtained from the image *image.bmp* (see session 4).
   - Modify *ofdm_mod.m*, such that the OFDM modem transmits a single training packet first and then *all* data packets (without intermediate training). The training packet contains L_t training frames, where L_t can be set by the user, and contains the QAM training symbols from the vector `trainblock`. The data packets contain the data from the `qamStream` vector.
   - Use the simulink model *recplay.mdl* to play the OFDM signal in the loudspeaker, and record it with the microphone. Align input and output using the function *alignIO.m* (see session 5).
   - Modify *ofdm_demod.m*, such that the OFDM modem uses the received training packet to obtain an initial estimate of the channel response (see session 5). Use the inverse of this initial estimate as the initial value for the adaptive filters (for each tone). Use the DD adaptive equalization of Exercise 7-1 on each tone. The function *ofdm_demod.m* should return a demodulated QAM symbol sequence `rxQamStream` and a matrix `channel_est` that contains the different channel estimates in its columns (one for each data frame).
   - Use the demodulated data in `rxQamStream` to reconstruct the transmitted image, and compute the BER.
   - Find good values for the stepsizes so that the channel tracking performance is sufficiently fast.

2. Adapt the m-file *visualize_demod.m* of session 6 to visualize the OFDM function (with DD adaptive equalization) during its operation. How does the DD adaptive equalization approach compare with the block-training and pilot tones based approaches of session 5 and 6?

3. Run experiments with and without on-off bitloading.

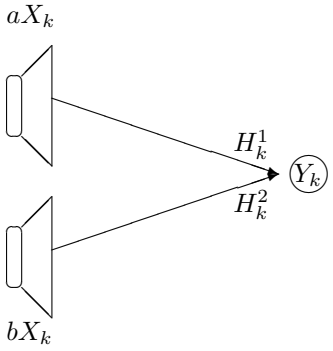
4. Slowly increase and decrease the volume during the transmission. Can your (N)LMS filter track the changes in the channel? If not, can you explain why?

5. Slowly move the microphone around during the transmission. Can your (N)LMS filter track the changes in the channel? If not, can you explain why?

3 PART II: Exercise 7-3: Implementation of a fixed transmitter-side beamformer

We will use the stereo loudspeaker setup to implement transmitter-side beamforming, i.e. to steer the audio signal more effectively towards the microphone. The advantage of the OFDM format is that such transmitter-side beamforming (filtering) can be realized with simple per-frequency bin scalar multiplications. In frequency bin k, each loudspeaker then transmits a scaled version of the symbol X_k, i.e. loudspeaker 1 transmits aX_k and loudspeaker 2 transmits bX_k. To enable a fair comparison with the single loudspeaker case, a and b are such that $\sqrt{a^*a + b^*b} = 1$, meaning that the available power budget is shared between the two loudspeakers. If (in frequency bin k) one loudspeaker transmits more power, the other loudspeaker transmits less power, so that the sum power is constant.

It can be shown that the scalars a and b that maximize the received SNR are $a = \frac{H_k^{1*}}{\sqrt{H_k^1 H_k^{1*} + H_k^2 H_k^{2*}}}$ and $b = \frac{H_k^{2*}}{\sqrt{H_k^1 H_k^{1*} + H_k^2 H_k^{2*}}}$, where H_k^1 and H_k^2 represent the channel transfer function (in frequency bin k) from loudspeaker 1 and 2 to the microphone, respectively.



$$
\begin{aligned}
Y_k &= \begin{bmatrix} H_k^1 & H_k^2 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} X_k + N_k && (2)\\
&= \begin{bmatrix} H_k^1 & H_k^2 \end{bmatrix} \begin{bmatrix} \frac{H_k^{1*}}{\sqrt{H_k^1 H_k^{1*} + H_k^2 H_k^{2*}}} \\ \frac{H_k^{2*}}{\sqrt{H_k^1 H_k^{1*} + H_k^2 H_k^{2*}}} \end{bmatrix} X_k + N_k && (3)\\
&= \left(\sqrt{H_k^1 H_k^{1*} + H_k^2 H_k^{2*}} \right) X_k + N_k && (4)\\
&= H_k^{1+2} X_k + N_k && (5)\\
\hat{X}_k &= \frac{1}{H_k^{1+2}} Y_k && (6)
\end{aligned}
$$

1. First, we will implement the fixed transmitter-side beamformer (use equations 3 and 6) and simulate the data transmission by using two predefined (random) impulse responses. Create an m-file *transmit_pic_stereo_a.m* in which the following steps are implemented:

   (a) Generate two random impulse responses of user defined length (modeling the acoustic channels from loudspeaker one and two to the microphone). Are these a good model for the acoustic channels? Why (not)?

   (b) Create a Matlab function *fixed_transmitter_side_beamformer.m* that takes the two impulse responses as inputs and outputs the optimal

per-frequency bin scalars a and b. Calculate also the combined channel transfer function H_k^{1+2} and plot this together with H_k^1 and H_k^2 in one figure. What can you conclude about the combined channel transfer function H_k^{1+2}?

(c) Adapt your Matlab function *ofdm_mod.m* to *ofdm_mod_stereo.m* such that it outputs two ofdmstreams (`ofdmStream1` and `ofdmStream2`) using the optimal per-frequency scalars a and b. This function should still be working with training frames and data frames.

(d) Generate the received microphone signal by convolving the two OFDM signals generated in 1c with the two impulse responses and adding the results. Do not (yet) add any noise to the microphone signal!

(e) For the receiver operation, create an m-file *ofdm_demod_stereo.m* by adjusting *ofdm_demod* such that the channel is correctly equalized. Note that only the combined channel transfer function H_k^{1+2} is important for the equalization. Do you need to change a lot in this function?

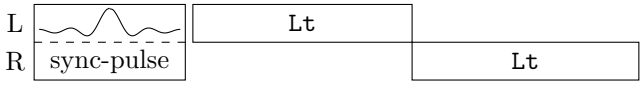
(f) Check the BER. **You can only continue when the BER is exactly zero!**

(g) Now add noise to the received signal and check the BER in the case of mono transmission with loudspeaker 1 ($a = 1, b = 0$), mono transmission with loudspeaker 2 ($a = 0, b = 1$) and stereo transmission with both loudspeaker 1 and 2 (optimal a and b). Do you see an advantage of using two loudspeakers?

2. Now we will measure the true transfer functions between loudspeaker 1 and the microphone (H^1) and between loudspeaker 2 and the microphone (H^2). Create an m-file *transmit_pic_stereo_b.m* in which the following steps are implemented:

(a) Create a Matlab function *initparams_stereo.m* that is the same as *initparams.m*, but now the variable `simin` contains a left and right loudspeaker signal in its two columns. In the previous sessions, we used mono transmission and the two columns were identical.

(b) In one playback session, emit first `Lt` training frames from loudspeaker 1, while loudspeaker 2 is silent, and then `Lt` training frames from loudspeaker 2, while loudspeaker 1 is silent. Estimate the transfer function `H_1` using the first received `Lt` training frames and the transfer function `H_2` using the last received `Lt` training frames by reusing the function *ofdm_channelsest.m* from session 5 on the correct signal segments. The structure of the emitted signal is:

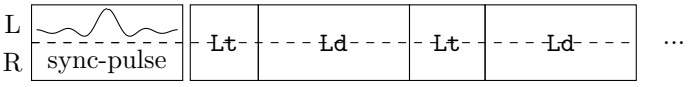


(c) Execute *fixed_transmitter_side_beamformer.m* using the estimated channel transfer functions H^1 and H^2 and look at the figure. Do H^1 and H^2 look alike? Do the transfer functions vary a lot for different microphone positions?

3. We will now transmit the QAM symbols with an optimal power distribution over both loudspeakers. Adjust the m-file *transmit_pic_stereo_b.m* as follows:

   (a) Use *ofdm_mod_stereo* to determine the two OFDM streams that will be sent by loudspeaker 1 and loudspeaker 2, based on the optimal per-frequency bin scalars a and b you estimated in 2.

   (b) Transmit the data over the acoustic channel using *initparams_stereo.m* and the Simulink model *recplay.m*.



   (c) Demodulate the received signal with *ofdm_demod_stereo.m*.

   (d) Check the BER in the case of mono transmission with loudspeaker 1 ($a = 1, b = 0$), mono transmission with loudspeaker 2 ($a = 0, b = 1$) and stereo transmission with both loudspeaker 1 and 2 (optimal a and b). Do you see an advantage of using two loudspeakers? Try this for different microphone positions and determine in which position you get the biggest advantage from the transmitter-side beamformer. Experiment with different values of `CP`, QAM constellation size `M`, number of training blocks `Lt` and `DFTsize`.

4. Do you think it is possible to replace the channel estimation using training frames by an adaptive filter in decision-directed mode (cf. section 2)?

4 Milestone demo

The fourth milestone demo takes place at the start of the eighth exercise session. You will be asked to show the following demo(s). Before the start of the eighth exercise session, e-mail the Matlab code of your demo(s) in one zip-package to

`jeroen.verdyck@esat.kuleuven.be`.

Make sure to mention your group number as well as the names of all group members in the name of the zip-file. Do not send additional files (only the files that are required to execute the demo).

- Demo 1: A single m-file named *milestone4.m* that
  - first shows the convergence of your (N)LMS filter in the ideal case of ex. 7-1 (plot the error versus iterations as in subtask 4 of ex. 7-1).

  - then runs your files *transmit_pic.m* and *visualize_demod.m*. It should do this twice: once with `BWusage=100;` (no bitloading) and once with `BWusage=50;` (add a `pause` command between both experiments). Also show the BER in each experiment. If you have used adaptive bitloading in exercise 6-2, you can choose your own bitloading strategy in the second experiment, but explain it to the TA.

- Demo 2: A single m-file named *transmit_pic_stereo_b.m*. This should demonstrate the performance of your modem with transmitter-side beamformer with transmission over the acoustic channel.