Part A

Source Coding with the PCM Method

PCM is a waveform coding method that converts an analog signal into digital data. Typically, the PCM method consists of three main parts: a sampler, a quantizer, and an encoder. The output of the encoder is a sequence of code words (symbols) of fixed length N bits. The purpose of this exercise is to familiarize you with the operation of the quantizer. Specifically, you are asked to implement a uniform and a non-uniform quantizer of N bits, i.e. 2N levels. The quantizers must be implemented as MATLAB functions.

Uniform Quantizer

[xq, centers] = my_quantizer (x, N, min_value, max_value)

x: the input signal in the form of a vector

- N: the number of bits to be used

- max_value: the maximum acceptable value of the input signal

- min_value: the minimum acceptable value of the input signal

- xq: the vector of the output signal encoded with the quantization levels represented by the integers {1, 2, ..., 2N}, where the largest positive quantization level corresponds to the integer '1' (as shown in Figure 1) and the smallest quantization level corresponds to '2N'. These integers can be represented binary with N bits.

- centres: the centres of the quantization regions where the 1st element of the centres vector must correspond to the smallest quantization level - '2N' and the last element to the largest quantization level - '1'.

In particular, the quantizer follows the following steps:

1. Limits the dynamic range of the input signal to the values [min_value max_value], setting samples outside the dynamic range to the corresponding extreme acceptable value.

2. It then calculates the quantization step D and the centers of each region in the centers vector.

3. It finds the region to which each sample of the input signal belongs, and outputs as output the vector xq = [1, 2, ..., 2N].

4. Vector xq can be used as a pointer to the vector centers and get the quantized signal.

An example of the quantization regions for N = 2 bits is shown in the next figure.

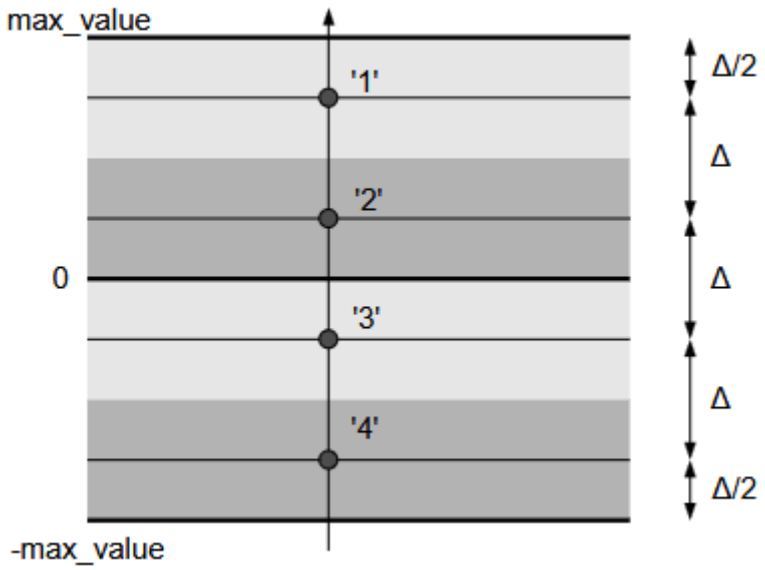

Figure 1: An example of a uniform quantizer layer for N = 2bits.

Non-Uniform Quantizer

For non-uniform quantization of the input vector, the Lloyd-Max algorithm will be used, which allows the design of an optimal quantizer for any number of levels.

[xq, centers, D] = Lloyd_Max(x, N, min_value, max_value)

The inputs are the same as in the case of the uniform quantizer. The outputs have a similar correspondence to the uniform one with an additional parameter D. Specifically, the outputs are as follows:

xq: the encoded vector after Kmax iterations of the algorithm,

- centers: the centers of the quantization regions after Kmax iterations of the algorithm

- D: vector containing the values [D1 D2 ... DKmax] where Di corresponds to the average distortion in iteration i of the algorithm.

Below is a brief description of the algorithm

First you choose a random set of quantization levels:

$$\left\{\tilde{x}_1^{(0)}, \tilde{x}_2^{(0)}, \ldots, \tilde{x}_M^{(0)}\right\}$$

In the exercise, choose these levels to correspond to the centres of the uniform quantizer.

In each iteration i of the Lloyd-Max Algorithm:

1. Calculate the boundaries of the quantum bands, which must be in the middle of the quantum levels, i.e:

$$T_k = (\tilde{x}_k^{(i)} + \tilde{x}_{k+1}^{(i)})/2, \quad 1 \le k \le M-1$$

In every iteration except the last, you must include min_value and max_value as centers in the vector of centers. That is, in every iteration the range of the signal must be [min_value, max_value].

2. Calculate the quantized signal based on these regions and measure the average distortion Di based on the given signal

3. The new quantum levels are the centroids of the bands:

$$\tilde{x}_k^{(i+1)} = E\left[x \middle| T_{k-1} < x < T_k\right]$$

4. Repeat the last 3 steps until:

$$\left|D_i - D_{i-1}\right| < \varepsilon$$

The value of e determines the number of Kmax iterations.

The centres after each iteration will be the same in number. What changes is their location, as the quantization regions will be defined as

[min_value, {centers(1) +centers(2)}/2], [{centers(1)+centers(2)}/2, {centers(2)+ centers(3)}/2] , ... , [{centers(end-1)+centers(end)}/2, max_value]

In case you detect a NaN when running a quantizer, the problem may be in the input and some normalization of the input data is needed.

Audio source

The source you will use is a voice signal that extends up to about 4 KHz. A digital audio signal is given in the form of a waveform (.wav) file which we will consider a satisfactory representation of the corresponding analog signal. The file 'speech.wav', contains samples of a speech signal with sampling rate fs = 8 KHz quantized with N = 16 bits (PCM coding).

To retrieve information about the above audio source, in MATLAB, use the command:

• audioinfo('speech.wav')

To load the signal into MATLAB, use the command:

• [y,fs] =audioread('speech.wav');

and to listen to it, the command

• sound(y,fs);

In case the source value range is larger than the range [min_value, max_value], then you need to normalize the source to this range.

Part A1 questions

1. Implement the PCM scheme using

    a. the uniform quantizer and

    b. the non-uniform one using the Lloyd-Max algorithm.

2. Encode the audio source for min_value = -1, max_value = 1 and N = 2, 4 and 8 bits. Evaluate the PCM scheme based on the following:

    i. Plot for the quantization scheme (b) how the SQNR varies with respect to the number of iterations of the Kmax algorithm (use any value of ε = [10-16, 10-6]). Plot on the same graph the SQNR variation curves for each different value of N so that the comparison can be made using different colors, line type and associated captions provided by the plot() function in Matlab.

    ii. For quantization scheme (b) compare the value of SQNR after Kmax iterations with that obtained using scheme (a) (uniform quantization). Calculate and comment on the performance of the two quantizers.

    iii. Evaluate the result for each quantizer using sound() (acoustic result) and the output waveforms. The comparison is to be made with respect to the original signal (without encoding) and for each value of N. For each N, the waveforms of the original and the encoded signal are to be plotted on the same graph with different colors and line type so that the comparison can be made.

    iv. For each type of quantizer, comment on the efficiency of PCM coding based on the Mean Squared Error (MSE) versus the values of N. Plot on the same graph the MSE curves for the two quantizers with different colors and line types so that the comparison can be made.

Footnote: The final SQNR calculated should be given in dB, i.e. in the form 10log10(.).

Source coding with the DPCM method

Introduction

DPCM (Differential Pulse Code Modulation) coding can be seen as a generalization of Delta coding where the signal that is quantized and sent to the receiver is the difference between the current sample (of time n) and a linear prediction of it. That is, in DPCM coding, we compute, at each time instant, a prediction of the value of the current sample based on the values of previous samples that have already been coded and then compute the error of that prediction. The prediction error signal is then encoded using one or more binary digits per sample.

DPCM encoding

The encoder and decoder of a DPCM system are shown in Figure 2. In order to encode the value of the current sample we first calculate a prediction of the value based on encoded values of previous samples. The prediction of the signal $x(n)$ is denoted as $y\hat{}'(n)$. In Figure we observe a memory device (both at the transmitter and the receiver) that keeps the reconstructed values of the previous samples stored on the basis of which the prediction of the current sample value will be computed. Our goal is to minimize the dispersion of the error signal $y(n) = x(n) - y\hat{}'(n)$ so that it has a small dynamic range and can be satisfactorily described by a small number of binary digits.

The process of quantization of the error signal $y(n)$ leads to the signal $y\hat{}(n)$ which is sent to the receiver. In particular, the quantizer in Figure 2 should limit the dynamic range of the prediction error between a minimum and a maximum value by setting the samples that are outside the dynamic range to the corresponding extreme acceptable value. The quantizer then calculates the quantization step Δ, the centers of each region, calculates which region the input sample belongs to, and outputs the encoded sample $\hat{y}(n)1$ as the output. This sample will be used as a pointer to the vector $centers$ to get the quantized sample as $centers(\hat{y}(n))$. Use the uniform binary digit quantizer N implemented earlier to quantize the prediction error which has a smaller dynamic range compared to the input signal.

The quantizer in particular will quantize each sample of the prediction error separately:

$$\hat{y}(n) = my\_quantizer(y(n), N, min\_value, max\_value)$$

$y(n)$: the current sample of the prediction error as input to the quantizer

max_value: the maximum acceptable value of the prediction error

min_vaalue: the minimum acceptable value of the prediction error

$\hat{y}(n)$: the quantized sample of the current sample of the forecast error

At the receiver, the signal $y\hat{}(n)$ is combined with the signal $y\hat{}'(n)$ (the prediction of $x(n)$). Since the previously reconstructed values as well as the prediction method used by the

transmitter are known at the receiver, it follows that the transmitter and the receiver are able to compute exactly the same values for the prediction $y\hat{}'(n)$. As in the case of delta coding, the transmitter includes as part of the transmitter the receiver device which computes the reconstruction $x\hat{}(n)$. These values are used by the transmitter to compute the prediction, rather than the actual values $x(n)$, in order to fully mimic the receiver device which of course does not know the actual values. By using the reconstructed values to compute the prediction and then the prediction error, we ensure (as in the case of delta coding) that we do not have an accumulation of quantization error.

In the simple case where we rely only on the prediction of the previous sample, the equations that describe the operation of the DPCM system in Figure 2 are as follows:

$$y(n) = x(n) - \hat{y}'(n-1)$$

$$\hat{y}(n) = Q\big(y(n)\big)$$

$$\hat{y}'(n) = \hat{y}(n) + \hat{y}'(n-1)$$

where Q(-) is the input-output function of the graded (uniform) quantizer used. From the above relations we obtain for the quantization error the expression:

$$y_Q(n) = \hat{x}(n) - x(n) = \hat{y}(n) - y(n)$$

We note that if we set $y\hat{}'(n) = 0$ in the above equations, i.e., a DPCM system that does not use prediction, then this system is equivalent to a simple PCM coding system

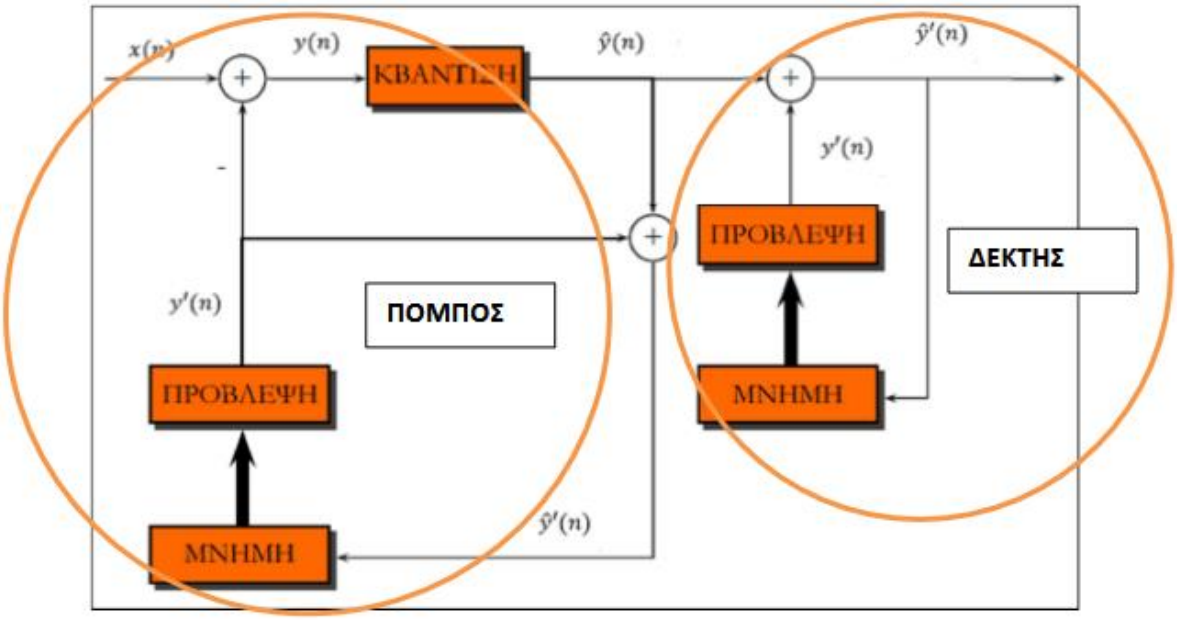

Figure 2. DPCM coding

Calculation of the Prediction Filter

In a general DPCM system, the sample prediction $x(n)$ is given as a linear combination of p previous values y^'(n - i) which have already been encoded and then reconstructed, i.e.

$$\hat{y}'(n) = \sum_{i=1}^{p} a_i \hat{y}'(n-i) = a_i\big(\hat{y}(n-j) + \hat{y}'(n-j-1)\big)$$

Our goal is to compute the coefficients ai with the criterion of minimizing the mean squared error between each current input sample and its prediction:

$$MSE = E[e^2(n)] = E\left[x(n) - \sum_{i=1}^{p} a_i \hat{y}'(n-i)\right]$$

The above criterion, however, is difficult to minimize since MSE depends on the coefficients and the quantizer we use. Therefore, it is a non-linear minimization problem. To get around this difficulty, we replace y^'(n - i) in the above relation with x(n - i) assuming that since the latter is the quantized version of the former we do not make a large error.

Thus, we can minimize the expression:

$$\widehat{MSE} = E[e^2(n)] = E\left[\left(x(n) - \sum_{i=1}^{p} a_i x(n-i)\right)^2\right]$$

$$\widehat{MSE} = E\left[\left(x(n)\right)^2 - 2x(n)\sum_{i=1}^{p} a_i x(n-i) + \left(\sum_{i=1}^{p} a_i x(n-i)\right)^2\right]$$

$$= E\left[\left(x(n)\right)^2\right] - 2E\left[x(n)\sum_{i=1}^{p} a_i x(n-i)\right] + E\left[\left(\sum_{i=1}^{p} a_i x(n-i)\right)^2\right]$$

$$= E\left[\left(x(n)\right)^2\right] - 2\sum_{k=1}^{p} a_i E[x(n)x(n-i)] + E\left[\sum_{i=1}^{p} a_i x(n-i)\sum_{j=1}^{p} a_j x(n-j)\right]$$

$$= E\left[\left(x(n)\right)^2\right] - 2\sum_{k=1}^{p} a_i E[x(n)x(n-i)] + E\left[\sum_{i=1}^{p}\sum_{j=1}^{p} a_i a_j x(n-i)\,x(n-j)\right]$$

$$= E\left[\left(x(n)\right)^2\right] - 2\sum_{k=1}^{p} a_i E[x(n)x(n-i)] + \sum_{i=1}^{p}\sum_{j=1}^{p} a_i a_j E[x(n-i)x(n-j)]$$

$$= R_x(0) - 2\sum_{k=1}^{p} a_i R_x(i) + \sum_{i=1}^{p}\sum_{j=1}^{p} a_i a_j R_x(i-j)$$

By deriving the previous relation of the error with respect to each of the coefficients ai of the prediction filter and setting the derivative equal to zero, we obtain a set of linear equations for the filter coefficients, i.e,

$$\frac{\partial \widehat{MSE}}{\partial a_i} = 0 \Longrightarrow$$

$$\sum_{k=1}^{p} a_i R_x(i-j) = R_x(i), 1 \le i,j \le p$$

$$\boldsymbol{R} * \boldsymbol{a} = \boldsymbol{r} \;\; \acute{\eta} \;\; \boldsymbol{a} = \boldsymbol{R}^{-1} * \boldsymbol{r}$$

$\boldsymbol{R}$: the autocorrelation matrix of dimension $p \times p$ whose (i, j) element is $Rx(i - j)$

$\boldsymbol{r}$: autocorrelation vector of dimension $p \times 1$ whose element i is $Rx(i)$

$\boldsymbol{a}$: dimension vector $p \times 1$ with the coefficients of the prediction filter

Rx: the autocorrelation function of the random process $x(n)$. By solving the above set of Yule-Walker equations, we find the optimal coefficients of the prediction filter. In

addition, for this stationary point to correspond to the minimum of the function it is sufficient that the matrix $\boldsymbol{R}$ is positively definite.

The stochastic quantities appearing in the above expression can be statistically estimated for an input sequence $x(n)$ of dimension $1 \times N$ based on the relations:

$$\hat{R}_x(i) = \frac{1}{N-p} \sum_{n=p+1}^{N} x(n)x(n-i), 1 \leq i \leq p$$

$$\hat{R}_x(i-j) = \frac{1}{N-p} \sum_{n=p+1}^{N} x(n-j)x(n-i), 1 \leq i,j \leq p$$

The prediction filter is calculated at the transmitter, and then the prediction filter coefficients are quantized and sent to the receiver. At this point it is worth noting that the transmitter should also use the quantized values of the prediction filter coefficients so that the transmitter and receiver operate in agreement. To calculate the quantized values of the coefficients, use the uniform quantizer you will construct, setting $\boldsymbol{N} = \boldsymbol{8}$ bits and dynamic range [-2 2].

Input source

The source you are asked to encode/decode is a 20,000-sample signal.The samples of the source you will be experimenting with have good predictability, i.e. a current sample can be predicted (in the statistical sense) with a small prediction error by combining previous values of the same signal. The samples of the source you will experiment with are stored in the file named source.mat. To retrieve the input data it is sufficient to type

load source.mat

Questions - Part A2

In the experiments you will perform, have the dynamic range of the quantizer be between the values max_value = 3.5 , min_value = -3.5 .

1. Implement the above DPCM coding/decoding scheme.

2. Choose two values of $p \geq 5$ and for N = 1,2,3 bits plot the original signal and the prediction error y on the same graph. Comment on the results. What do you observe?

3. Evaluate its performance with a graph showing the mean squared prediction error with respect to N and for different values of p. Specifically, for a number of binary digits N = 1,2,3bits which the uniform quantizer uses to encode the prediction signal and for a

predictor order p = 5:10. In addition, for each p, record in your report and annotate the values of the predictor coefficients.

4. For N = 1,2,3bits plot the original and reconstructed signal at the receiver for p = 5,10 and comment on the reconstruction results with respect to the quantization bits.

Part B

Performance Study of the M-PAM Bandpass Wireless System

In this exercise you are asked to compare the performance of the M-PAM configuration for M=2 and M=8 respectively. This comparison will be based on Bit Error Rate (BER) and Symbol Error Rate (SER) probability measurements performed on a homodyne bandpass system with orthogonal pulse.

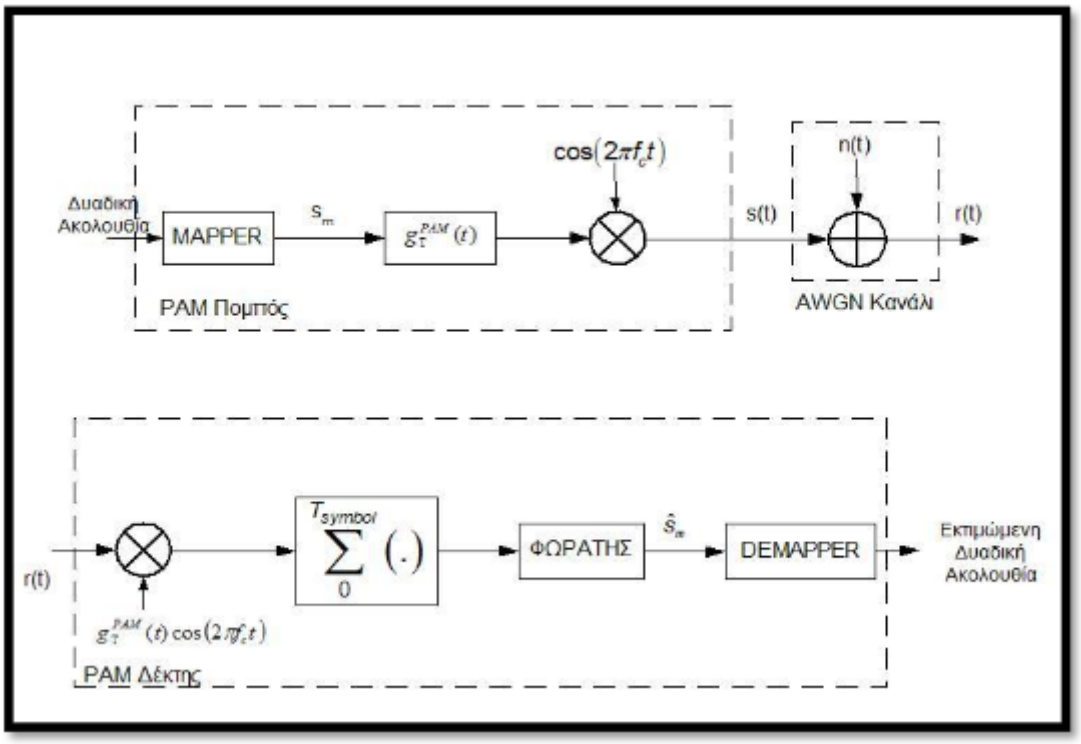

Figure 3. M-PAM Homodyne Live Passing System

Homodyne M-PAM

As shown in Figure 3, the transmitter of the M-PAM system receives as input a binary sequence, converts it into symbols, multiplies it by the orthogonal pulse, and then the signal is transferred to the transmission band through the modulator. AWGN noise is added to the sent signal, and it reaches the system's receiver. There it is demodulated to produce a one-dimensional vector, which is fed into the receiver where it is decided which symbol was sent. Finally, the demapper does the reverse mapping from symbols to bits. The system is described below.

Binary Input Sequence

The input of the system is a sequence of bits, where the values 0 and 1 are displayed isochronously. Such a sequence can be generated if you use any of the functions randsrc, rand, randn appropriately. The number of bits to send should be of the order of Lb = 10000 - 100000 bits.

Matching Bits – Symbols

The mapper is essentially a converter from bits to symbols. Each symbol corresponds to a specific sequence of M binary digits. Therefore, the mapper must for each M binary digits output one of the symbols in the configuration. Similarly, the demapper takes as input the symbol detected by the receiver's decision device (decision device), and outputs the corresponding sequence of M binary digits.

In the case of PSK, PAM, QAM configurations, an important element in this mapping is the Gray coding. According to this, if two symbols are adjacent in two-dimensional or one-dimensional signal space, then they are assigned arrangements of binary digits that differ by only one binary digit from each other.

Rectangular Pulse

The M-PAM system you are asked to simulate uses a rectangular pulse to transmit the symbols which is defined as:

$$g_T^{PAM}(t) = \begin{cases} \sqrt{\frac{2E_s}{T_{symbol}}} = \sqrt{\frac{2}{T_{symbol}}}, 0 \le t \le T_{symbol} \\ 0, \alpha\lambda\lambda o\acute{u} \end{cases} \quad (1)$$

where Es is the energy per symbol, which we normalize as Es = 1, and $Tsymbol$ is the symbol period.

M-PAM configuration

The M-adic PAM waveforms are one-dimensional signals, which can be expressed as

$$s_m(t) = A_m g_T^{PAM}(t) cos(2\pi f_c t), 0 \le t \le T_{symbol} \qquad (2)$$

where Am = (2m - (M + 1))A, m = 1, . . , M. A determines the energy of the symbols. In the case where the PAM signals have different energies (e.g., when M > 2), we want the average energy of the transmitted symbols to be equal to 1.

Simulation Time Units

The system we want to simulate transmits symbols at a rate $Rsymbol$ = 250 Ksymbols/sec so the symbol period is $Tsymbol$ = 4 μsec. In the transmission band, the carrier frequency fc = 2.5 MHz is used, so the carrier period is Tc = 0.4 μsec. Within the simulation, in order to obtain a satisfactory representation of the bandpass signals, a sampling 2 times larger than the Nyquist limit is performed, i.e., we obtain 4 samples per carrier period, and thus the sampling period is $Tsample$ = Tc 4 = 0.1μsec.

AWGN channel

The bandpass signal transmitted by the system transmitter passes through an ideal additive noise channel. The noise is white and follows a $Gaussian$ distribution of zero mean and dispersion $\sigma 2$ = $N02$. The noise can be generated using the $randn$ function.

The dispersion of the noise is determined each time by SNR/bit that we want to have at the system receiver. Recall that due to the normalizations we have done, the energy per symbol is Es = 1, so the energy per bit Eb = Es /Eb = 1 Es /M is Eb = 1 /M. The calculation of the SNR is based on the relation

$$SNR = 10log_{10}\left(\frac{E_b}{N_0}\right) = 10log_{10}\left(\frac{2E_b}{\sigma^2}\right) = 10log_{10}\left(\frac{2}{M\,\sigma^2}\right).$$

Demodulator M-PAM

The demodulator of the M-PAM system correlates (i.e. multiplies and integrates or sums) the received signal with the carrier and the orthogonal pulse. The correlation is performed in the time frame of a symbol period. In the simulation we assume that the M-PAM system is coherent. This means that the receiver knows the phase of the carrier and the time frames of each symbol, i.e. it is fully synchronized with the transmitter.

The demodulator correlates the received signal with the carrier component, resulting in a value r, which is the estimated value of the current symbol on the M-PAM constellation.

Foratis M-PAM

The node takes as input the value r, and decides which symbol (as defined vectorially above) it is closest to. The symbol Am that will have the smallest distance to r, corresponds to the symbol sent.

BER-SER measurements

To measure the BER (Bit Error Rate), i.e. the probability of a bit error occurring, you need to compare the bit value you received with the one you sent. To make reliable BER measurements, they must be derived from a sufficiently large amount of data. A "rule of thumb" is that to measure a BER value of 10^-2 you need 10^4 bits of data, for a BER of 10^-3 you need 105 bits of data, and so on.

BER curves are usually plotted on a logarithmic scale along the y-axis, i.e., in terms of error probability (see, e.g., Fig. 7.57, where some SER (symbol error probability) curves are shown).

To measure the SER (Symbol Error Rate), i.e. the probability of a symbol error, you need to compare the symbol value received with the one sent. Use the same amount of data that you used to calculate the BER.

Questions - Part B

1. Based on the above suggestions, implement the M-PAM system and describe its main points.

2. Measure the bit error probability and plot the BER curve for M = 2 & 8 for simple coding for values of SNR = 0: 2: 20 dB. Repeat the question for M = 8 if the symbols in the mapping were Gray coded. The curves should all be drawn on the same graph.

3. Measure the probability of symbol error and plot the SER curve for M = 2.8 for simple coding for values of SNR = 0: 2: 20 dB. The curves should again all be drawn on the same graph.