ALaRI: Digital Signal Processing

Exercise Session 3: Quadrature Amplitude Modulation (QAM)

Deepaknath Tandur, Paschalis Tsiaflakis, and Toon van Waterschoot[1]

May 2013

Goal: to create m-files for modulating and demodulating a binary signal using the QAM technique, and for evaluating the transmission performance.

Requirements: Matlab in Windows or Linux.

Outcome: 4 m-files (qam_experiment.m, qam_mod.m, qam_demod.m, ber.m).

Exercise 3-1: QAM – modulation, transmission, and demodulation

Create an m-file "qam_experiment.m" in which the following six steps are implemented:

1. Generate a pseudo random binary sequence of a user defined length.
2. Create a Matlab function "qam_mod.m" that modulates a sequence of bits into M-ary quadrature amplitude modulation (QAM) format where M is by definition an integer power of 2, i.e., $M = 2^{N_q}$. To make life easier, your function should only be able to deal with some particular values of M, i.e., $M \in \{4, 16, 64\}$. For the mapping of binary numbers to complex numbers, preferably use the so-called "Gray code", which has the property that neighboring points in the constellation diagram only differ by 1 bit (ask yourself why this could be useful!). Call the qam_mod.m function from within the qam_experiment.m script to modulate the sequence that you have generated in step 1.
3. Check the constellation diagram for the generated QAM symbol sequence. An example of a 16-QAM constellation diagram is shown in Fig. 1.
4. Now add Gaussian white noise (GWN) to the QAM symbol sequence and again check the constellation diagram. Can you explain what you observe?
5. Create a Matlab function "qam_demod.m" that can demodulate a QAM symbol sequence back to a binary sequence. As in step 2, your function should be able to deal with M-ary QAM for $M \in \{4, 16, 64\}$. Call this function from within the qam_experiment.m script to demodulate the noisy QAM symbols back to bits.

[1]For questions and remarks: toon.vanwaterschoot@esat.kuleuven.be

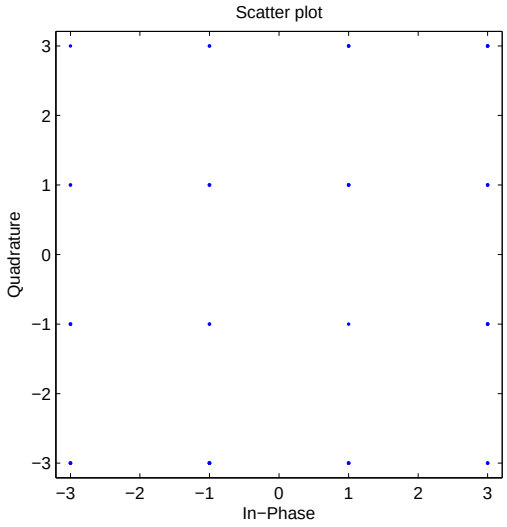

Figure 1: Scatter diagram of 16QAM constellation

6. Finally, create a Matlab function "ber.m" that takes two binary sequences (a transmitted sequence and a received sequence) and calculates the so-called bit error rate (BER). Call this function from within the qam_experiment.m script and check the BER for different M-ary QAM constellations, i.e., different values of M, and different signal-to-noise ratios (SNR).

7. *(Optional)* If you have time left, try to generalize the qam_mod.m and qam_demod.m functions created in step 2 and 4 such that these can deal with arbitrary $M = 2^{N_q},\ N_q \in \mathbb{N}$. *Hint:* splitting up each function in two parts by discriminating between the cases when N_q is even or odd might save you a headache.