ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE
School of Computer and Communication Sciences

Handout 21 Modern Digital Communications
Final Exam December 17, 2025

Name:

Note:

- You have 2 hours to work on the exam.
- The exam is closed book, but you are allowed one double-sided A4 page of handwritten notes, with NO processing (plain and pure handwriting on paper). Resources from the internet as well as code written outside this exam are not allowed (unless the code is written on the sheet of handwritten notes).
- The code will be evaluated according to the usual criteria, namely correctness, speed, form, and readability. Short comments that allow us to follow what you are doing will improve readability.
- You will upload (to Moodle) your solution to the problems that require writing MATLAB or Python code. Do so in a single archive.

To get started with the exam, do the following:

1. Close all the windows and programs on your laptop.
2. Launch the MATLAB/Python editor and close all the tabs (previously written code).
3. From Moodle, download the file `mdc_final_2025.zip`. Unzip the file to create the directory `mdc_final_2025`. For the rest of the exam you are required to work inside that directory. The MATLAB/Python files for Problem n, are found in subfolder pn.
4. Turn your WiFi off until you are ready to upload your solutions.
5. Wait until you receive the go-ahead signal.

PROBLEM 1. 20 points (MATLAB/Python)

You are given the received samples (stored in `rx_signal.mat`) of an OFDM signal transmitted over a discrete-time AWGN channel. The OFDM transmitter uses a total number of $N = 256$ carriers and a cyclic prefix of length $L = 20$. All the carriers are used for transmission. The transmitted signal is affected by noise and the channel impulse response is assumed to be ideal, that is, $h(t) = \delta(t)$. There is no Doppler. The transmitter sends as well a preamble (a P/N sequence, stored in `pn_sequence.mat`), so that the receiver can detect the start of the first OFDM data block within the received signal.

Please write your code in the script `p1.m[py]`.

1. Detect the position of the P/N sequence within the received signal.
2. Implement the OFDM receiver for the portion of the received signal which comes after the P/N sequence. Note that some samples might miss from the end as well, so you need to keep only the full OFDM blocks.
3. Plot the symbols constellation after the OFDM receiver. You should obtain a noisy 4-QAM constellation.
4. Since the noise is not very high, one can actually decode with no errors the transmitted symbols. Decode one of the OFDM data blocks and use that to estimate the channel coefficients (the lambdas) using the Least Squares (LS) estimator.
5. Plot the absolute values of the estimated lambdas. You should obtain something which is consistent with the fact that $h(t) = \delta(t)$ (the estimated lambdas are affected by noise, of course).

Problem 2. 20 points (Paper and Pencil)

Let $\boldsymbol{x} = (x_0, \dots, x_{N-1})^T$ be a symbol sequence transmitted over a symbol-level channel and let

$$y_n = \sum_{i=0}^{L-1} h_i x_{n-i} + z_n, \quad n = 0, \dots, N+L-2,$$

be the noisy channel output (where $x_n = 0$ if $n \notin \{0, \dots, N-1\}$). All quantities are complex-valued and $N \geq L$.

1. Let $\boldsymbol{y} = (y_0, \dots, y_{N+L-2})^T$, $\boldsymbol{h} = (h_0, \dots, h_{L-1})^T$ and $\boldsymbol{z} = (z_0, \dots, z_{N+L-2})^T$. Specify the matrix C such that
$$\boldsymbol{y} = C\boldsymbol{h} + \boldsymbol{z}.$$
2. Write down the objective function which is being minimized by the least squares (LS) estimate $\hat{\boldsymbol{h}}$ of $\boldsymbol{h}$ based on $\boldsymbol{y}$.
3. The vector $C\hat{\boldsymbol{h}}$ is an element of an inner-product space $\mathcal{V}$. What is $\mathcal{V}$?
4. The projection theorem specifies conditions that need to be satisfied by the LS estimate $\hat{\boldsymbol{h}}$. What are those conditions? Express them in terms of a matrix equation.
5. Find the expression for $\hat{\boldsymbol{h}}$ as a function of C and $\boldsymbol{y}$.
6. Suppose that $\boldsymbol{h} \in \mathbb{C}^L$ is the realization of a Gaussian random vector $\boldsymbol{H} \in \mathbb{C}^L$ which is independent of the noise. Each z_n is a sample from an independent zero-mean Gaussian random variable of variance σ^2. The receiver wants to estimate $\boldsymbol{h}$, using $\boldsymbol{x}$ as a training sequence. (Hence $\boldsymbol{x}$ is known to the receiver.) Derive the expression for the MMSE estimator $\hat{\boldsymbol{h}}_{\text{MMSE}}(\boldsymbol{y})$. Your expression can only depend on the covariances $K_{\boldsymbol{Z}}$ and $K_{\boldsymbol{H}}$, as well as on C.

Problem 3. 20 points (MATLAB/Python)

You are asked to implement and test the LS and MMSE channel estimation derived in the previous problem. The variables $\boldsymbol{x}$, $\boldsymbol{y}$, σ, $K_{\boldsymbol{H}}$ are already loaded for you. Note that you are given the actual channel as well (in $\boldsymbol{h}$), so that you can test your code (e.g. by plotting the absolute value of $\boldsymbol{h}$ and that of the estimate $\hat{\boldsymbol{h}}$).
You might find useful the command `toeplitz(c,r)` (`scipy.linalg.toeplitz(c, r)`) which returns a non-symmetric Toeplitz matrix with `c` as its first column and `r` as its first row. You might also find useful the commands:
`m\n` (for MATLAB) and
`numpy.linalg.lstsq(m, n, rcond=None)[0]` (for Python).
Of course, you need to figure out what to use for the matrices `m` and `n`.
Note that the vectors $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h}$ are given to you as row vectors.
Also, for Python, pay attention to define the data type as `complex` where needed.

Please write your code in the script `p3.m[py]`.

1. Implement the LS channel estimation.
2. Implement the MMSE channel estimation.
3. Plot on the same figure the two estimated channels and the actual one. You should see a quite good agreement between them (the LS estimation should be a bit noisy).