Advanced Wireless Communications: Algorithms and Architectures **Assignment** 6
Spring Semester 2013 Out: 2013-05-03
Telecommunications Circuits Laboratory Due: 2013-05-10
EPF Lausanne *Revision* : 1.1

Introduction

This exercise has two objectives. The first objective is to give you a better understanding of linear MIMO detectors while implementing three different simple detectors. The second objective is to introduce the simulation environment, that you will use during a small project at the end of the course.
To achieve the first objective you are going to implement the two basic linear receivers that were explained in the lecture. The first of those is the zero forcing (ZF) detector. In a second step you are going to extend your receiver in order to implement the biased MMSE detector. This detector shows slightly better performance than the ZF detector with minimal additional computational complexity. After these two basic linear detectors, you will implement, as a preparation for your project, a successive interference cancellation (SIC) detector.
At the end of this exercise, you will get some intuition about diversity gains of MIMO using different antenna configurations on both sides of the wireless link. Furthermore, you will get some real world questions about MIMO.

Scenario

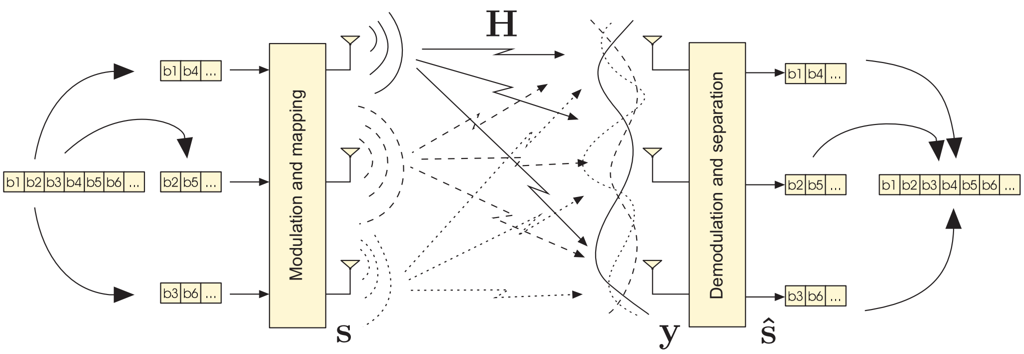

Figure 1: Basic MIMO system model with 3 antennas at the transmitter and at the receiver.

A simple model of the used MIMO system is shown in Fig. 1. At the transmitter the bits are modulated using a QAM modulation alphabet and then concurrently transmitted in the same frequency band over multiple spatial streams. A weighted superposition of these streams are received at each antenna at the receiver. This relationship is represented with the following base-band system equation:

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n}$$

The task of the receiver is to compute an estimate $\tilde{\mathbf{s}}$ based on knowledge about the channel state information given by the channel matrix $\mathbf{H}$ and the noise power, as well as the received symbol $\mathbf{y}$. In the entire exercise we assume perfect channel knowledge at the receiver. For simplicity and in order to speed up the simulations, coding is not used during this exercises. As basic configuration we use at the beginning of the exercise similar as the wireless communication standard IEEE 802.11n 4 transmit and 4 receive antennas.

Framework

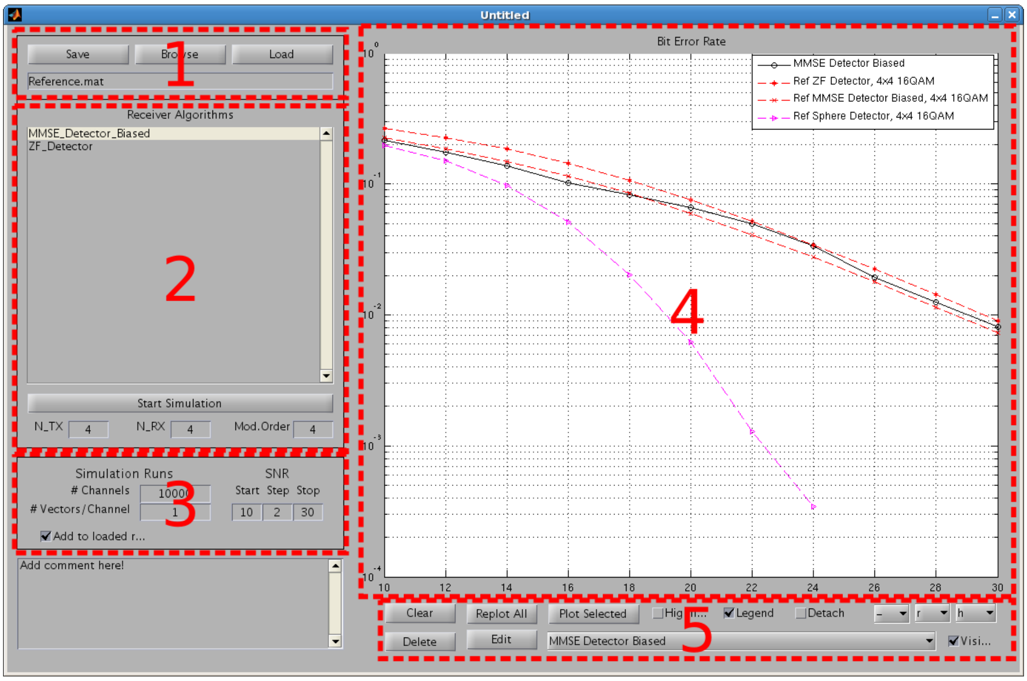

Figure 2: Graphical user interface for MIMO simulations. Three dashed reference curves and one trial curve are shown in the bit error rate plot in section 4 of the GUI.

A new framework is provided to you in this exercise. The framework has a GUI shown in Fig. 2. In the section in the left upper corner marked with 1 you can load and store simulation data or reference curves. Such reference curves are provided for ZF and MMSE detection. You can load them by selecting the file `Reference.mat` and press the button load. In the region below, marked with 2, you can select the detection algorithm you like to simulate. Multiple selections are possible. Furthermore, the antenna configuration and the modulation order is specified in this section. A modulation order of 4 as shown in Fig. 2 corresponds to 16-QAM modulation. In the 3^{rd} section of the GUI you can specify some simulation options, such as the number of channels and the SNR points you would like to simulate. The resulting BER curves of your simulation and of the reference data are shown in section 4 of the GUI. In Fig. 2 the reference curves are shown dashed and a simulation curve with less channels is shown as a straight line. At the bottom of the GUI some display properties are shown, that can be used to highlight some more simulation results.

If you would like to specify a new detector algorithm, you have to create a new file in the `Receivers` folder. All files within this directory are listed as available detectors when you restart the simulation environment. The detector functions share all the same interface given below:

```
function [sHat, complexity] = DETECTORNAME(H, y, ...
                                        Pn_dB, Constellations)
```

The first return value of each detector function is the computed estimate $\hat{\mathbf{s}}$ of the transmitted symbol vector. Each element of the symbol vector is represented as an integer between 0 and $2^{modulation-order} - 1$ representing which constellation point the detector was chosen for the symbol from the given modulation alphabet. Hence the detector does not return demapped bits, but the index of the constellation. The second return value is less important for your implementation. It could be used to compare the complexity of different detection algorithms, especially if the detection algorithm has variable run-time. For this exercise you can just assign a constant value to this return value. The inputs for the function are the channel matrix $\mathbf{H}$, the received symbol $\mathbf{y}$, the noise power in dB, and a complete list of possible constellation points. For further information about the inputs, please use the Matlab debugger.

Exercises

As usual, download the required files for exercise 6 from the course web-page on moodle.

Task 1

You can start the provided framework with the command `NB_Sim_GUI`. Your first task is to implement a ZF detector according to page 18 of the lecture slides. For that reason open the simulation framework and also open the file `Receivers/ZF_Detector.m`. Set a break-point at the beginning of the ZF_Detector function and start the simulation with the ZF_Detector selected in section 2 of the GUI. When the simulation stops at the break-point check the input variables in size and content. For each transmitted item you have to specify the index of the corresponding constellation point minus 1 according to the ZF detection algorithm. `Hint:` We provide some detector reference data curves that you can load by specifying the the file `Reference.mat` in section 1 of the GUI and then pressing the button load.

Task2

Implement now similarly to the ZF detector an MMSE detector and a SIC detector. Having implemented them, compare the performance and the complexity of the detectors. For complexity specify the number of multiplications, divisions, and matrix-inversions required for the computation of the different algorithms. Summarize the results in a small report. `Hint:` You can store the plotted results with the buttons in section 1 of the GUI and reload your simulation data in Matlab using the `load` command.

Task 3

Having compared the different detectors with a specific antenna configuration and one specific modulation scheme, it would now be interesting to know more about the effect of different

configurations. For that reason play with the antenna configuration first. Start with a fixed number of transmit antennas and vary the number of receive antennas. What happens if the number of receive antennas is less than the number of transmit antennas? Why do you see this effect? Is there a limit on the number of receive antennas that you could use in a wireless device such as a cell phone?
In a second plot compare the effect of different modulation schemes and describe the effects you can see.

Hand In Instructions

You can submit your solutions online on to the moodle website of the lecture until 2013-05-10.