HW 5: OFDM, BICM, and MIMO

ECE 5680 – Wireless Communication

- **Due date:** Friday, November 5, 2015 at 11:59pm
- **Submission instructions:** Upload your solutions (solutions and Matlab code) to Cornell Blackboard as a *single* `zip`-file; name the file ⟨netid⟩-hw⟨number⟩.zip. Include all simulation plots in either `spdf` or `eps` format in your archive. If you want to get detailed comments on your homework solutions, create a *single* `pdf`-file that contains all answers. I strongly suggest the use of LaTeX to typeset your responses, but a scanned version of your handwritten answers is also OK.
- **Total points:** 100pts

Problem 1: Single-input single-output OFDM with repetition coding (25pts)

In this problem, we consider a simple single-input single-output (SISO) frequency selective fading communication system that uses orthogonal frequency-division multiplexing (OFDM). In practical OFDM systems, one usually deals with the following frequency-domain (FD) input-output relation:

$$y_w = \lambda_w x_w + n_w,\ w = 0, 1, \ldots, W-1, \tag{1}$$

which is obtained after performing OFDM processing at both the transmitter and the receiver. In (1), $y_w \in \mathbb{C}$ is the received FD signal on the wth subcarrier, $\lambda_w \in \mathbb{C}$ is the FD flat-fading channel gain given by the discrete Fourier transform of the time-domain impulse response

$$\lambda_w = \frac{1}{\sqrt{W}} \sum_{\ell=0}^{L-1} h_\ell \exp\left(-\frac{j2\pi}{W}\ell w\right), \tag{2}$$

where we assume a frequency-selective channel with L i.i.d. complex normal time-domain taps $h_\ell \sim \mathcal{CN}(0,1)$, $\ell = 0, 1, \ldots, L-1$. The scalar $x_w \in \mathcal{X}$ in (1) is the transmit symbol on the wth OFDM tone with $\mathcal{X}$ representing the constellation set and $n_w \sim \mathcal{CN}(0, N_0)$ is the FD noise that is i.i.d. across frequencies. In what follows, we assume that the considered OFDM system has 64 subcarriers, i.e., $W = 64$, and a sufficiently long cyclic prefix to avoid inter-symbol interference.

Part 1 Implement a Monte–Carlo MATLAB simulator that generates $2W$ random bits and maps consecutive pairs of bits onto QPSK symbols using the Gray mapping from Problem 1 of Homework 4. Transmit these symbols over the FD channel (1) and perform optimal detection, separately for every subcarrier $w = 0, 1, \ldots, W-1$. Generate the FD channel coefficients as described in (2) with $L = 4$ taps. Plot the SNR (in decibel) versus the average (over tones) bit error rate (BER). What is the diversity of this transmission system (just inspect the simulation results)? *Hint: You can use the fast Fourier transform (FFT) to compute the channel coefficients λ_w; be careful with MATLAB's normalization.*

Part 2 Virtually all practical communication systems use some sort of coding to improve the reliability of data transmission. The simplest (and most likely the worst) form of forward error correction is *repetition coding*. In this problem, we simulate the BER of the above OFDM system with repetition coding. To this end, extend the simulator from Part 1. Generate W random bits. Map consecutive pairs of bits onto QPSK symbols and transmit every QPSK symbol twice over consecutive OFDM subcarriers. The equivalent FD system model should look something like this:

$$\begin{aligned} y_{2w} &= \lambda_{2w}x_w + n_{2w} \\ y_{2w+1} &= \lambda_{2w+1}x_w + n_{2w+1}, \end{aligned}$$

i.e., we transmit the same symbol x_w, $w = 0, 1, \ldots, W/2 - 1$, over neighboring OFDM tones. Rewrite the above equations as a 2×1 SIMO system and perform optimal detection. Simulate the BER performance and generate a BER vs. SNR MATLAB plot. Is the error-rate performance better than that of the system in Part 1? What is the diversity of this transmission system (just inspect at the simulation results)?

Remark: In practical systems that use coding, the transmitted symbols are *interleaved* to improve the error-rate performance. The goal of interleaving is to ensure that replicas of the same symbol are transmitted over *independently* fading channel realizations.

Part 3 Instead of transmitting the same symbol over neighboring subcarriers as in Part 2, transmit the same symbol over subcarriers that are $W/2$ subcarriers apart (this is the maximum one can space all pairs of symbols over the subcarriers). Simulate the resulting BER and compare your result to that of Part 2. Do you observe a performance gain?

Remark: Forward error correction (short FEC) and interleaving are used in virtually all modern wireless communication systems. In practice, however, far better codes than repetition codes are used. The most prominent (and also effective) codes are convolutional codes, low-density parity check (LDPC) codes, or turbo codes. These codes enable superior error protection over repetition coding for the same redundancy (e.g., when transmitting 2 coded bits for 1 information bit).

Problem 2: Bit-interleaved coded modulation (BICM) (25pts)

In Problem 1, we considered a simple system that performs repetition coding and interleaving. Modern wireless systems use a similar concept known as *bit-interleaved coded modulation* or short BICM [1]. This approach is extremely versatile and enables reliable data transmission over a variety of wired and wireless channels (including SISO, MISO, SIMO, and MIMO systems). In this problem, you learn to use the main principles of BICM for the SISO-OFDM system considered in Problem 1.

BICM in a nutshell Fig. 1 illustrates the main building blocks of a wireless system that uses BICM. A data source generates a binary data vector $\mathbf{b} \in \{0, 1\}^B$ consisting of B information bits. Then, a forward error correction (FEC) unit encodes the information bits and generates a coded bit vector $\mathbf{c} \in \{0, 1\}^C$, where $C \geq B$ and $R = B/C$ is the so-called rate of the code (typical code rates are 1/3, 1/2, 2/3, 3/4, and 4/5). The goal of the FEC unit is to add redundancy to the information bits, which enables the receiver to correct errors that happen during data transmission over a variety of communication channels. The interleaver (usually denoted by the symbol $\prod$) permutes the coded bit vector and generates the interleaved vector $\tilde{\mathbf{c}} \in \{0, 1\}^C$. Interleaving ensures that neighboring coded bits are spread out over the entire coded vector.

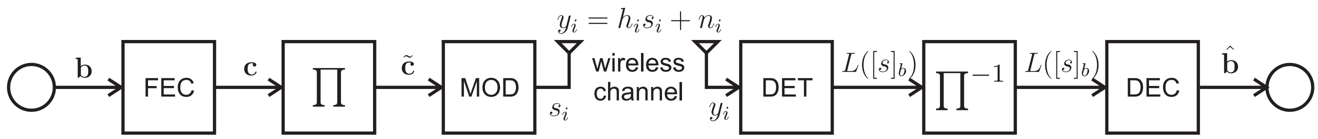

Figure 1: Simple BICM system: A data source generates a binary data vector $\mathbf{b}$. The forward error correction (FEC) unit computes a coded bit vector $\mathbf{c}$ from the data bits $\mathbf{b}$. The interleaver (denoted by $\prod$) permutes the coded bit vector and generates $\tilde{\mathbf{c}}$. The modulator (MOD) maps the coded and interleaved bits onto transmit symbols $s_i \in \mathcal{X}$ and transmits them sequentially over the wireless channel. The receiver performs soft-output detection (DET) using the received signal y_i (and knowledge of the channel coefficients) and computes log-likelihood ratio (LLR) values $L_{\tilde{c}}$ for every coded and interleaved bit. The sequence of LLR values is passed to the de-interleaver (denoted by $\prod^{-1}$). The de-interleaved LLR values L_c are then passed to the decoder (DEC), which computes estimates $\hat{\mathbf{b}}$ for the transmitted bit vector $\mathbf{b}$.

For example, coded bits will not get transmitted over the same fading channel but rather be transmitted over different and independently fading channels. Note that the receiver knows the permutation pattern. The modulator (MOD) then takes $q = \log_2 |\mathcal{X}|$ consecutive bits and maps them onto constellation symbols $s \in \mathcal{X}$ (e.g., using a Gray mapping). In what follows, the notation $[s]_b$ indicates the bth bit associated with the constellation symbol s. All mapped constellation symbols are then transmitted over the channel. The detector (DET) computes so-called log-likelihood ratio (LLR) values (or so-called *soft-information*) for every transmitted coded and interleaved bit. More specifically, for every bit $[s]_b$ associated with the transmit symbol s, the detector computes the following LLR

$$L([s]_b) = \log\left(\frac{\mathbb{P}([s]_b = 1 \,|\, y, h)}{\mathbb{P}([s]_b = 0 \,|\, y, h)}\right), \tag{3}$$

which indicates the probability of bit $[s]_b$ being either a 1 or 0. Large positive values indicate strong confidence that a 1 has been transmitted; large negative values indicate that a 0 has been transmitted; zero indicates that either a 1 or a 0 has been transmitted with probability 1/2 (i.e., the detector is unsure which bit has been transmitted). Note that each LLR value in (3) depends on the received signal y, the channel coefficient h, and the mapping rule (from q bits to constellation symbol s in $\mathcal{X}$). The sequence of C LLR values is then de-interleaved (usually denoted by the symbol $\prod^{-1}$). De-interleaving is simply reverses the effect of the interleaver at the transmit side. As a result, we have LLR values for each of the C transmitted, coded bits. The decoder then uses the sequence of C LLR values to compute estimates for the transmitted information bits, which are subsumed in the vector $\hat{\mathbf{b}} \in \{0,1\}^B$.

BICM system details In this problem, we consider BICM transmission of $B = 64$ information bits over a SISO-OFDM system as considered in Problem 1. We consider repetition coding as forward error correction (FEC), i.e., for each information bit, we copy the bit and insert it to the bit stream to obtain the coded vector $\mathbf{c} \in \{0,1\}^C$, where $C = 2B$. Hence, we use a so-called "rate-1/2 repetition code." Use the same interleaving pattern as in Part 3 Problem 1, i.e., spread neighboring bits as far apart as you can (the maximum spacing is 64 bits). Map every pair of consecutive bits from the permuted vector $\tilde{\mathbf{c}}$ onto QPSK symbols (use a Gray mapping). Transmit the resulting 64 QPSK symbols over the W FD channels (1). Use the same time-domain channel model as in Problem 1 with $L = 4$ taps.

At the receiver, compute LLR values for every coded and interleaved bit. Here, it is important to realize that for every QPSK symbol, two bits are transmitted. Hence, we have to compute separate LLR values for every bit associated with each QPSK constellation point, i.e., for symbol s we have the two bits $[s]_1$ and $[s]_2$. To simplify the notation, define two sets $\mathcal{X}_b^0$ and $\mathcal{X}_b^1$, which contain the subset of QPSK constellation points for which the bth bit is mapped to either a 0 or a 1, respectively. For example, consider the QPSK constellation set $\mathcal{X} = \{-1-j, -1+j, +1-j, +1+j\}$ with bit labels $\{00, 01, 10, 11\}$. The subsets are $\mathcal{X}_1^0 = \{-1-j, -1+j\}$ and $\mathcal{X}_1^1 = \{+1-j, +1+j\}$, as well as $\mathcal{X}_2^0 = \{-1-j, +1-j\}$ and $\mathcal{X}_2^1 = \{-1+j, +1+j\}$. With these subsets, we can now compute the LLR for the first bit $b = 1$ as follows:

$$L([s]_1) = \log\left(\frac{\mathbb{P}([s]_1 = 1 \,|\, y, h)}{\mathbb{P}([s]_1 = 0 \,|\, y, h)}\right) = \log\left(\frac{\sum_{s\in\mathcal{X}_1^1} f(y \,|\, s, h)p(s)}{\sum_{s'\in\mathcal{X}_1^0} f(y \,|\, s', h)p(s')}\right). \tag{4}$$

Once again, $[s]_1$ is the first bit associated to the QPSK constellation point s (according to the used mapping), $f(y|s,h)$ is the conditional probability of observing y, given the symbol s and the channel h, and $p(s)$ is the prior probability of s, which we assume to be equal for all transmitted symbols. Also compute the LLR values for the second bit $L([s]_2)$ using

$$L([s]_2) = \log\left(\frac{\sum_{s\in\mathcal{X}_2^1} f(y \,|\, s, h)p(s)}{\sum_{s'\in\mathcal{X}_2^0} f(y \,|\, s', h)p(s')}\right). \tag{5}$$

After computing all LLR values for all bits in the vector $\tilde{\mathbf{c}}$, perform de-interleaving of all LLR vector. Now, you should have a vector containing LLR values for every coded bit.

For repetition coding, it turns out that the decoder (DEC) performs the following steps. Take the pair of LLR values that belongs to the same information bit; remember that the FEC unit generated two coded bits for every information bit. For example, the first and the second LLR values L_1 and L_2 correspond to the first information bit and the copy of the first information bit. Then, simply sum the LLR values for both bits, i.e., $D = L_1 + L_2$. Finally, generate an estimate for the first information bit as follows: $\hat{b}_1 = 1$ if $D \geq 1$ and $\hat{b}_1 = 0$ if $D < 0$. Repeat this procedure for all remaining information bits.

This decoder rule computes the most likely transmitted bit using the code structure (a repetition code in this case) and the available LLR values. The decoding rule assumes that the received LLR values are independent—the interleaver ensures that this assumption is approximately satisfied.

Part 1 Simplify the LLR expressions in (4) and (5) by using the appropriate probability functions.

Part 2 Write a Monte–Carlo MATLAB script that simulates the average BER for the BICM SISO-OFDM system described above. Compare the BER performance to that of Part 3 Problem 1.

Part 3 Simulate the same BICM system as above, but ignore the interleaver (i.e., do *not* perform interleaving and deinterleaving). Compare your result to that of Part 2. Does interleaving really help?

Problem 3: Comparison of MIMO data detectors (25pts)

In class, we learned the principles of multiple-input multiple-output (MIMO) wireless communication and spatial multiplexing. In this problem, we compare the performance and complexity of the most prominent data-detection schemes. To simplify this problem, you should use the "simple MIMO simulator" that can be found online at: `http://www.csl.cornell.edu/~studer/software_mimo.html`

Part 1 Download the simulator from the above website and read the `README.txt` file. Simulate the BER of a zero-forcing (ZF) detector, minimum mean-square error (MMSE) detector, and ML detector. Use $M_T = 2$ transmit antennas and $M_R = 2$ receive antennas, and select QPSK modulation. Note that these detectors are already built-in. Show a BER vs. SNR plot that compares all three detectors. Do the obtained diversity orders make sense (recall the diversity order of linear and optimal schemes as discussed in class)?

Remark: Have a close look at the simulator. It is supposed to be a good example of how to write a fast simulator that can be parametrized easily and extended with other data detectors.

Part 2 The built-in ML detector in the "simple MIMO simulator" uses a technique called *sphere decoding* [2]. This method performs exact ML detection but requires lower (average) computational complexity than simply evaluating the ML rule:

$$\hat{\mathbf{s}}^{\text{ML}} = \underset{\mathbf{x} \in \mathcal{X}^{M_T}}{\arg\min} \|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 \tag{6}$$

with a search over all possible transmit vectors $\mathcal{X}^{M_T}$. Implement your own ML detector that performs an exhaustive search over all constellation points, i.e., directly solve (6). Simulate the BER for a $M_T = M_R = 4$ system with QPSK transmission with your own ML detector and the built-in ML detector that uses sphere decoding. Extract the simulation time for both schemes for every SNR value (in decibel) and plot the average simulation time versus the SNR. Which ML detector is faster? *Hints: (i) Both detectors should achieve exactly the same BER performance; (ii) use MATLAB's `tic` and `toc` commands to extract the run times.*

Part 3 In class, we briefly discussed the successive interference cancellation (SIC) data detector. Implement this detection algorithm in MATLAB and add it to the simulator. Simulate your implementation for a $M_T = M_R = 4$ system with QPSK transmission, and compare it to ZF, MMSE, and ML detection in terms of BER performance and runtime. Is SIC better than MMSE detection?

Remark: There exists a trade-off between complexity and error rate performance for MIMO data detection algorithms. While ML detection is generally considered to be the best possible, its computational complexity is prohibitive, even in fairly small MIMO systems. Linear detectors enable suboptimal performance, but can be implemented at very low complexity. One generally tries to design a suitable algorithm that achieves both: low complexity and near-ML error-rate performance. The goal of the final project of this class is to implement a variety of existing data detectors and to compare them in terms of performance and complexity.

Problem 4: MIMO beamforming and precoding (25pts)

In class, we learned the principles of beamforming and precoding. The goal of this problem is to compare the error-rate performance of (eigen-)beamforming and ZF precoding. I strongly suggest that you use the "simple MIMO simulator" used in Problem 3 to build the simulator for this problem.

Part 1 Consider the following standard MIMO input-output relation:

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n},$$

where $\mathbf{y}$ is the received vector, $\mathbf{H}$ the MIMO channel matrix, $\mathbf{x} \in \mathcal{X}^{M_T}$ the transmit vector (which we normalize to have unit expected energy, i.e., $\mathbb{E}[\|\mathbf{x}\|_2^2] = 1$), and $\mathbf{n} \sim \mathcal{CN}(\mathbf{0}, \mathbf{I}N_0)$ is additive Gaussian noise. Assume that both the transmitter and receiver know the channel matrix $\mathbf{H}$. The transmitter computes the singular value decomposition (SVD) of the channel matrix $\mathbf{H} = \mathbf{USV}^H$ and transmits $\tilde{\mathbf{x}} = \mathbf{Vx}$. The receiver also computes the SVD and forms $\tilde{\mathbf{y}} = \mathbf{U}^H\mathbf{y}$. The resulting input-output relation corresponds to

$$\tilde{\mathbf{y}} = \mathbf{Sx} + \tilde{\mathbf{n}},$$

where $\mathbf{S}$ is diagonal and contains the singular values of $\mathbf{H}$ on its main diagonal; the resulting noise vector $\tilde{\mathbf{n}} = \mathbf{U}^H\mathbf{n}$ has the same statistics as $\mathbf{n}$. Simulate the BER performance of this so-called eigen-beamforming system. Consider an $M_T = M_R = 4$ system with QPSK transmission. Assume that both the receiver and transmitter perfectly know the channel matrix $\mathbf{H}$. *Hint: Use the fact that one can perform optimal, antenna-wise detection at the receive side—no joint processing is necessary! Be careful with your SNR definitions.*

Part 2 The approach considered in Part 1 only works if the receiver is able to compute $\tilde{\mathbf{y}} = \mathbf{U}^H\mathbf{y}$. In scenarios, where each receiver corresponds to an independent user with one antenna, e.g., y_1 is the received signal at the first user, this processing step is not possible. In such multi-user (MU) MIMO systems, the transmitter typically uses zero-forcing (ZF) precoding, i.e., transmits $\tilde{\mathbf{x}} = \mathbf{H}^+\mathbf{x}$, where $\mathbf{H}^+ = \mathbf{H}^H(\mathbf{HH}^H)^{-1}$ is the right pseudo-inverse of $\mathbf{H}$. By transmitting $\tilde{\mathbf{x}}$, the MIMO input-output relation reduces to

$$\mathbf{y} = \mathbf{x} + \mathbf{n},$$

and it is obvious that now every user can detect its own data independently of the other users. Note that interference from other users has been cancelled perfectly. The main drawback of this approach is that the transmitted signal energy is in general much higher than that of eigen-beamforming. In particular $\|\tilde{\mathbf{x}}\|_2^2 = \|\mathbf{H}^+\mathbf{x}\|_2^2$ can be much larger than transmitting $\mathbf{x}$. Hence, to ensure a fair comparison with the approach in Part 1, we have to normalize the transmit power of ZF precoding. To this end, transmit $\tilde{\mathbf{x}} = \alpha\|\mathbf{H}^+\mathbf{x}\|_2^2$, where α needs to be set so that the instantaneous transmit energy $\|\mathbf{H}^+\mathbf{x}\|_2^2 = 1$. In other words, for every transmission, you need to normalize your transmit vector to unit energy. Consider an $M_T = M_R = 4$ system with QPSK transmission and compare the BER to that of the approach in Part 1. Discuss your results. *Hint: Be careful when normalizing your transmit signals and when defining the SNR.*

Remark: (Eigen-)beamforming is commonly used in point-to-point Wi-Fi systems. Multi-user precoding, as in Part 2, is used in cellular systems, where one has to avoid inter-user interference. The challenge in both scenarios is to acquire accurate channel state information at the transmitter. This can either be achieved by explicitly feeding back channel state information from the receiver to the transmitter (e.g., in data packets) or by using channel reciprocity. This concept assumes that the downlink channel is equivalent (i.e., the adjoint) to the uplink channel.

References

[1] G. Caire. G. Taricco, and E. Biglieri, "Bit-interleaved coded modulation," IEEE Transactions on Information Theory, 1998

[2] A. Burg, *et al.*, "VLSI implementation of MIMO detection using the sphere decoding algorithm," IEEE Journal of Solid-State Circuits, 2005