ECOLE POLYTECHNIQUE FEDERALE DE LAUSANNE
School of Computer and Communication Sciences

Software-Defined Radio:
A Hands-On Course

Assignment date: Oct 12, 2016
Due date: Oct 19, 2016

OFDM

Goal: The purpose of this lab exercise is to become acquainted with OFDM (Orthogonal Frequency Division Multiplexing). This is an ingenious and currently quite popular technique used to communicate across channels that do not have a flat frequency response (in the band of interest).

Assignment: The goal of this homework assignment is to create an OFDM sender, and a receiver. The sender consists of a function called `OFDM_TX_FRAME` that takes two input vectors. The first input vector, assumed to be known to the receiver, constitutes the preamble that is used for channel estimation. The second vector contains the symbols that make up the message to be transmitted. Both the symbols for the preamble and the data symbols will use 4-QAM modulation. You may assume that the receiver knows perfectly the channel impulse response.

Functions that you may need include `fft`, `ifft`, and those already programmed by you for QAM modulation/demodulation.

Exercise 1. Create the `OFDM_TX_FRAME` function that generates an OFDM frame, according to the following interface:

```
% OFDM_TX_FRAME Generates an OFDM frame
% TX_SIGNAL = OFDM_TX_FRAME(NUM_CARRIERS, NUM_ZEROS, PREFIX_LENGTH,
% PREAMBLE_SYMBOLS, DATA_SYMBOLS)
%   NUM_CARRIERS: number of sub-carriers (power of 2)
%   NUM_ZEROS: number of zero carriers at either end of the spectrum
%   PREFIX_LENGTH: length of cyclic prefix (in number of samples)
%   PREAMBLE_SYMBOLS: vector of symbols for the preamble, used to aid
%       channel estimation (length equal to NUM_CARRIERS - 2*NUM_ZEROS,
%       i.e, the number of useful carriers)
%   DATA_SYMBOLS: vector of symbols to transmit (if the number of data
%       symbols is not a multiple of the number of useful carriers it will
```

```
%       be padded with zeros)
%
%   TX_SIGNAL: The generated OFDM signal, corresponding to one OFDM frame with
% the preamble transmitted during the first OFDM (super)symbol and the data
% transmitted in the subsequent OFDM (super)symbols.
```

Parameter `NUM_ZEROS` specifies the number of carriers that are set to zero on each end of the spectrum. Because of this possibility offered by OFDM to shape the spectrum by independently adjusting the power on each carrier, it is no longer necessary to pass the symbols through a (Nyquist) pulse-shaping filter. The total number of unused carriers is `2*NUM_ZEROS`.

For now the length of the preamble should be equal to the number of useful carriers, so that it fits exactly in the first OFDM symbol of the frame.

Notice that the length of the output vector `TX_SIGNAL` will be a multiple of `NUM_CARRIERS+PREFIX_LENGTH`, its exact value depending on the length of `DATA_SYMBOLS`.

Exercise 2. Create the `OFDM_RX_FRAME` function that implements the receiver for an OFDM transmitter as the one designed in the first exercise. The following header describes the interface of the function:

```
% OFDM_RX_FRAME Receiver for ODFM signals (for AWGN channel)
% RF = OFDM_RX_FRAME(RX_SIGNAL, NUM_CARRIERS, NUM_ZEROS, PREFIX_LENGTH)
%   RX_SIGNAL: vector with the input samples in the time domain
%   NUM_CARRIERS: number of subcarriers (FFT/IFFT size)
%   NUM_ZEROS: number of zero carriers at either end of the spectrum
%   PREFIX_LENGTH: length of cyclic prefix (in number of samples)
%
%   RF: Matrix containing the received OFDM frame. The received symbols are
%   arranged columnwise in a num_carriers x num_ofdm_symbols matrix. After
%   removing the cyclic prefix we go back to frequency domain and also
%   remove the zero carriers.
```

Exercise 3. You can assume that the receiver has perfect knowledge of the impulse response `h` of the channel. Complete the `channel_est1` function that determines the channel coefficients in the frequency domain, `LAMBDA`, from the channel impulse response, `h`. (The channel response is assumed to remain constant during the whole frame.)

In order to test your OFDM implementation, we provide the test file `test_ofdm.m`. This script implements a whole OFDM transmission system. It suffices to complete the above three parts. We also provide solutions of the above three parts in `.p` format so that you could test your solutions individually.

`test_ofdm.m` does other things as well. It implements also other channel estimation methods and compares them.

In order to test your OFDM implementation properly, in `test_ofdm` we provide several alternatives for creating `h`. You can comment/uncomment the appropriate lines to choose different types of channels:

- The line `h = [1; zeros(num_carriers-1, 0)]` implements a channel with no ISI, just AWGN noise (useful for debugging)
- The line `h = 1/sqrt(2) * (randn(...` sets randomly an impulse response with the only constraint that its length is shorter than the cyclic prefix. You can use this as second step to check that your implementation is completely generic and that it works correctly with ISI channels.
- Finally, a third uses the function `create_multipath_channel_filter` to generate the impulse response `h` (at symbol rate) of the discrete multipath fading channel model described in Appendix C of lecture notes.