WRITTEN BY: Andrea Manzini
GROUP WITH: Osman Berk Satir

OFFDM Communication System

1. THE PROBLEM STATEMENT

This project consists in building a communication system, using sound in air as mean. We were asked to implement an OFDM system, which is currently employed in most of the commercial implementations.

Differently from a single carrier system, a multi carrier system divides the transmission over the frequency spectrum and assign a very narrow band to each sub-channel. An OFDM symbol typically lasts longer than a single carrier symbol, but it is composed by a multitude of sub-symbols, one for each sub-carrier. This allows to reach quite high data rates.

However, the most important advantage of an OFDM transmission is the possibility of exploiting as well as possible all the available bandwidth. The frequency selective behavior (such as frequency selective fading) of a typical channel heavily distorts a wide band (single carrier) signal, making it very difficult to revert its effect. Timing and phase estimation in time domain can’t deal with this kind of distortion. Instead, channel estimation and equalization can be easily performed in the frequency domain. The narrow band of the sub-carriers allows to treat each sub-channel as an ideal one (flat in band), so, once the channel is estimated, its effect on the symbols can be easily reverted.

In addition, the fact that several symbols can be transmitted at the same time allows to insert guard intervals among symbols without penalizing the data rate too much. This allows to reduce the ISI due to multipath fading. Using a cyclic prefix as guard interval brings even more advantages, making the channel distortion be applied as a (pseudo) circular convolution, thus, making the channel behave more linearly.

In order to estimate the channel, known training symbols can be periodically sent. The introduction of a phase tracking algorithm on each sub-carrier can reduce the number of training symbols needed and then their negative influence on the data rate.

The main problems of this kind of systems are caused by fast changes in the channel (for example moving obstacles), frequency desynchronization and Doppler effect. In particular, this last phenomenon usually shifts all the spectrum, resulting in a complete frequency misalignment and symbols misunderstanding.

In our implementation, great efforts were taken to make the system robust to channel modifications. In particular, considering that QPSK demodulation does not consider the module, we tried to well understand the phase behavior of the channel and we consequently refined the tracking. No measure was adopted to reduce the doppler effect, so the speaker and the microphone must be still during the transmission. However, if they are moved and then released, or if an obstacle is put between them, the system will automatically realign after the closest training symbol, losing only a small part of the transmission.

2. IMPLEMENTATION OF OFDM FOR ACOUSTIC TRANSMISSION

We implemented the OFDM communication system in Matlab. All the signal processing is performed in the digital domain. The audio stream is then managed by Matlab, which interfaces with the DAC and ADC of the computer. Most of the tests were performed using a Bose SLIII as speaker, directly connected to the analog output of the PC and the internal microphone of the PC itself. We made sure that all the software audio enhancements were disabled.

In Figure 1, it is possible to see the general block diagram of an OFDM communication system. We extended this core with other functionalities which are explained in detail in the following sections.

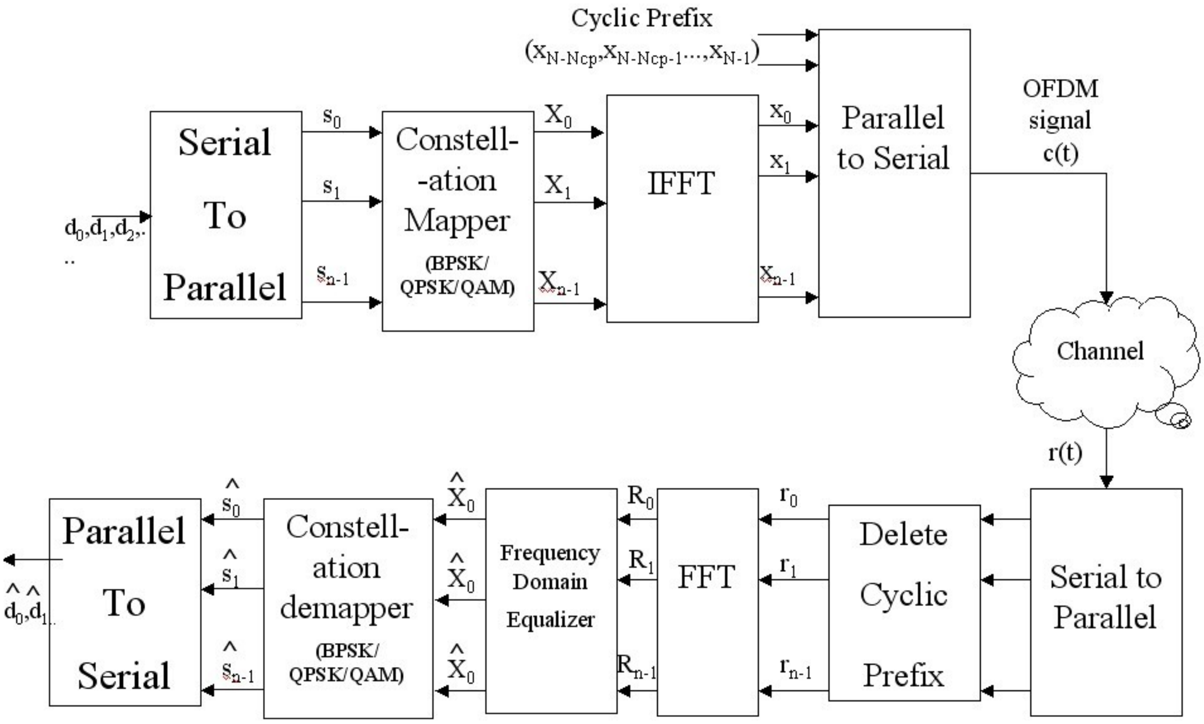


Figure 1. Block Diagram of OFDM

2.1. Bitstream generation

For the first tests, a random bitstream was used, then we implemented the transmission of an image. This, in addition to making the project more enjoyable, let us visually understand where the transmission fails and detect patterns in the errors.

The raw image bits are preprocessed in two steps:

- The data is encoded with a Hamming code (15,11), which can correct 1 bit each 11 of them.
- The encoded data is pseudo-randomized to reduce the PAPR (high peak-to-average power ratio). The randomization is performed doing a xor between the encoded bitstream and a pseudo-random sequence produced by the same LFSR of the preamble.

2.2. Transmitter

- **Mapping**
  Bits are grouped two by two and mapped into the QPSK constellation
- **Training symbols insertion**

One training OFDM symbol[1] is inserted each 32 OFDM symbols. This number strongly influence the system and must be chosen carefully. If the phase tracking works well and the channel is not characterized by very sharp changes, it can be increased. The training symbols are mapped in a BPSK constellation and they are generated by the LFSR. It is important that they are pseudo-random in order to reduce the PAPR and avoid peaks.

- **IFFT**
  The symbols are parallelized in a matrix where the columns are the OFDM symbols, then a IFFT is performed over the columns to go into the time domain. Also the upsampling is done inside the IFFT function. We chose a number of sub-carriers of 1600 aiming to a compromise among bandwidth, data rate and frequency misalignment robustness.
- **Cyclic prefix**
  The cyclic prefix is added in the time domain, copying the last part of each OFDM symbol back to the start. After studying the delay spread of the channel, we chose a prefix length of 200..
- **Preamble**
  - The preamble is generated using the LSFR developed in the previous labs. A BPSK constellation is used.
  - It is scaled accordingly with the maximum power of the OFDM symbols. Without doing this, the system is forced to scale all the other symbols only to be able to produce this much more powerful sequence.
  - It is upsampled to 48000 sample per second.
  - It is pulse shaped with a root raised cosine with 21 taps.
  - It is added in front of the data transmission.
- **Upconversion**
  The signal is shifted around the selected carrier frequency of 6 KHz. Considering that the bandwidth is 4 KHz (8 KHz in total), this value was chosen to let the speaker not use very high tones, which are typically more difficult to produce and hear. The final spectrum goes from 2 to 10 KHz.

2.3. Receiver

Most part of the receiver are not explained because they just reverse what done in the transmitter.

- **Downconversion**
- **Preamble detection**
  A matched filter is applied to all the signal but only to detect the preamble, then the filtered signal is discarded. During this step, the oversampling is exploited to also perform a timing estimation. The frame_sync function, indeed, tries to select the best oversampled tap after exceeding the threshold.
- **FFT**
- **Channel estimation**
  The training symbols are used to estimate the channel in frequency domain. Thanks to the cyclic prefix we can approximate a circular convolution between the signal and the channel, which become a simple constant multiplication in the frequency domain.

[1] As symbol I mean a single QPSK tap, while as OFDM symbol a group of n symbols, where n is the number of subcarriers.

That's why the channel can be simply estimated dividing the received training symbols by the corresponding transmitted ones. Here a low pass filter should be somehow implemented, otherwise a peak of noise could be taken as estimation, making the tracking less efficient. On the other hand, the channel could totally and sharply change over time, making the lowpass filter not beneficial.

- **Phase tracking**
  Starting from the phase estimation with the training symbols, the phase is tracked with a Viterbi-Viterbi algorithm, which become particularly simple in case of a QPSK mapping. Here a low pass filter is implemented, which configuration was carefully tuned. It needs to be as fast as the channel but not as fast as the noise.
- **Channel equalization**
  All the information acquired before on the channel are used here to compensate its effect. I point out that as channel, I mean all the modification applied to the signal from the transmitter to the receiver, including digital/analog conversion, clipping, speaker and microphone frequency response, real channel, lowpass filter of the downconversion and so on.
- **Demapping**

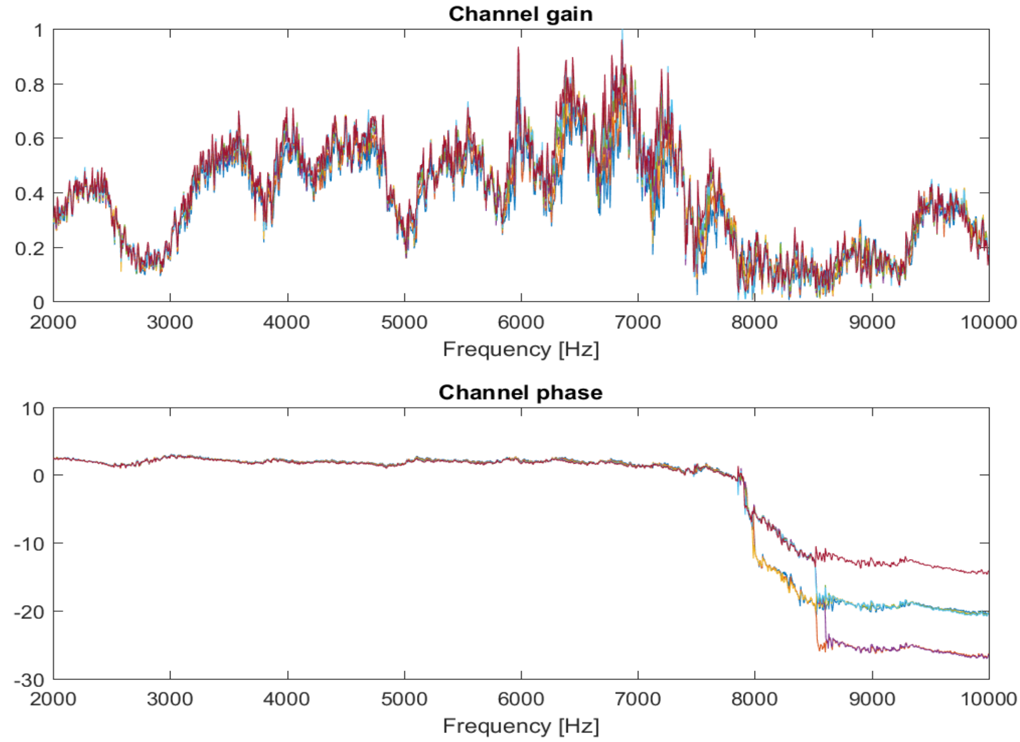
3. RESULTS

Below I report some meaningful plots that are generated when running the code and their analysis. The first group of graphs are the result of a test done in ideal conditions (silence in the room, no object moving, speaker next to the microphone, high volume). Then I present the same graphs under adverse conditions (noisy room, speaker being moved during the transmission and people moving in the room, speaker far from the microphone).

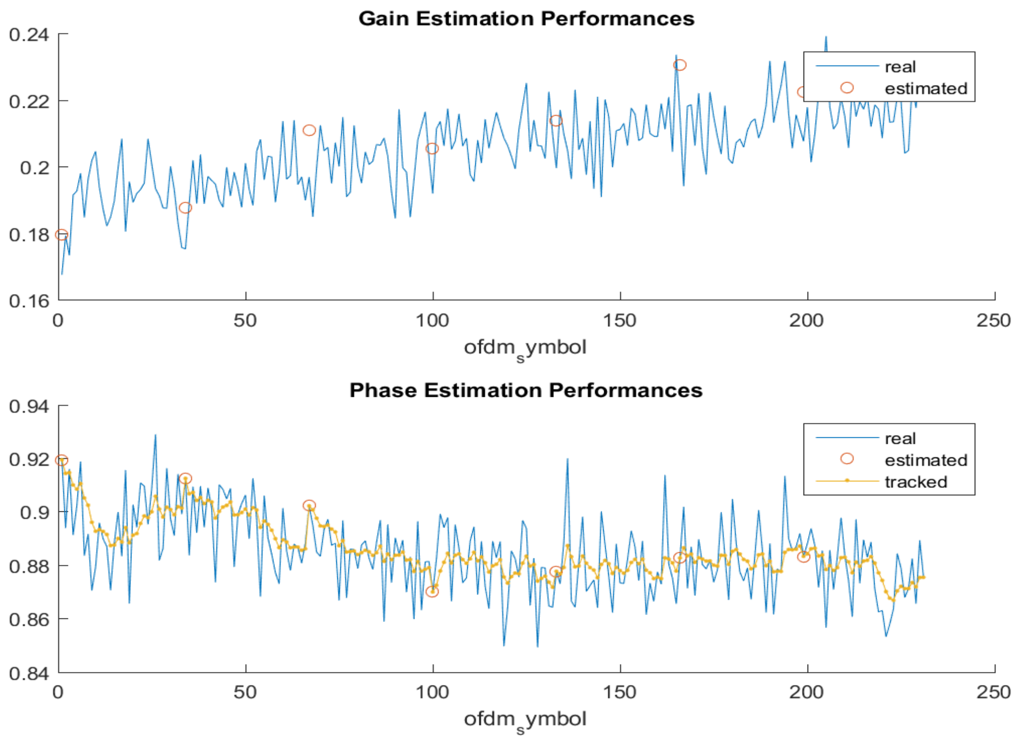
3.1. Ideal Conditions

BER on raw bits	0.0017
BER after hamming decoding	0.0011

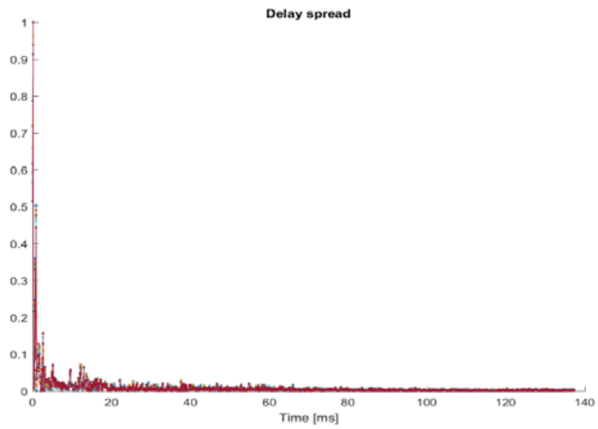
The hamming encoding was tuned to correct all the errors. With a BER of 0.0017 the system makes a mistake each 588 bits and can correct up to 1 bit each 11 bits. So, I expected the encoding to correct all the errors. The fact that the BER slightly decrease, means that the errors are localized and when a mistake occurs, it is followed by many others, in the same Hamming block. This means that the problem here is not the gaussian noise (which would be equally distributed) but still something connected to the channel.



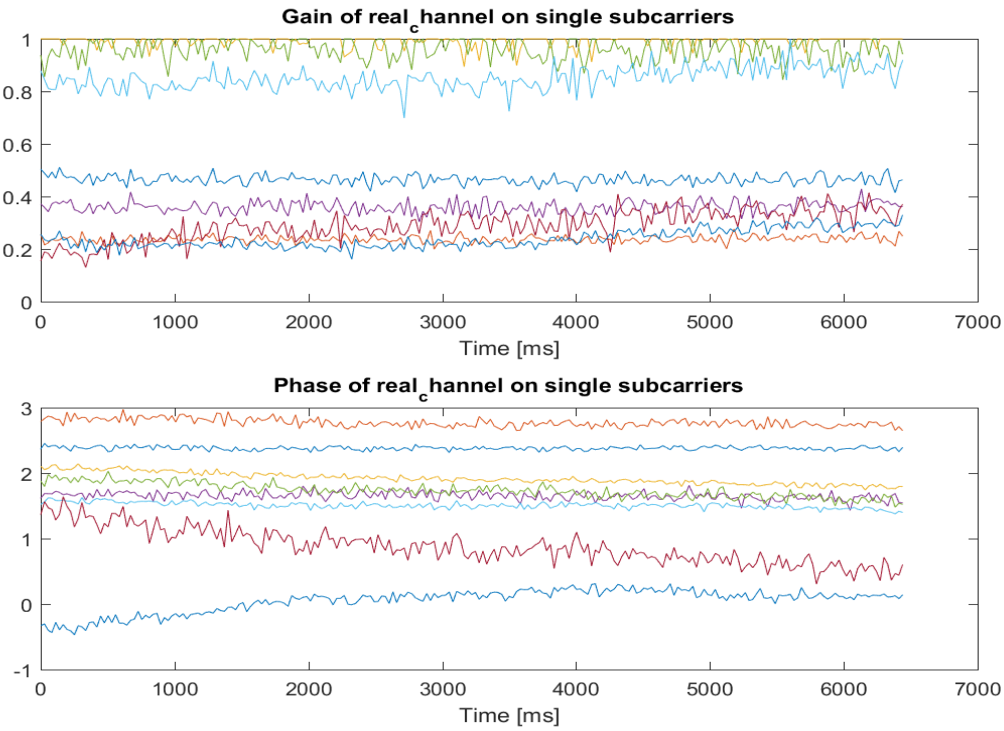
Channel estimation over frequency. Different colors represent different time instants.



Performces of channel estimation on the 1590th subcarrier. The red circles are estimations with training symbols. We can appreciate the phase noise and the very slow noise random walk deviation. In ideal conditions, the phase tracking works very well and we could use only one training symbol. Even without phase tracking, the system works quite well, with a small deviation of the phase over more than 200 ofdm symbols. On the y axis I normalized the phase by pi, so a deviation of 1 means a deviation of pi. The phase tracking is not needed until the phase shift does not exceed pi/4 (0.25 in the graph) and we can see that here this condition is fullfilled for more than 50 seconds (entire transmission).



On the left I present the delay spread of all the training symbols. We can state that after approximately 20 ms the symbol is exhausted. This time corresponds to 160 symbols in our implementation, that's why we chose a length of the cyclic prefix of 200. With this value, the system gains an efficiency of 13.7 Kb/s, with an improvement of the 177% from the default case when the length of the cyclic prefix equaled the number of sub carriers. For the transmission of the entire image the system takes 50 seconds, considering also the preamble and the padding. On the right there is the received image of Lena. Since the BER is low the image is very clear and no pattern is visible.

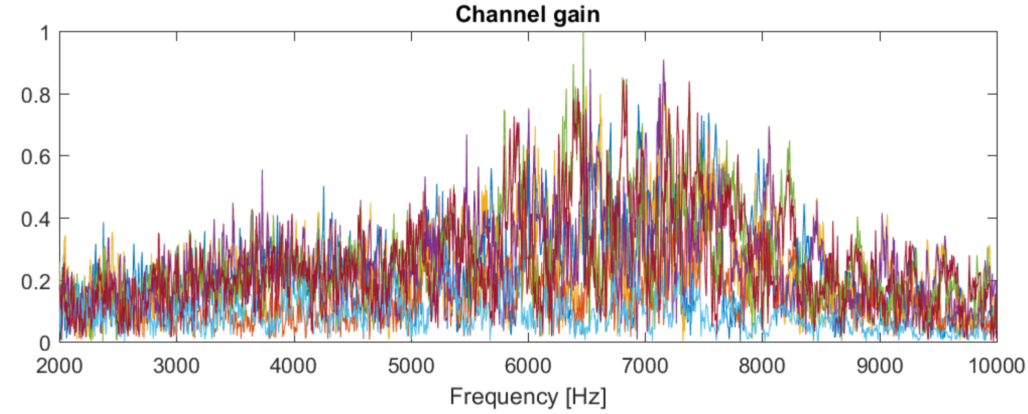


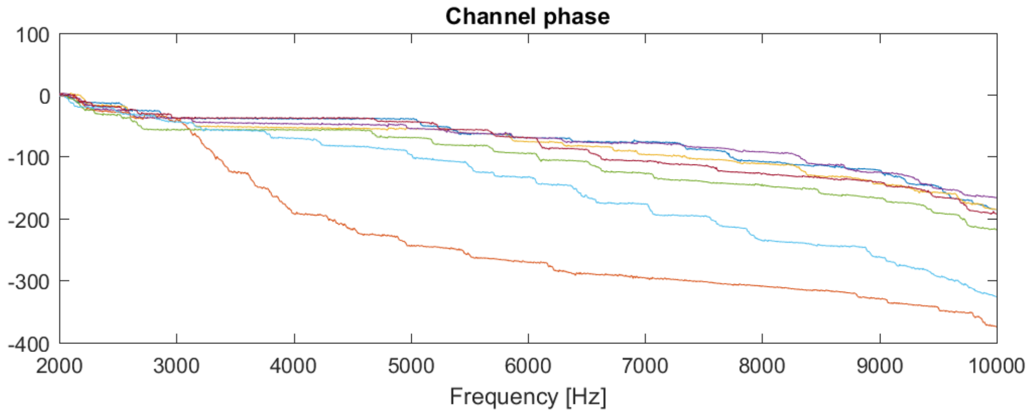
Finally I present the plot of some subcarriers (colors) over time. We can appreciate how in ideal conditions the channel is pretty stable.

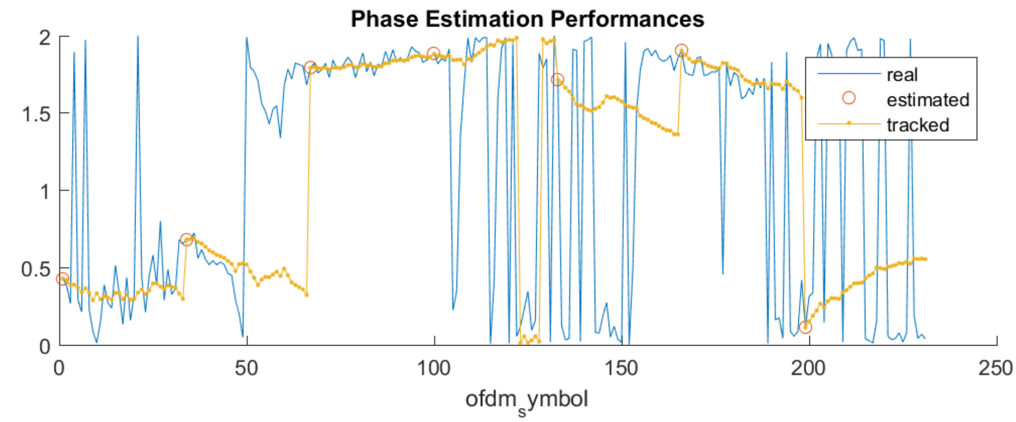
3.2. Adverse conditions

BER on raw bits	0.19
BER after hamming decoding	0.19

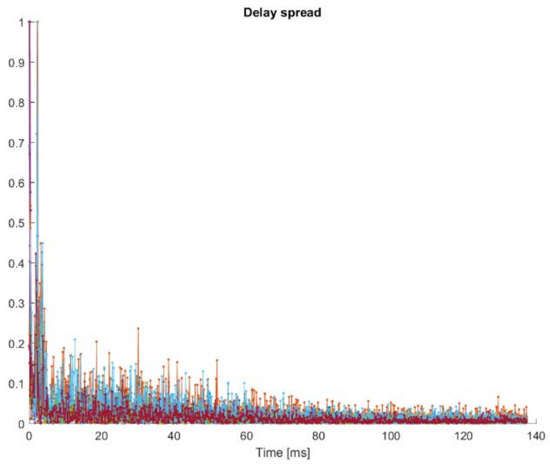
In this case the BER is quite high. The fact that the hamming coding doesn't bring any improvement means that the errors are completely localized, as we can clearly see in the received image.

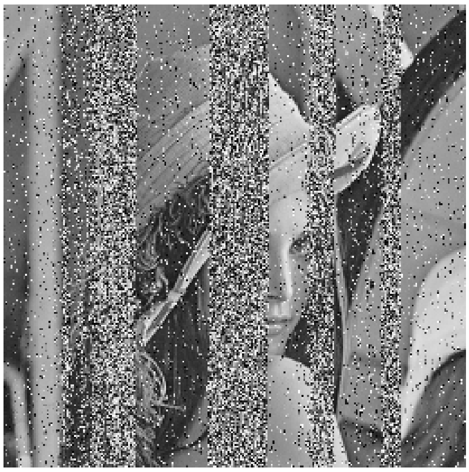




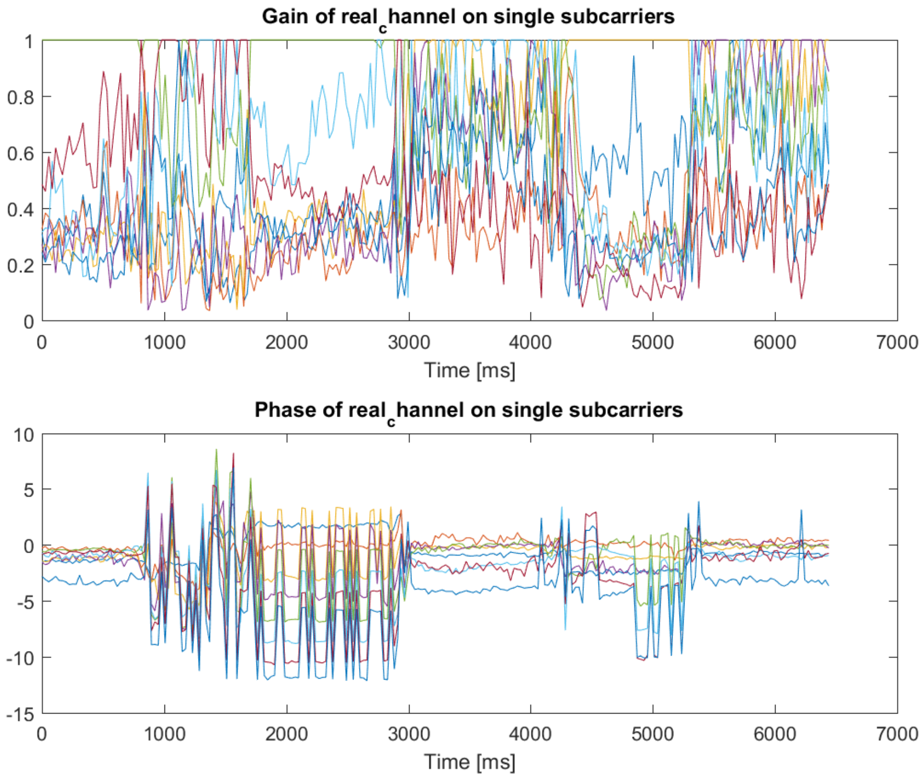


We can see how in such adverse conditions the phase tracking is almost useless because there are very steep changes. Instead, the training symbols can realign the system very well, as happen around the 50th symbol (when the speaker was moved). Better results can be obtained by the phase tracking if an OFDM symbol is much faster than the change of the channel, which could happen in an electromagnetic system, with higher carrier and sampling frequencies. In general, we can conclude that the number of symbols the system can transmit without phase tracking or without sending a new training symbol largely depends on the channel condition and behavior. For example, in this case, it depended on my behavior, moving the speaker or passing in between.





The delay spread is longer, but only for a few training symbols (few colors). The received image is still recognizable, but error stripes are clearly visible. Once the speaker is moved, the phase misaligns and all the following symbols are misunderstood until the next training symbol restores the alignment.



Finally, here we can appreciate how instable is the channel, especially in the first half, when moving the speaker.