The Q Function and Baseband Data Communication

Eren Can Gungor
Riccardo Miccini
Technical University of Denmark - DTU

December 13, 2016

Contents

1 Eye Diagram for a Digital Communication Channel

1.1 Eye diagram

1.2 `c5ce2.m`: explanation

Here follows a thoroughly commented version of the provided `c5ce2.m` MATLAB script. The code below generates and plots the eye diagrams of four band-limited signals composed of random sequences of bits.

Listing 1: ../scripts/1/c5ce2.m

```
% clean figure and load signal package (only for Octave)
clf
pkg load signal

% simulation parameters:
%  - nr of symbols (must be divisible by 4)
%  - nr of samples per symbol
%  - filter cutoff values (normalized values)
nsym = 100;
nsamp = 50;
bw = [0.4 0.6 1 2];

% for each filter..
for k = 1:length(bw)
  % generate filter coefficients
  lambda = bw(k);
  [b,a] = butter(3,2*lambda/nsamp);

  l = nsym*nsamp;

  % Total sequence length
  y = zeros(1,l-nsamp+1);

  % Initalize random output vector with +1 and -1
  x = 2*round(rand(1,nsym))-1;

  % for each overlap..
  for i = 1:nsym
    % place symbols into vector y
    kk = (i-1)*nsamp+1;
    y(kk) = x(i);
  end
  % zero-order hold
  datavector = conv(y,ones(1,nsamp));
```

```
  % apply filter to complite sequence
  filtout = filter(b, a, datavector);

  % splice sequence into sub-sequences of 4 symbols
  datamatrix = reshape(filtout, 4*nsamp, nsym/4);

  % discart the first 6 sub-sequences
  datamatrix1 = datamatrix(:, 6:(nsym/4));

  % plot and format
  subplot(length(bw),1,k)
  plot(datamatrix1, 'k')
  ylabel('Amplitude')
  axis([0 200 -1.4 1.4])
  legend(['Bn = ', num2str(lambda)])
  if k == 4
    xlabel('t/Tsamp')
  end
end
```

1.3 Channel model

1.4 `c5ce2.m`: different bandwidths

1.5 `c5ce2.m`: plots

This section will elaborate on the results and implications of the plots generated by the two scripts.

2 The Q function

2.1 Normal probability density function 2.1

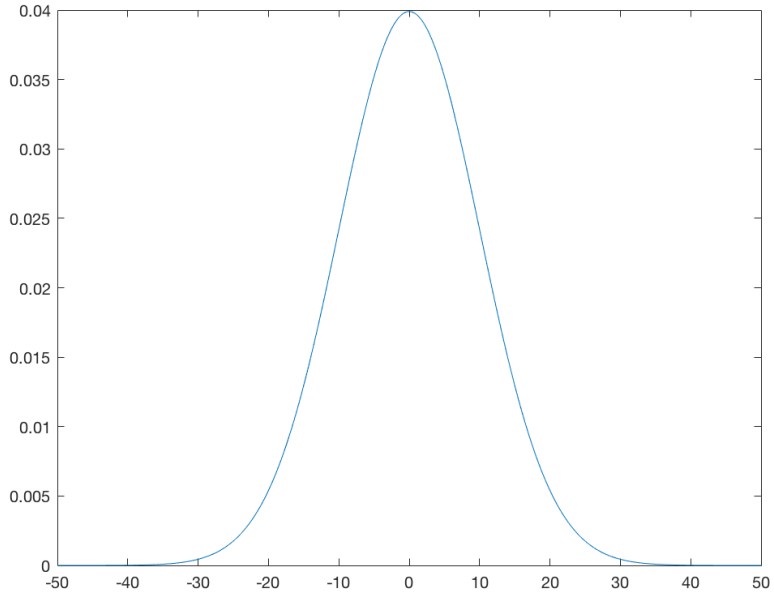

Figure 1: Normal Gaussian pdf graph with defined intervals

Normal gaussian distribution is one of the most important concept in Communication system and in statistics. It is used as a powerful tool when investigating random signals in communication systems. Such as investigating behaviour and application of noise signals.

The other reason is that because it is characterised by the limited variables. Such as mean μ and σ. It is easier compute and understand when we apply on communication systems.

All the variables that has been used in graphingpdf.m has been explained below;

mu . This is the mean value (μ) for the normal probability density function.

sigma This the sensible standard deviation number. (σ).

MAX 50; Maximum x value that x vector will get

MIN -50; Minimum x value that x vector will get

Also general formula for gaussian pdf is ; y=$f(x|\mu,\sigma) = (\frac{1}{\sigma\sqrt{2\pi}})e^{\frac{-(x-\mu)^2}{2(\mu)^2}}$
As we can see and understand from the variables above and the formula that all variables has been specified and we only need σ, μ and range.

2.2 Explanation of Q(u) function in relation to the normal probability density function 2.2

As we have explained and investigated probability density density function above . We can easily link $Q(u)$ function by exploiting the properties of cumulative distribution function and Probability density function.

- Q function is the 1- minus the cumulative distribution function of the standardised normal variable.
- Gaussian pdf with unit variance and zero mean is R=$(\frac{1}{\sqrt{2\pi}})e^{\frac{-(x)^2}{2}}$
- And corresponding cumulative distribution function become; $P = \int_{-\inf}^{x} Z$
- In last step Gaussian Q function defined as $Q(x) = 1 - P(x) = \int_{x}^{-\inf} Z$

2.3 Preparing a script file for plotting the $Q(u)$ function for argument values of relevance to the detection problems for digital communication receivers and inserting them in Appendix 2.3 and 2.4

Script file that has been created for the assignment 2.3 and 2.4 has been added to the Appendix. Important concept has been explained and relevant digital communication input values specified.

2.4 2.5

2.4.1 Complementary error function

2.5 Plots

Plotting for the questions 2.3 and 2.4

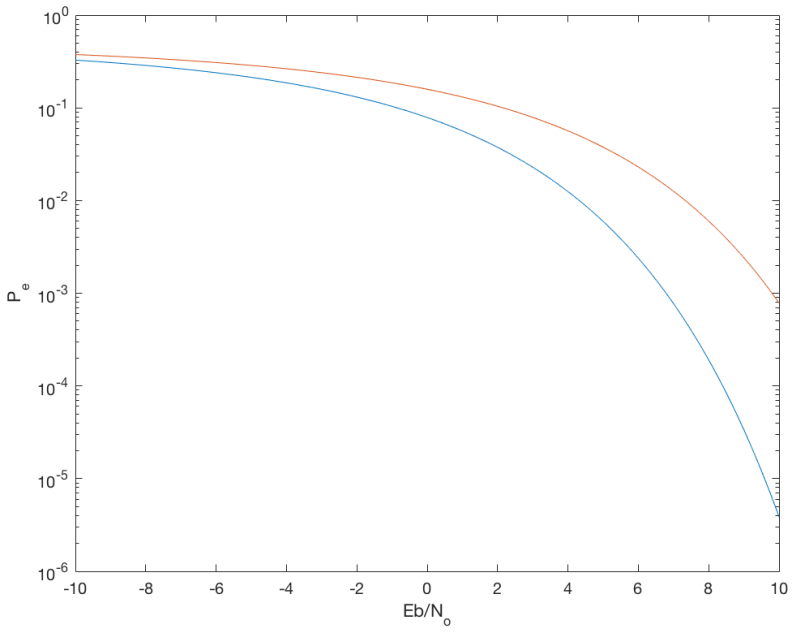

Figure 2: Plot for the $Q(u)$ function relevant to the argument values for the detection problems for digital communication receivers

3 Source Code

Here we have the properly prepared MATLAB codes for the second part of the second project. It has been used for observations, calculations and comparing with specified commands that given in this project.

The code belows computes and graphs normal(Gaussian) probability density function (pdf) in an appropriate intervals

Listing 2: ../scripts/2/graphingpdf.m

```
%graphing PDF function with random variables
mu=0;
sigma=10;
MAX = 50;
MIN = -50;
STEP = (MAX - MIN) / 1000;
PDF = normpdf(MIN:STEP:MAX, mu, sigma);
plot(MIN:STEP:MAX, PDF)
```

For the question 2.3 and 2.4 we have created the following code below. This code will plot the $Q(u)$ function for argument values relevant to the

digital communication receivers. As we know from previous chapters that there are two common argument values :

- $R_1 2 = 0$ (Orthogonal Signals)
- $R_1 2 = -1$ (Antipodal Signals)

We also used $z_d b$ for the ratio for E_b/N_o. We should also notice that $z_d b$ is dimensionless ratio.

After that we have investigated the graph for further investigation to see whether we have achieved a satisfactory results for the assignment 2.3 and 2.4. Mathematical calculations are matching up with MATLAB simulation results.

Listing 3: ../scripts/2/qfunction.m

```
z_db = -10:.1:10;
r12 = [-1 0];
z = db2pow(z_db);
p_e(1,:) = qfunc(sqrt((1 - r12(1))*z));
p_e(2,:) = qfunc(sqrt((1 - r12(2))*z));
semilogy(z_db, p_e(1,:))
hold on
semilogy(z_db, p_e(2,:))
xlabel('Eb/N_o')
ylabel('P_e')
```

4 The Matched Filter Base Band Receiver

4.1 Additive white gaussian noise model

4.2 `c8ce1a.m`: explanation

4.3 Extend the code in c8ce1a.m such that the number of input correlations can be up to 8. Presently 4 is the maximum of input arguments 3.4

For the part 3.4 `question35.m` MATLAB code has been created. It can be seen that code is able take 8 correlation input. It is accessible from the appendix part of this report.

4.4 Apply the following correlation coefficients in the c9ce1a script $[-1 -0.75 -0.50 0.5 0.75 0.8 .995]$ and insert the resulting plot in the project report. Explain the plot and verify that it is correct 3.5

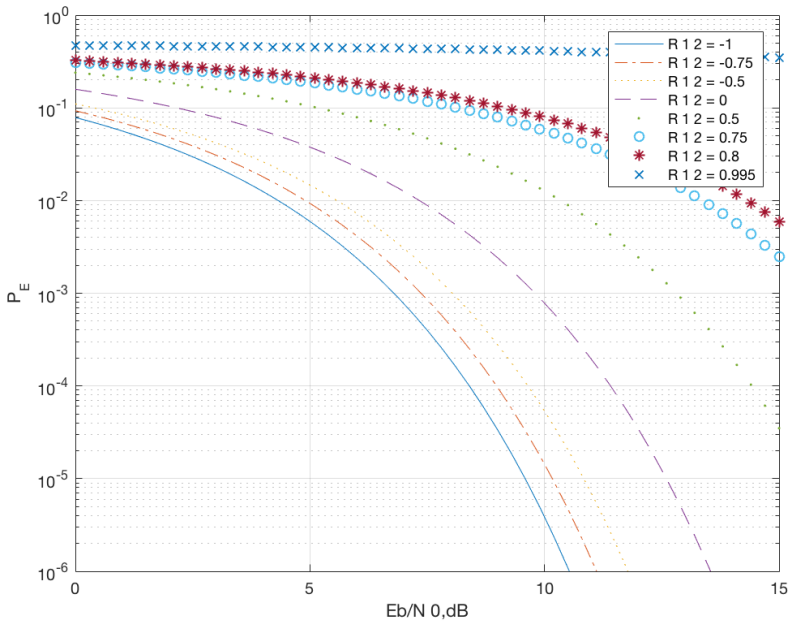

Figure 3: P_E over E_b/N_o graph for investigating how it will change with introduced correlation coefficients

4.5 Explain how a matched-filter receiver can be implemented by a correlator. The explanation should include the most important formulas leading to the matched filter implementation using a cross correlation Part 3.6

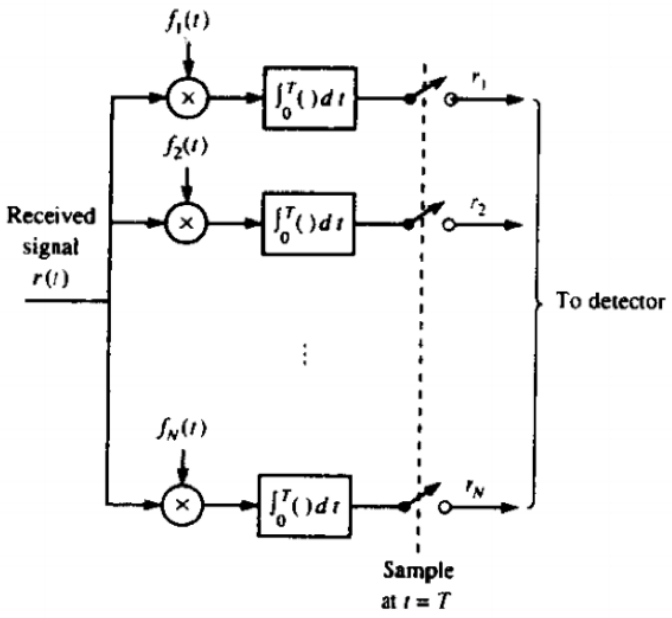

Figure 4: Correlator Matched Filter

As we can from the Figure 3 that we can implement the matched filter by implementing the correlation detectors. The cross-correlator does the cross-correlation between the noisy signal and noiseless signal.

If we look at the matched filter, it does the convolution between the received signal and the time-reversed copy of the reference signal. We can mathematically prove that both matched filter and the cross-correlator will give us same output signal values. To prove this we will derive mathematical derivation.

Suppose we have received $r(t)$ and is passed through parallel bank of N ?crosscorrelators? which basically compute the prediction of $r(t)$ onto the N-basis functions as $f_n(t)$.This can be seen in Figure 3.

For cross-correlator signal is now represented by $s_m(t)$ with components $s_{mk}(t)$. $s_{mk}(t)$ values depend on how many M_{signal} is transmitted. Now other

components of cross-correlator will be explained below for the mathematical derivation for the output signal.

- $\int_0^T r(t)f_k(t) = \int_0^T [s_m(t) + n(t)]f_k(t)dt$
- $r_k(t) = s_m(t) + n_k(t)$
- $\int_0^T s_m(t)f_k(t)$ for the values of $k = 1, 2, 3, 4, 5, ...$
- $\int_0^T n(t)f_k(t)$ for the values of $k = 1, 2, 3, 4, 5.....$

Now our signal has been represented by the $s_m(t)$ and the $n(k)$ (random variables as noise)

Now the signal can be represent $r(t)$ as

- $r(t) = \sum_{k=1}^{N} = s_m(t)f_k(t) + n(t)' + \sum_{k=1}^{N} = n_k(t)f_k(t)$
- $\sum_{k=1}^{N} = r_k(t)f_k(t)$

As we can see that $n(t)'$ is irrelevant to which signal is going to transmitted. The decision will be made upon entirely on correlator output and the basis functions.

Now we can compare with Matched filter output.

For matched filter, we use N bank linear filters. Impulse responses of the N filters are;

- $h_k(t) = f_k(T - t)$ for invertals between , $0 < t < T$

The output filters become:

- $y_k(t) = \int_0^T r(\tau)f_k(T - \tau)d(\tau)$
- $\int_0^T r(\tau)f_k(T - t + \tau)d(\tau)$ for the $k = 1, 2, 3, 4, 5, ...N$

If we sample outputs of the linears at $t = T$

- $y_k(T) = \int_0^T r(\tau)f_k(\tau)d(\tau)$ which will eventually become r_k

Hence our theoritical approach has been proved by mathematical derivation of matched filter and cross-correlator. Both have same output even though calculations have been done in a different way for both.

4.6 Explain what the consequences are with respect to a matched-filter receiver if there is noise on the timing synchronisation in the receiver Part 3.7

Even if we manage to recover the timing, it does not guarantee that the correct operation of data-aided frequency estimation algorithms. The reason is that for is the presence of noise on the timing synchronisation in the receiver. Due to fact that for a frequency offset in order of $1/T$ the signal will be severely distorted when it passed through the matched filter. Severely distorted matched filter will not able to maximise the signal to noise ratio in the presence of additive noise.

4.6.1 Example for the Part 3.7

For example, if our noise signal is delta, sampling the matched filter?s output at some time $T+\delta$ (where represents a receiver timing offset due to introduction of noise) will significantly reduce the effective SNR seen by subsequent receiver blocks. This example that linked to theory above proves that it is important to keeping receiver timing offset close to zero as possible and thus delivers motivation for the inclusion of a timing recovery loop in the receiver.

4.7 Plots

4.8 Source Code

For the question 3.5 following code has been created to take relevant correlation coefficients

Listing 4: ../scripts/2/question35.m

```
% file: c9ce1
% Bit error probability for binary signaling;
% vector of correlation coefficients allowed
%
clf
R12 = input('Enter␣vector␣of␣desired␣R␣1␣2␣values;␣<=␣8␣values␣'
    );
A = char('-','-.',':','--','.','o','*','x');
LR = length(R12);
z_dB = 0:.3:15;
z = db2pow(z_dB);
for k = 1:LR
    P_E=qfn(sqrt(z*(1-R12(k)))); % Probability of error for
        vector of z-values
```

```
                                % Plot probability of error
                                   versus Eb/N0 in dB
    semilogy(z_dB,P_E,A(k,:)),axis([0 15 10^(-6) 1]),xlabel('Eb/
       N␣0,dB'),ylabel('P_E'),...
    if k==1
    hold on; grid % Hold plot for plots for other values of R12
       end
    end
end
if LR == 1 % Plot legends for R12 values
legend(['R␣1␣2␣=␣',num2str(R12(1))])
elseif LR == 2
legend(['R␣1␣2␣=␣',num2str(R12(1))],['R␣1␣2␣=␣',num2str(R12(2))
   ])
elseif LR == 3
legend(['R␣1␣2␣=␣',num2str(R12(1))],['R␣1␣2␣=␣',num2str(R12(2))
   ],['R␣1␣2␣=␣',num2str(R12(3))])
elseif LR == 4
legend(['R␣1␣2␣=␣',num2str(R12(1))],['R␣1␣2␣=␣',num2str(R12(2))
   ],['R␣1␣2␣=␣',num2str(R12(3))],['R␣1␣2␣=␣',num2str(R12(4))])
elseif LR == 5
legend(['R␣1␣2␣=␣',num2str(R12(1))],['R␣1␣2␣=␣',num2str(R12(2))
   ],['R␣1␣2␣=␣',num2str(R12(3))],['R␣1␣2␣=␣',num2str(R12(4))],[
   'R␣1␣2␣=␣',num2str(R12(5))])
elseif LR == 6
legend(['R␣1␣2␣=␣',num2str(R12(1))],['R␣1␣2␣=␣',num2str(R12(2))
   ],['R␣1␣2␣=␣',num2str(R12(3))],['R␣1␣2␣=␣',num2str(R12(4))],[
   'R␣1␣2␣=␣',num2str(R12(5))],['R␣1␣2␣=␣',num2str(R12(6))])
elseif LR == 7
legend(['R␣1␣2␣=␣',num2str(R12(1))],['R␣1␣2␣=␣',num2str(R12(2))
   ],['R␣1␣2␣=␣',num2str(R12(3))],['R␣1␣2␣=␣',num2str(R12(4))],[
   'R␣1␣2␣=␣',num2str(R12(5))],['R␣1␣2␣=␣',num2str(R12(6))],['R␣
   1␣2␣=␣',num2str(R12(7))])
elseif LR == 8
legend(['R␣1␣2␣=␣',num2str(R12(1))],['R␣1␣2␣=␣',num2str(R12(2))
   ],['R␣1␣2␣=␣',num2str(R12(3))],['R␣1␣2␣=␣',num2str(R12(4))],[
   'R␣1␣2␣=␣',num2str(R12(5))],['R␣1␣2␣=␣',num2str(R12(6))],['R␣
   1␣2␣=␣',num2str(R12(7))],['R␣1␣2␣=␣',num2str(R12(8))])

end
```