Problem 1 – Minimal orthonormalization (from prof. Cioffi) (10pts)

Each column of A given below is a data symbol that is used to construct its corresponding modulated waveform from the set of orthonormal basis functions $\{\phi_1(t), \phi_2(t), ..., \phi_6(t)\}$. The set of modulated waveforms described by the columns of A can be represented with a smaller number of basis functions.

$$A = [a_0\ a_1\ ...\ a_7] = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 3 & 5 & 7 & 9 & 11 & 13 & 15 \\ 2 & 4 & 6 & 8 & 10 & 12 & 14 & 16 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 2 & 0 & 2 & 0 & 2 & 0 & 2 \\ 2 & 4 & 2 & 4 & 2 & 4 & 2 & 4 \end{bmatrix}$$

The transmitted signals $a_i(t)$ are represented (with a superscript of * meaning matrix or vector transpose) as

$$a_i(t) = a_i^* \begin{bmatrix} \phi_1(t) \\ \phi_2(t) \\ \vdots \\ \phi_6(t) \end{bmatrix}$$

$$A(t) = A^*\phi(t)$$

Thus, each row of $A(t)$ is a possible transmitted signal.

1) Use MATLAB to find an orthonormal basis for the columns of A. Record the matrix of basis vectors. The MATLAB commands **help** and **orth** will be useful. In particular, if one executes Q = orth(A) in MATLAB, a 6 × 3 orthogonal matrix Q is produced such that $Q^*Q = I$ and $A^* = [A^*Q]Q^*$. The columns of Q can be thought of as a new basis – thus try writing $A(t)$ and interpreting to get a new set of basis functions and description of the 7 possible transmit waveforms. Note that **help orth** will give a summary of the **orth** command. To enter the matrix B in MATLAB (for example) shown below, simply type B=[1 2; 3 4];

$$B = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

Solution:

$$\text{Q = orth(A)} = \begin{bmatrix} \text{-3.3941e-01} & \text{3.2336e-02} & \text{-1.1243e-01} \\ \text{-6.1902e-01} & \text{2.2144e-01} & \text{3.5120e-01} \\ \text{-6.7883e-01} & \text{6.4671e-02} & \text{-2.2486e-01} \\ \text{-3.3837e-02} & \text{-2.4499e-01} & \text{3.4793e-01} \\ \text{-6.7675e-02} & \text{-4.8998e-01} & \text{6.9586e-01} \\ \text{-1.8730e-01} & \text{-8.0351e-01} & \text{-4.5626e-01} \end{bmatrix}$$

$$A(t) = A^*\phi(t) = A^*QQ^*\phi(t) = (A^*Q)Q^*\phi(t) = \hat{A}^*\phi_Q(t)$$

2) How many basis functions are actually needed to represent our signal set? What are the new basis functions in terms of $\{\phi_1(t), \phi_2(t), ..., \phi_6(t)\}$?

rank(Q) is three. Therefore we need three basis functions to represent the signal set. The new basis functions are: $\phi_Q(t) = Q^* \phi(t)$

3) Find the new matrix $\hat{A}$ which gives the data symbol representation for the original modulated waveforms using the smaller set of basis functions found in (2). $\hat{A}$ will have 8 columns, one for each data symbol. The number of rows in $\hat{A}$ will be the number of basis functions you found in (2).

$$\hat{A} = \begin{bmatrix} -2.6907e+00 & -6.1696e+00 & -8.5609e+00 & -1.2040e+01 & -1.4431e+01 & -1.7910e+01 & -2.0301e+01 & -2.3780e+01 \\ -1.2239e+00 & -3.4513e+00 & -1.4804e-02 & -2.2422e+00 & 1.1943e+00 & -1.0331e+00 & 2.4034e+00 & 1.7598e-01 \\ -1.1235e+00 & -1.5609e-01 & -8.4300e-01 & 1.2439e-01 & -5.6252e-01 & 4.0486e-01 & -2.8204e-01 & 6.8534e-01 \end{bmatrix}$$

Problem 2 - Equalization (50pts)

In this problem, you will get to implement several equalization techniques using finite length filters and characterize the performance of these different approaches.

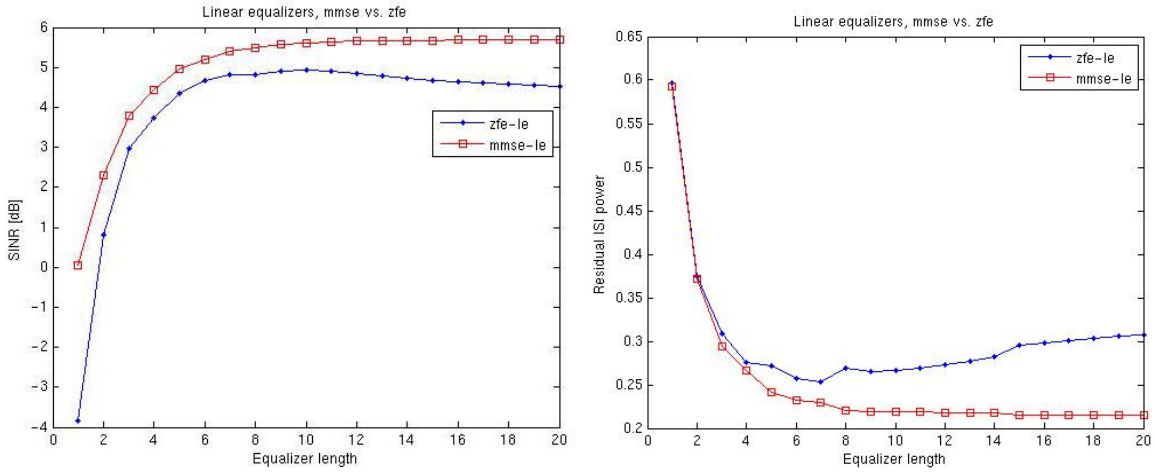
1) Zero-forcing linear equalization (10pts)

The basic zero-forcing linear equalization algorithm is implemented for $0.9D^{-1}+1$ channel in the script **eq.m**. Modify this script to investigate the equalizer performance (variance of residual ISI, and SNR) for different equalizer lengths (1-20). Make sure that for each equalizer length you find the optimal delay (marked delta in the equalization script). Explain the results.

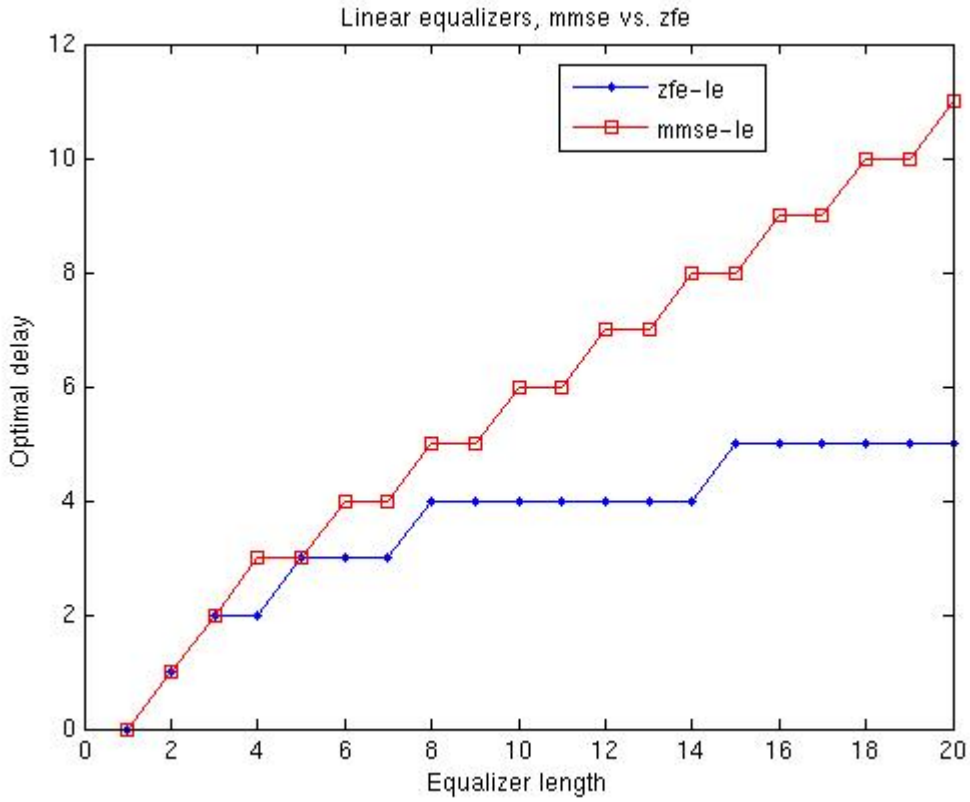
2) Minimum-mean square error linear equalization (15 pts)

Modify the script from 1) to compute the equalizer coefficients that minimize the mean-square error (i.e. both ISI and noise). Hint: Look at the lecture notes or the Chapter 3 of the reader. It is only a small modification to the ZFE formula.

For MMSE equalizer, plot the variance of residual ISI, and SNR vs. equalizer length. Again, make sure that you find the optimum delay for each equalizer length (1-20). Explain the results.

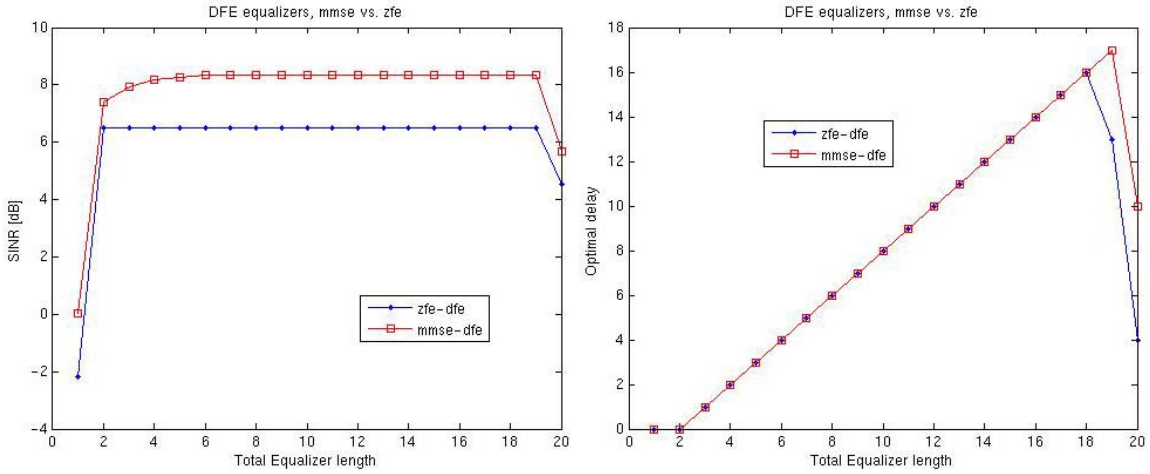


As expected, we see that the SINR or mmse algorithm is better than that of the zero-forcing algorithm (since mmse limits the noise amplification). Zero-forcing algorithm makes use of longer equalizers to remove the ISI, but enhances the noise so that the net SINR drops after an optimal value of 10 taps.

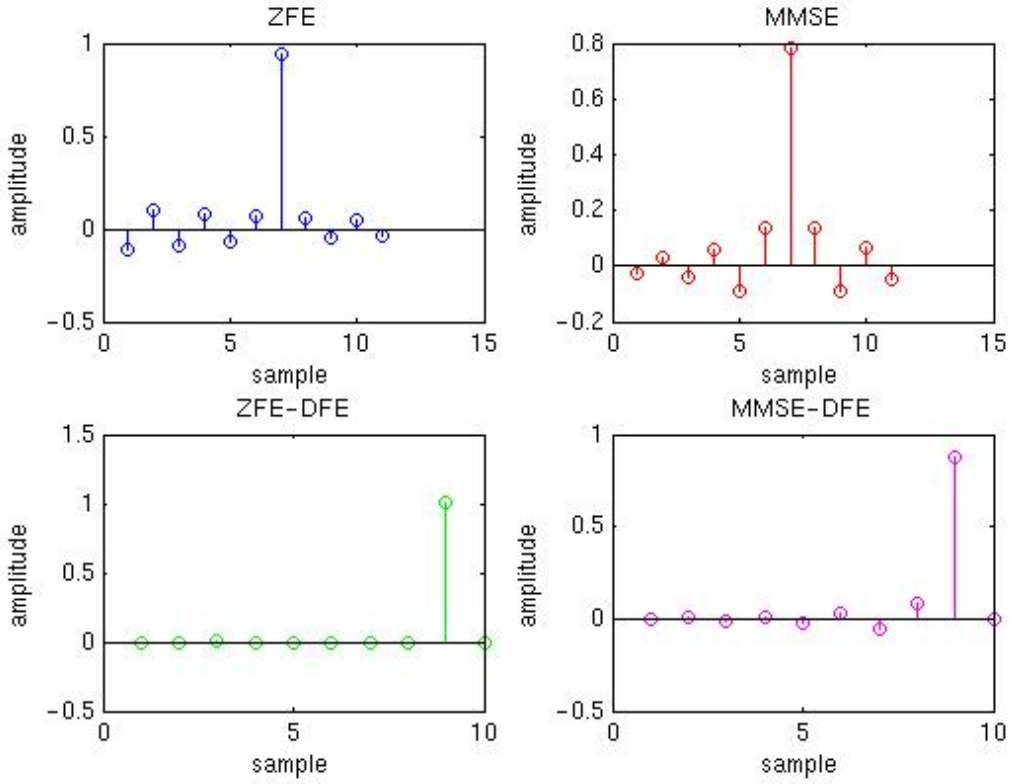


For more details on the computations, see **ps1_problem2_soln.m** script.

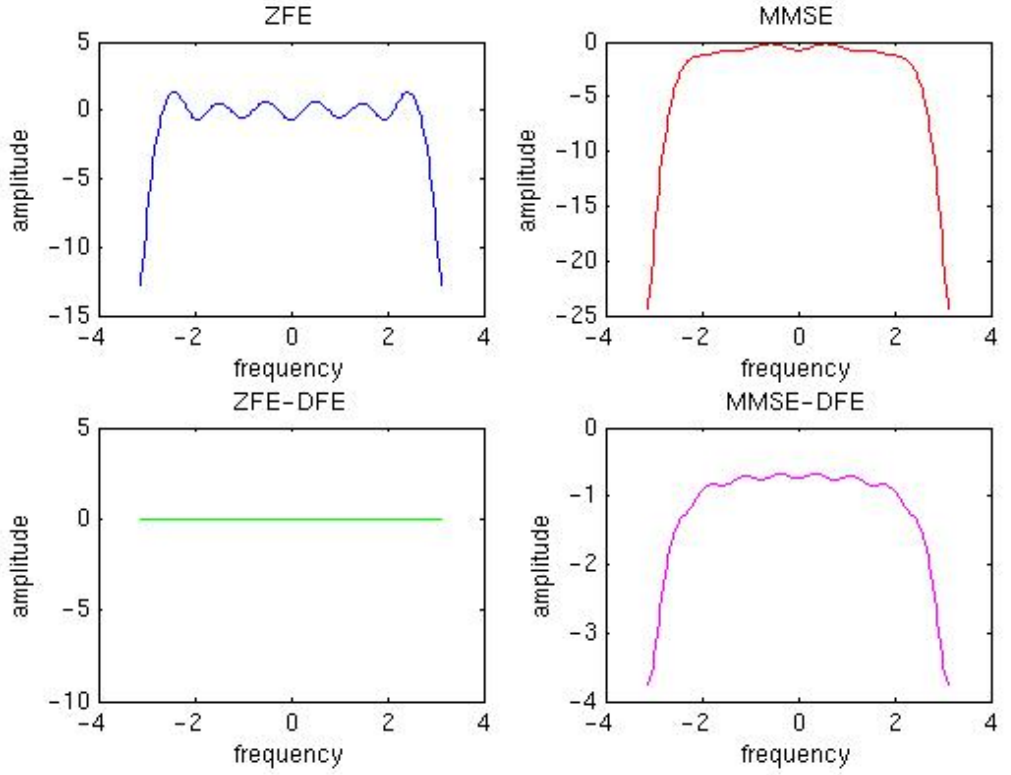
3) Decision-feedback equalization (25 pts)
By puncturing certain columns of the channel matrix P, implement the decision-feedback equalizer. Find feedforward equalizer coefficients for both ZFE and MMSE cases. Also find feedback equalizer coefficients. Plot the residual ISI and SNR vs. the number of feedback and feed-forward taps that you used (adjust the delay delta for optimal performance). The total number of taps should be 1-20.



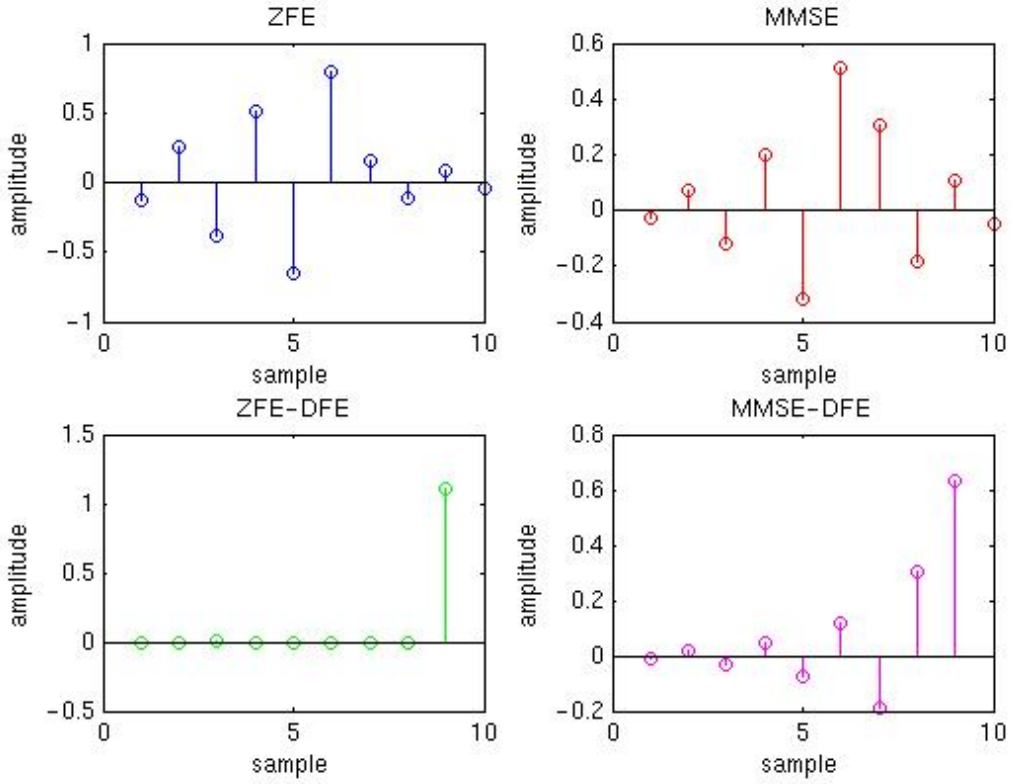
Pick a 10 tap total equalizer length and plot the channel response, equalized pulse response, and equalizer response in time and frequency domain, for ZFE and MMSE linear equalizers as well as ZFE and MMSE decision-feedback equalizers and explain the differences.



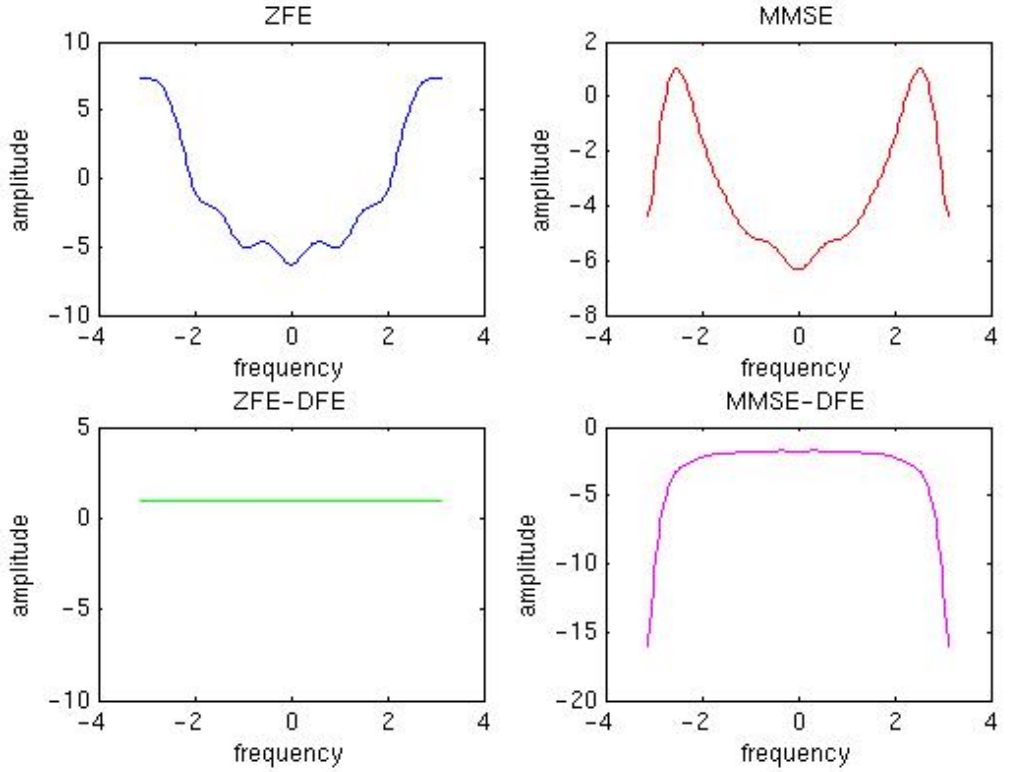
Equalized pulse response – time domain.



Equalized pulse – frequency domain.



Linear and feedforward equalizer response – time domain.



Linear and feed-forward equalizer response – frequency domain.

Problem 3 - Water-filling (40pts)

Our example channel is $0.9D^{-1}+1$. The script **waterfill.m** contains basic channel and noise information.

1) Capacity (20pts)

Design the waterfilling rate-adaptive algorithm (based on the iterative algorithm discussed in class). Find the capacity of the channel.

Solution: (waterfilling code is shown in **waterfill_soln.m** function and by running **ps1_problem3_soln.m**, for this part of the problem we get (by setting the gap to 0dB):

```
c_bar = 1.5541e+00 % total channel capacity per dimension
cn_bar = 9.6934e-01   1.8456e+00   2.2297e+00   2.3436e+00   2.2297e+00   1.8456e+00   9.6934e-01   0
en_bar = 9.5470e-01   1.1916e+00   1.2329e+00   1.2415e+00   1.2329e+00   1.1916e+00   9.5470e-01   0
Nstar =  7
gn =   2.9680e+00   1.0000e+01   1.7032e+01   1.9945e+01   1.7032e+01  1.0000e+01 2.9680e+00 5.5249e-02
```

2) QAM (10pts)

Assuming a $Pe=10^{-4}$, find the Gap for the QAM modulation and find the achievable bit rate for this channel using the algorithm in 1) modified to include the Gap for QAM at $Pe=10^{-4}$.

Solution: (waterfilling code is shown in **waterfill_soln.m** function and by running **ps1_problem3_soln.m**, for this part of the problem we get (by setting the gap to 6.637dB).

```
b_bar_qam =   8.0894e-01
bn_bar_qam =   1.1774e-01   9.9396e-01   1.3781e+00   1.4920e+00   1.3781e+00   9.9396e-01   1.1774e-01          0
en_bar_qam =   2.7540e-01   1.3677e+00   1.5581e+00   1.5976e+00   1.5581e+00   1.3677e+00   2.7540e-01          0
Nstar_qam =    7
gn_qam =   2.9680e+00   1.0000e+01   1.7032e+01   1.9945e+01   1.7032e+01   1.0000e+01   2.9680e+00   5.5249e-02
```

The gap for QAM can be calculated from following equations:

$$\overline{Ex} = d^2 \frac{M-1}{12};\ d \simeq 2\sigma Q^{-1}(P_e);\quad \Gamma_{QAM} = \frac{1}{3}\left(Q^{-1}(Pe)\right)^2$$

$$(\bar{b} = \frac{1}{2}\log_2(1+(M-1)) = \frac{1}{2}\log_2\left(1+\frac{12\overline{E}_x}{d^2}\right) = \frac{1}{2}\log_2\left(1+\frac{SNR}{\frac{1}{3}\left(Q^{-1}(Pe)\right)^2}\right) = \frac{1}{2}\log_2\left(1+\frac{SNR}{\Gamma_{QAM}}\right))$$

3) PSK (10pts)

Find the Gap for the PSK modulation at $Pe=10^{-4}$ and calculate the bit rate.

$$d = 2\sqrt{E_x}\sin\left(\frac{\pi}{M}\right) \approx 2\sqrt{E_x}\frac{\pi}{M};\ d \approx 2\sigma Q^{-1}(P_e);\quad \Gamma_{PSK} = \left(\frac{Q^{-1}(Pe)}{\pi}\right)^2$$

$$\bar{b} = \frac{1}{2}\log_2(M) = \frac{1}{2}\log_2\left(\frac{2\pi\sqrt{E_x}}{d}\right) = \frac{1}{4}\log_2\left(\frac{4\pi^2 E_x}{4\sigma^2\left(Q^{-1}(Pe)\right)^2}\right) \approx \frac{1}{4}\log_2\left(1+\frac{SNR}{\left(\frac{Q^{-1}(Pe)}{\pi}\right)^2}\right) = \frac{1}{4}\log_2\left(1+\frac{SNR}{\Gamma_{PSK}}\right)$$

Plot the energy and bit distributions for 1-3. Try increasing the number of dimensions (decrease tone spacing). Plot the capacity, bit_{QAM}, bit_{PSK} vs. number of dimensions. Explain the plots.

```
b_bar_psk =   6.8481e-01
bn_bar_psk =   3.7926e-01   8.1737e-01   1.0094e+00   1.0664e+00   1.0094e+00   8.1737e-01   3.7926e-01          0
en_bar_psk =   8.7918e-01   1.2112e+00   1.2691e+00   1.2811e+00   1.2691e+00   1.2112e+00   8.7918e-01          0
Nstar_psk =    7
gn_psk =   2.9680e+00   1.0000e+01   1.7032e+01   1.9945e+01   1.7032e+01   1.0000e+01   2.9680e+00   5.5249e-02
```

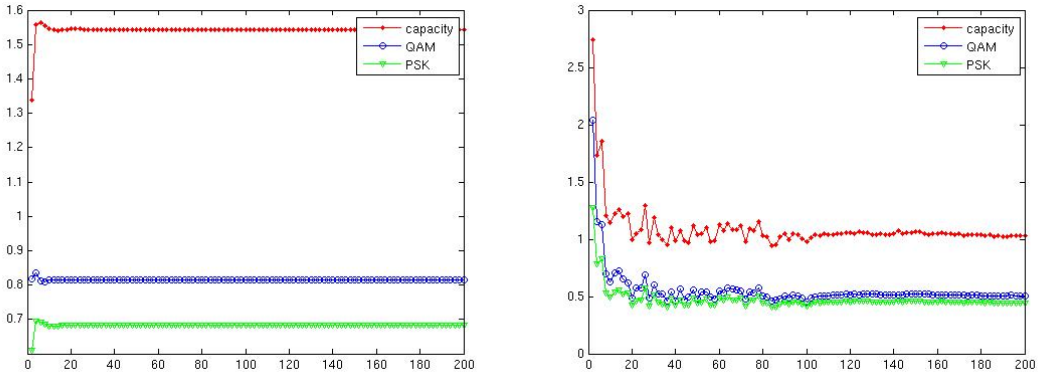


Fig.3.1. Average number of bits per dimension for a) $0.9D^{-1}+1$ channel, b) randomly distributed 100-tap multipath channel

Since the channel in Fig. 3.1 a) is pretty smooth, not many dimensions (>4) are needed to accurately estimate the capacity of the channel. On the other hand, in a heavily non-line of site (multi-path) channel significant notching occurs in frequency domain hence many points (dimensions) are needed to accurately estimate the channel capacity.