Signal Processing and Speech Communication Lab
Graz University of Technology

AS

Adaptive Systems—Homework Assignment 3

Name(s) Matr.No(s).

The analytical part of your homework (your calculation sheets) **as well as** the MATLAB simulation protocol have to be delivered as **hard copy** to our mailbox at Inffeldgasse 16c, ground floor, no later than **2017/2/24**. Use a printed version of **this entire document** as the title pages and fill in your **name(s) and matriculation number(s)**.

Your MATLAB programs (`⋆.m` files) and the simulation protocol (in pdf format!) have to be submitted via **e-mail** to the address `hw2.spsc@tugraz.at` no later than **2017/2/24**. The subject of the e-mail consists of the assignment number and your matriculation number(s) **"Assignment3, MatrNo1, MatrNo2"**. You have to zip (or tar) all your homework files to one single file with the name `Assignment3_MatrNo1_MatrNo2.zip`, e.g., `Assignment3_9833280_9933281.zip`, which has to be attached to the e-mail.

IMPORTANT:
A justification of all your answers is mandatory to achieve all points!
You need to document all necessary plots, else you will lose points if you do not comment on your observations!

Name(s) **Matr.No(s).**

MATLAB Problem 3.1 (12 Points)—Beamforming Application

Sensor array processing has broad applications in diverse fields such as wireless communications, microphone arrays, radar or seismic explorations. Direction of arrival (DOA) estimation and beamforming are two basic areas of sensor array processing that are common to its diverse application. In this example we want to exemplify beamforming using a simple antenna configuration.

In the application of beamforming, we desire to combine the measurements of an antenna array in order to maximize its gain along a particular direction. The antenna array is assumed to be far from a source radiating an electromagnetic wave of the form

$$r(t) = s(t)e^{j\omega_c t}$$

where ω_c denotes the carrier frequency and $s(t)$ denotes the envelope, also called the baseband signal. Since the source is sufficiently distant form the antenna array, the arriving wavefronts can be assumed to be planar at the array. The output of the array is obtained by linearly combining the measurements at the antennas. These measurements are subjected to noise with the noise at antenna j at time t denoted by $v_j(t)$ Let θ denote the direction of arrival of the wavefront relative to the plane of the antennas, as indicated in Figure 1. The interval of time τ that is needed for the wave to propagate from one antenna to the next is given by

$$\tau = \frac{2\pi d \cos\theta}{\omega_c \lambda}$$

The signal arriving at the n-th antenna at time t is of the form

$$s(t)e^{j\omega_c t}e^{j\frac{2\pi n}{\lambda}d\cos\theta}$$

At the antennas we observe the noisy observation

$$\mathbf{y} = \mathbf{h}r(t) + \mathbf{v}$$

where $\mathbf{h} = [1 \; e^{j\frac{2\pi}{\lambda}d\cos\theta} \; \dots \; e^{j\frac{2\pi(M-1)}{\lambda}d\cos\theta}]^T$and $\mathbf{v}$ denotes the noise vector.

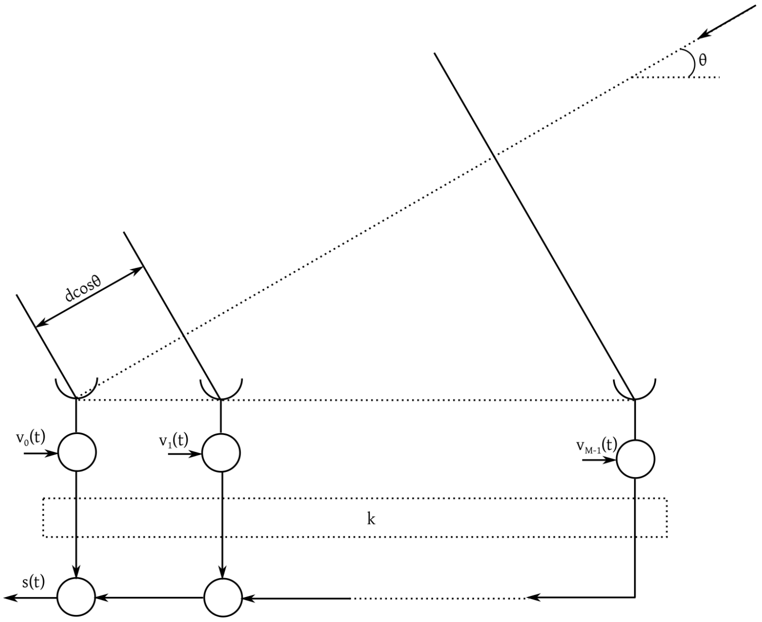

Figure 1: A uniformly-spaced array of antennas.

The beamforming problem is then to solve

$$\min_{\mathbf{c}} \mathbf{c}^H \mathbf{R_{vv}} \mathbf{c} \quad \text{subject to} \quad \mathbf{c}^H \mathbf{h} = 1$$

This constrained estimation problem has the solution

$$\mathbf{c_o}^H = \frac{\mathbf{h}^H \mathbf{R_{vv}}^{-1}}{\mathbf{h}^H \mathbf{R_{vv}}^{-1} \mathbf{h}}$$

and we obtain the estimated signal

$$\hat{\mathbf{s}}(t) = \mathbf{c_o}^H \mathbf{y}$$

Note, that this result is only valid if the noise at each sensor is uncorrelated with the noises at the other sensors, which is known as spatially whiteness. Also, the noise processes must fulfill the whiteness property. Consider the same uniform linear antenna array consisting of 4 elements, spaced by $d = \lambda/5$. We assume that the noise signal at each antenna is white with real and imaginary parts that are Gaussian with $\sigma_v = 0.2$. Now we are assuming a sinusoidal baseband signal $s(t)$ which gets sampled at a rate of 16kHz.

(a) Design an optimal beamformer with unit response along the direction $\theta_0 = 30°$. Therefore, simulate your baseband signal from 50Hz to 5kHz in 100Hz steps. Show the baseband signal and the estimated outcome of the beamformer in one plot. Also, plot the received signals at the different antennas. Explain you results.

(b) Assume now that the input signal is the sum of two sinusoids arriving along different directions. One impinges on the antennas at 30° while the other arrives at 45°. Simulate the operation of the beamformer by using the input signal

$$s(t) = \cos(2\pi f_1 t) + 0.2 \sin(2\pi f_2 t)$$

where $f_2 = 2f_1$. Again, simulate the scenario for different frequencies $f_1 \in [50\ 500\ 1000\ 2000\ 5000]$ Hz. Plot the baseband signals and the arriving signals at the antennas. Also plot the beamformer output and the baseband signal (overlap both signals).

(c) Now, we assume that the noise signals across the antennas are spatially correlated with an exponential autocorrelation function equal to

$$r[k] = 0.85^{|k|}$$

To generate the noise vector we make use of the so-called *Cholesky* factorization such that we can factorize the autocorrelation matrix as

$$\mathbf{R_{vv}} = \mathbf{P}\mathbf{P}^H$$

where $\mathbf{P}$ is a $M \times M$ lower triangular matrix. You can use the MATLAB function `chol` for Cholesky decomposition. You can then model the noise vector as

$$\mathbf{v} = \mathbf{P}\mathbf{e}$$

where $\mathbf{e}$ is a vector which fulfills the property of whiteness. Repeat tasks **(a)** and **(b)** for this case.

(d) Use your designed beamformer from task **(a)** and use the same setup. Perform a simulation that varies the direction of arrival of $s(t)$ between 1° and 180° in increments of 1°. For each direction θ determine the output of the beamformer and its averaged power

$$\hat{P}(\theta) = \frac{1}{N} \sum_{n=0}^{N-1} |\hat{s}[n]|^2$$

where N denotes the number of samples. The average power of the input signal is clearly to unity. The power gain of the beamformer at direction θ is then given by

$$G = 10 \log \hat{P}(\theta)$$

Use the MATLAB command `polar` to generate a plot of the power gain of the antenna as a function of θ. Simulate two cases: a beamformer with 4 antennas and a beamformer with 25 antennas.

Name(s) **Matr.No(s).**

MATLAB Problem 3.2 (12 Points)—Adaptive Channel Equalization

Consider the following channel given by the system function

$$C(z) = 0.5 + 1.2z^{-1} + 1.5z^{-2} - z^{-3}$$

The goal of this task is to design an adaptive equalizer. The structure of the equalizer is shown in Figure 2. The symbols $\{s[n]\}$ are transmitted through the channel and corrupted by additive complex-valued white noise $\{w[n]\}$. The received signal $\{y[n]\}$ is processed by the FIR equalizer to generate symbol estimates $\{\hat{s}[n-\Delta]\}$, which are then fed into the decision device. The equalizer possesses two modes of operation

(*i*) training mode

(*ii*) decision-directed mode

In the training a delayed replica of the input sequence is used as a reference. In the decision-directed mode, the output of the decision-device replaces the reference sequence.

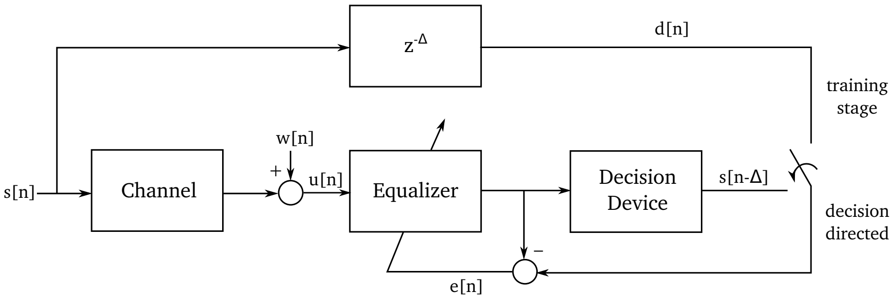

Figure 2: Linear Channel equalizer with training stage and decision-device.

(a) Write a program that trains the adaptive filter with 500 symbols from a QPSK constellation, followed by decision-directed operation during 5000 symbols from a 16-QAM constellation. Choose the noise variance σ_w^2 in order to enforce an SNR level of 30dB at the input of the equalizer. Choose $\Delta = 15$ and an equalizer length of $N = 35$. Plot the scatter diagrams of $\{s[n], y[n], \hat{s}[n-\Delta]\}$
Hint: Note that symbols chosen from QAM constellations do *not* have unit variance. For this reason, the noise variance needs to be adjusted properly for different QAM orders in order to enforce the desired SNR level.

(b) For the same setting as part **(a)**, plot and compare the scatter diagrams that would result at the output of the equalizer if training is performed only for 150, 300 and 500 iterations.

(c) Now assume the transmitted data are generated from a 256-QAM constellation rather than a 16-QAM constellation. Plot the scatter diagrams of the output of the equalizer. You can assume the same training procedure as above.

(d) Generate symbol-error-rate (SER) curves versus signal-to-noise ratio (SNR) at the input of the equalizer for $(4, 16, 64, 256)$-QAM data. Let the SNR vary between 5dB and 30dB in increments of 1dB.

(e) Now, design a decision-feedback equalizer according to Figure 3 with $L = 10$ feedforward taps and $Q = 2$ feedback taps. Use $\Delta = 7$ and plot the resulting scatter diagram of the output of the equalizer. Repeat for $L = 20$, $Q = 2$ and $\Delta = 10$. In both cases, choose the transmitted data form a 64-QAM constellation.

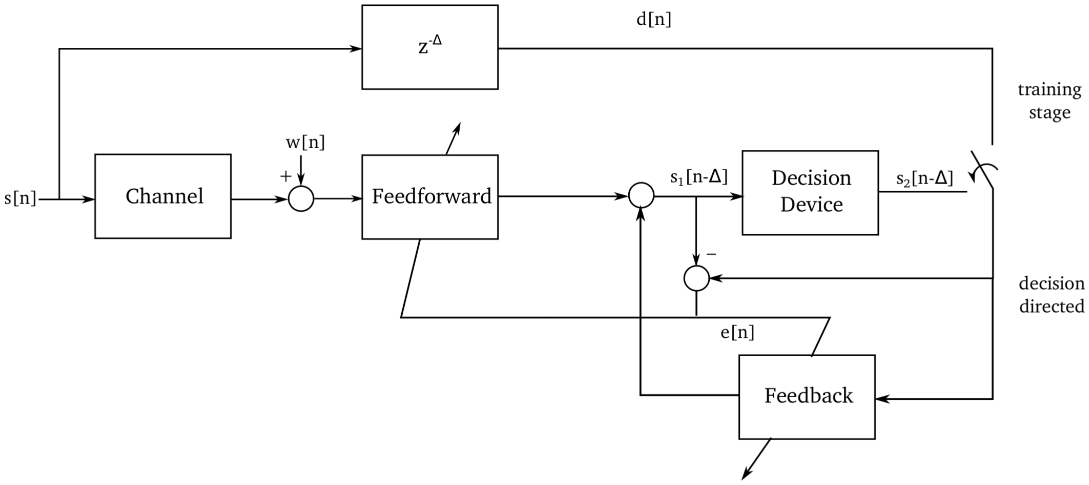

Figure 3: Decision-Feedback equalizer structure.

(f) Generate SER curves versus SNR at the input of the DFE for $(4, 16, 64, 256)$-QAM data. Let the SNR vary between 5dB and 30dB in 1dB increments. Compare the performance of the DFE with that of the linear equalizer of task **(d)**.

(g) Load the file `channel.mat` which you find on the webpage. This file contains the impulse response sequence of a more challenging channel with spectral nulls. Plot the impulse response and the frequency response of the channel, where you can observe the spectral nulls. Set the SNR level at the input of the equalizer to 40dB and select a linear equalizer structure with 55 taps. Set the delay to $\Delta = 30$. Train the equalizer with an NLMS algorithm for 2000 iterations before switching to decision-directed operation. Plot the resulting scatter diagram of the output and plot the frequency- , as well as the impulse response of the equalizer and the *combination* of both.

(h) BONUS: Train the setting from **(g)** again using a RLS algorithm for 100 iterations before switching to the decision-directed operation, and plot the resulting scatter diagram. Compare both diagrams.

Name(s) **Matr.No(s).**

MATLAB Problem 3.3 (9 Points)—(N)LMS Performance Analysis

For the system identification problem shown in Figure 4 we want to determine the convergence time constants τ_i using the *ensemble-averaged*[1] misalignment vector

$$\mathbf{v}[n] = \mathbf{c}[n] - \mathbf{c}_{MSE} = \mathbf{c}[n] - \mathbf{h}$$

The input signal $x[n]$ is a zero-mean, white Gaussian process with unit variance. The desired signal $d[n]$ is corrupted by additive white Gaussian noise with zero mean and variance $\sigma_\nu^2 = 0.008$. The impulse response of the unknown system is given as

$$\mathbf{h} = [0.6, 0.2, 0.4]^T$$

and $N = \dim(\mathbf{c}[n]) = \dim(\mathbf{h})$.

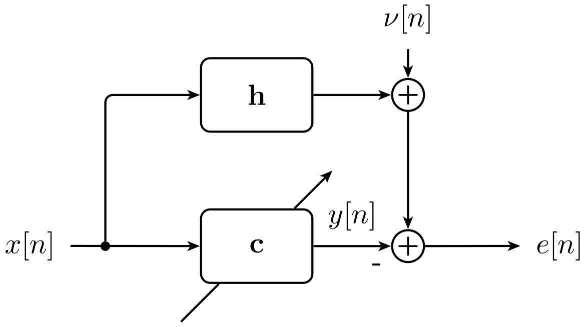

Figure 4: The system identification problem in a noisy environment.

For the following tasks, use a NLMS implementation. Call your function multiple times and use the coefficient matrices to compute the ensemble averages of the misalignment vector. Initialize you coefficient vector with a zero vector. Write a MATLAB script to plot

$$\ln \frac{\mathbb{E}(v_k[n])}{\mathbb{E}(v_k[0])}$$

as a function of time n for all components k of the vector. Your script should automatically determine the time constants τ_k and print them in the legend of the plot. On top of the curves, plot the logarithm of the MSE as a thick line, i.e.,

$$\ln \frac{\mathbb{E}\left(e^2[n]\right)}{\mathbb{E}\left(e^2[0]\right)}.$$

[1] For every considered time instant n we average over data obtained from independent trials, i.e., using multiple realizations of the input and noise processes.

(a) Investigate the effect of the step-size parameter μ for both the LMS and the NLMS algorithms. Use, e.g., $\mu = \{0.0001, 0.001, 0.01, 1\}$. Are the algorithms stable in all these cases?

(b) Now set the step size to $\mu = 0.001$ and vary the variance of the input signal $x[n]$. Use $\sigma_x^2 = \{0.2, 1, 5\}$. What is the difference between the LMS and the NLMS?

(c) Let $x[n]$ be a zero-mean, unit variance, white input process $w[n]$ filtered by a two-tap moving average filter, i.e.,

$$x[n] = \frac{w[n]}{2} + \frac{w[n-1]}{2}.$$

Use again a small step size of $\mu = 0.001$ and plot convergence behavior of the misalignment vector. What can you observe? Can you still determine the time constants? What can you say about the covergence behavior of the MSE as a function of time?

(d) In all previous tasks, how did the MSE behave *after* convergence?

(e) For all evaluated scenarios compute the *excess MSE*. What can you observe?

Name(s) **Matr.No(s).**

MATLAB Problem 3.4 (7 Points)—Bonus Problem and Contest: Periodic-Interference Cancelation

On our course web page you will find a wave file (8 kHz sampling rate) that contains speech with interfering tones turning on and off (DTMF signals). Implement an interference cancelation system of your choice in MATLAB that removes/attenuates the disturbing DTMF signals. For instance, you may choose an LMS-based predictor or a bank of adaptive/non-adaptive notch filters. Also switching between adaptive/non-adaptive filters may be a possibility. For an adaptive filter, an adaptive step size may be considered. The only requirement the system has to satisfy is a maximum input-to-output delay of 25 ms. Note that any delay $\leq$ 25 ms is accepted (but you have to tell us the exact delay Δ in samples) and that SNR is the only benchmark feature.

For evaluation, we provide you with the clean speech signal $s[n]$ (of course, the interference cancelation system is NOT allowed to use the clean speech signal). The SNR is calculated as $SNR_{dB} = 10\log_{10}\frac{\sigma_s^2}{\sigma_r^2}$, where $r[n]$ is any error on the output signal $\hat{s}[n]$ of your system ($r[n] = s[n-\Delta] - \hat{s}[n]$).