Designing and Simulating a Communication Link - Part 1

Table of Contents

Ivan Chowdhury, Hanoch Goldfarb

ECE-300: Communication Theory

Professor Keene

Fall 2018

```
clear all;close all;clc     % Reset workspace

% Set Simulation Parameters
numIter = 5;  % The number of iterations of the simulation.
nSym = 1000;    % Constraint: Max 1000 symbols per packet
SNR_Vec = 0:2:16;   % Vector that stores the Signal-to-Noise Ratios
lenSNR = length(SNR_Vec);   % Length of SNR Vector
BER_Vec = zeros(numIter, lenSNR);   % Vector that stores the BER
 computed during each iteration
```

Set modulation parameters

```
% M = 4;   % 4-QAM
% M = 16; % 16-QAM
M = 2;  % BPSK
k = log2(M);
```

Set channel

```
% chan = 1;      % No channel (for QAM)
chan = [1 .2 .4]; % Somewhat invertible channel impulse response,
 Moderate ISI (for BPSK)
```

Create Equalizer object (for BPSK)

These two equalizers performed the best of the 4.

```
Equalizer = dfe(5,3,lms(0.01));    % Decision Feedback / LMS
```

```
%Equalizer = lineareq(8, lms(0.01));   % Linear/LMS
```

These two equalizers performed worse, but also met specifications (approximately 10e-4 BER).

```
% Equalizer = lineareq(10,rls(0.3));  % Linear/RLS:
% Equalizer = dfe(3,2,rls(0.3));    % Decision Feedback / RLS
```

Configure Equalizer

```
Equalizer.SigConst = pskmod(((0:M-1)'),M)'; % Set ideal signal
 constellation.
Equalizer.ResetBeforeFiltering = 0; % Resets equalizer before use
trainlen = 500;     % Number of training symbols
```

Run simulation (numIter times)

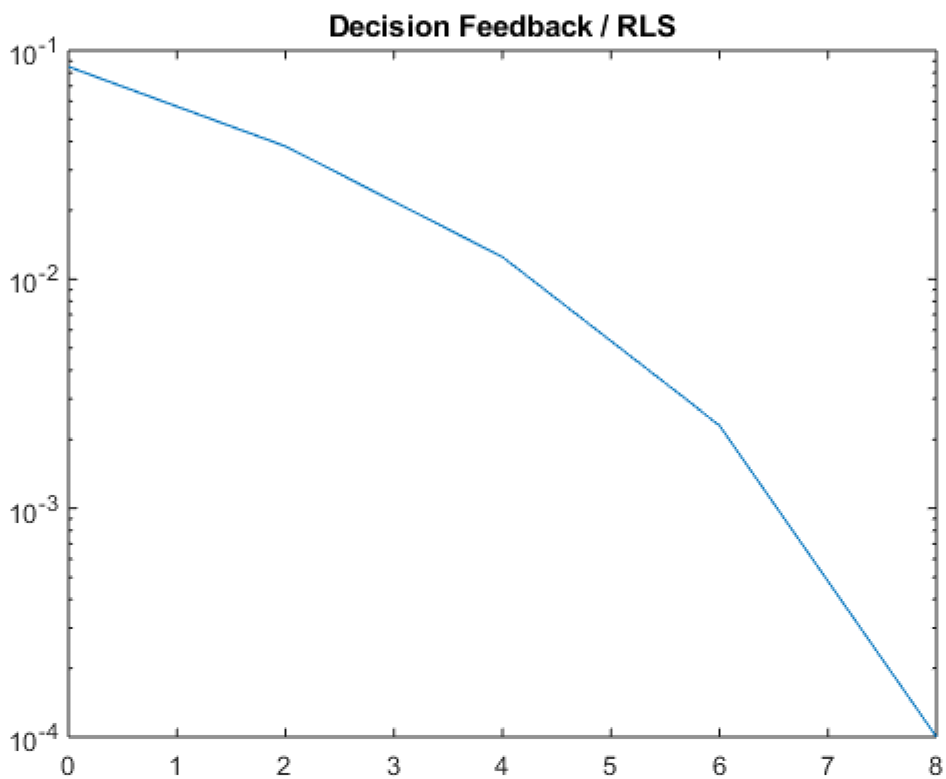
```
for i = 1:numIter

   bits = randi(2,[nSym*M, 1])-1;  % Generate random binary data for
 each iteration

   bitsMatrix = reshape(bits, length(bits)/k,k);   % Reshape data for
 bi2de function
   msg = bi2de(bitsMatrix); % Convert to bits to integers

   for j = 1:lenSNR % Perform one iteration of the simulation at each
 SNR Value

       % tx = qammod(msg,M);  % QAM modulate the signal
       tx = pskmod(msg,M);  % BPSK modulate the signal

       % Draw and apply channel
       if isequal(chan,1)
           txChan = tx;
       elseif isa(chan,'channel.rayleigh')
           reset(chan) % Draw a different channel each iteration
           txChan = filter(chan,tx);
       else
           txChan = filter(chan,1,tx);  % Apply the channel to
 transmitted signal.
       end

       txEq = equalize(Equalizer,txChan,tx(1:trainlen)); % Apply
 Equalizer (for BPSK)

       % Convert from EbNo to SNR. Add and scale noise
       txNoisy = awgn(txEq,10*log10(k)+SNR_Vec(j),'measured');     %
 Equalized channel (for BPSK)
       % txNoisy = awgn(txChan,10*log10(k)+SNR_Vec(j),'measured');
 % Unequalized channel (for QAM)

       % Demodulate signal
       % rx = qamdemod(txNoisy,M); % Unequalized (for QAM)
```

```
        rx = pskdemod(txNoisy,M); % Equalized (for BPSK)

        % Convert symbols back into bits
        rxMatrix = de2bi(rx, k);
        rxMSG = rxMatrix(:);

        [zzz BER_Vec(i,j)] = biterr(bits, rxMSG);  % Compute and store
 the BER for this iteration

    end  % End SNR iteration
end      % End numIter iteration
```

Compute & plot data

Compute and plot the mean BER

```
ber = mean(BER_Vec,1);

figure;
semilogy(SNR_Vec, ber)
title("Decision Feedback / RLS")
```



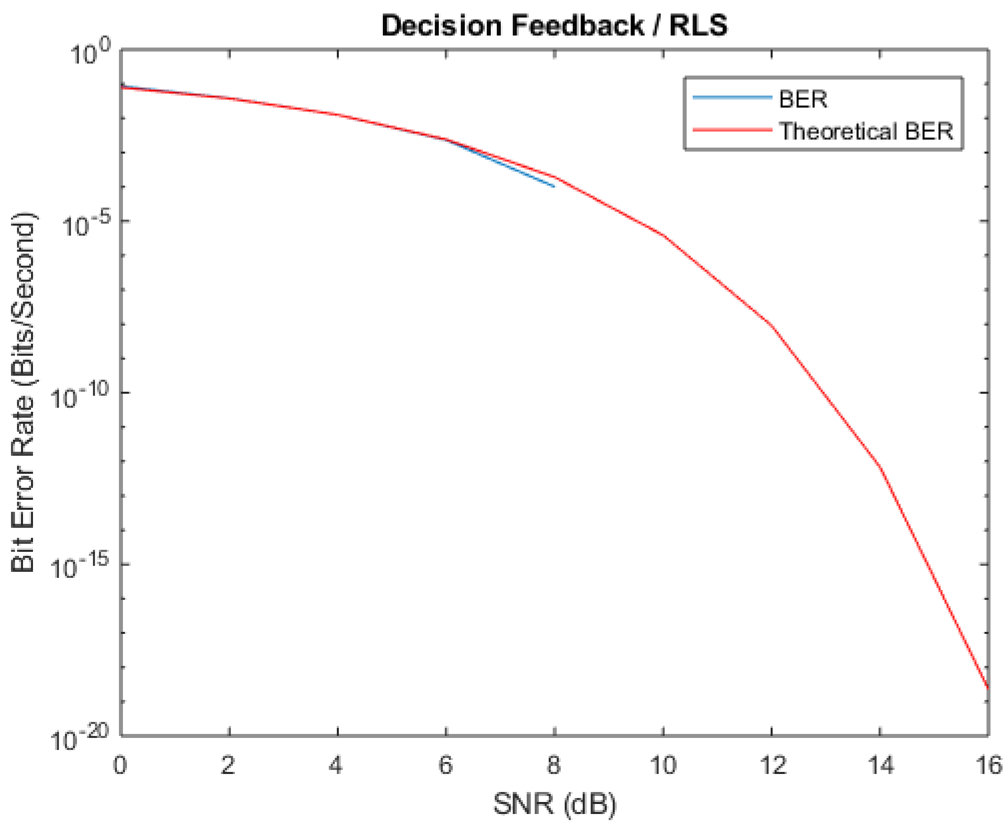
Compute the theoretical BER for this scenario

```
berTheory = berawgn(SNR_Vec,'pam',M);    % BPAM/BPSK (DEFAULT)
% berTheory = berawgn(SNR_Vec,'psk',M, 'nondiff');  % BPSK
```

```
% berTheory = berawgn(SNR_Vec,'qam',M); % QAM
```

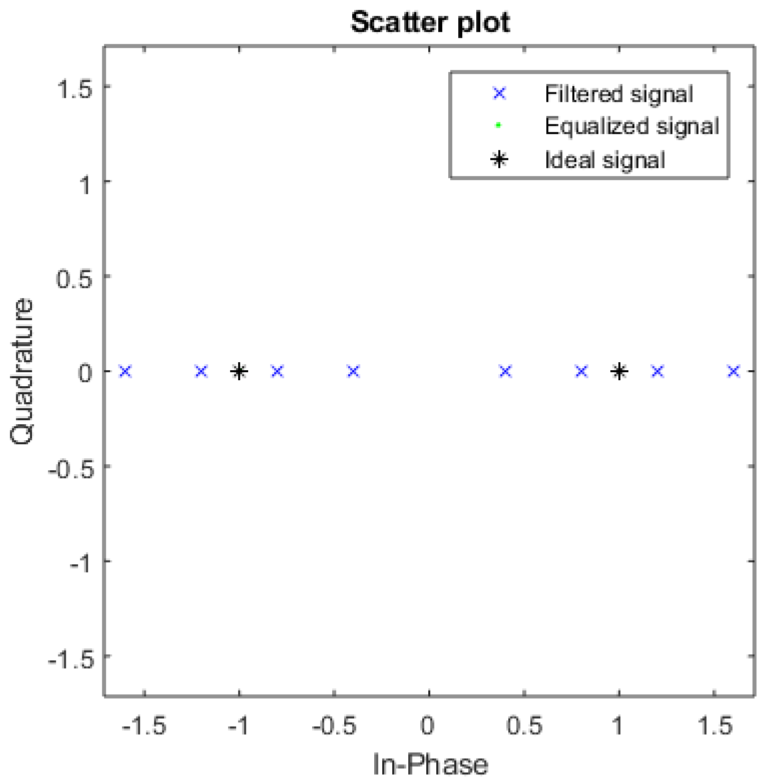
Plot the theoretical BER

```
hold on
semilogy(SNR_Vec,berTheory,'r');
xlabel('SNR (dB)')
ylabel('Bit Error Rate (Bits/Second)')
legend('BER', 'Theoretical BER')
hold off
```

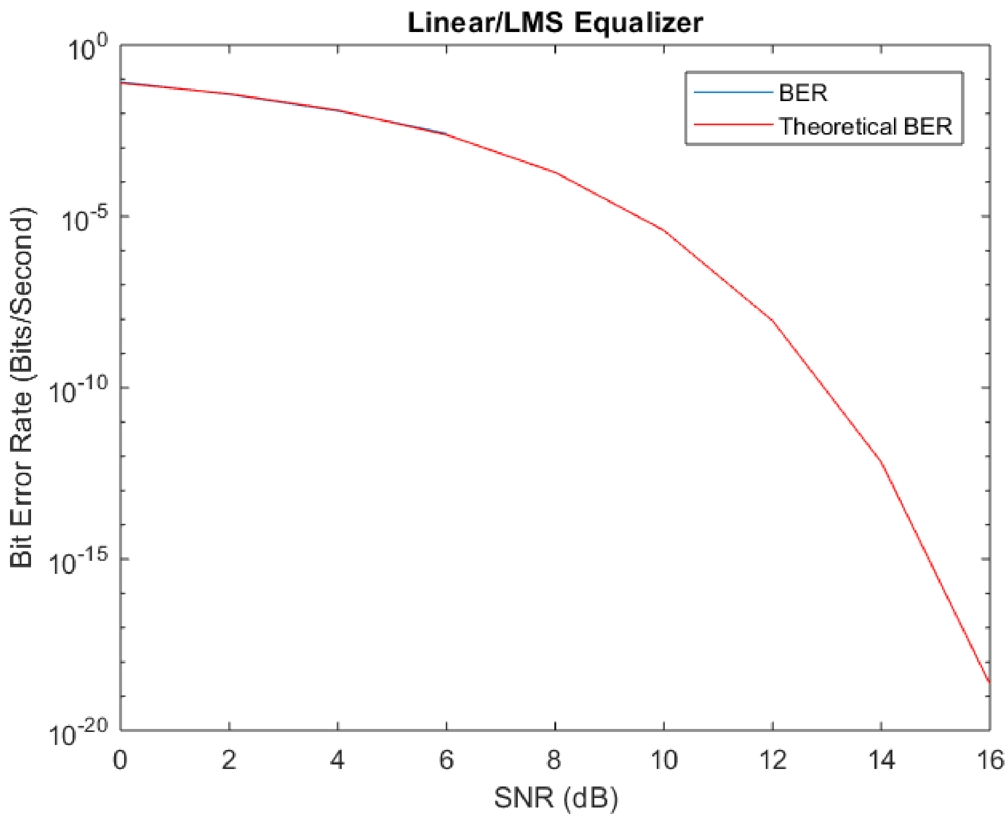


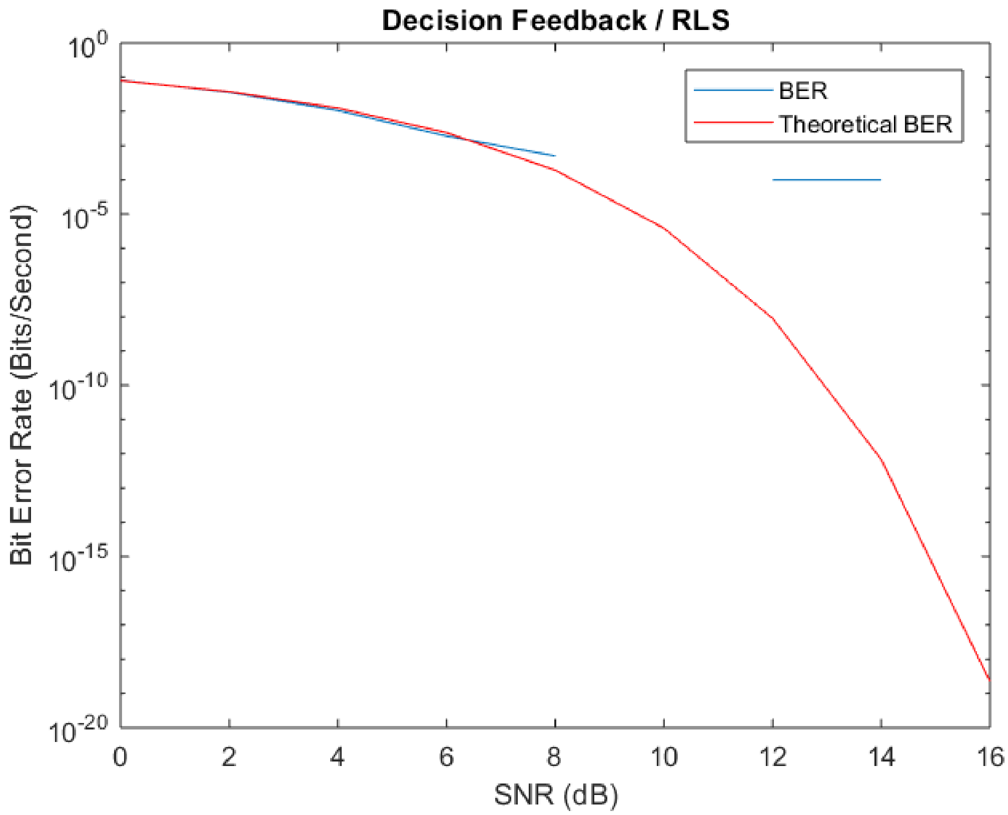
Compute and plot the signal constellation

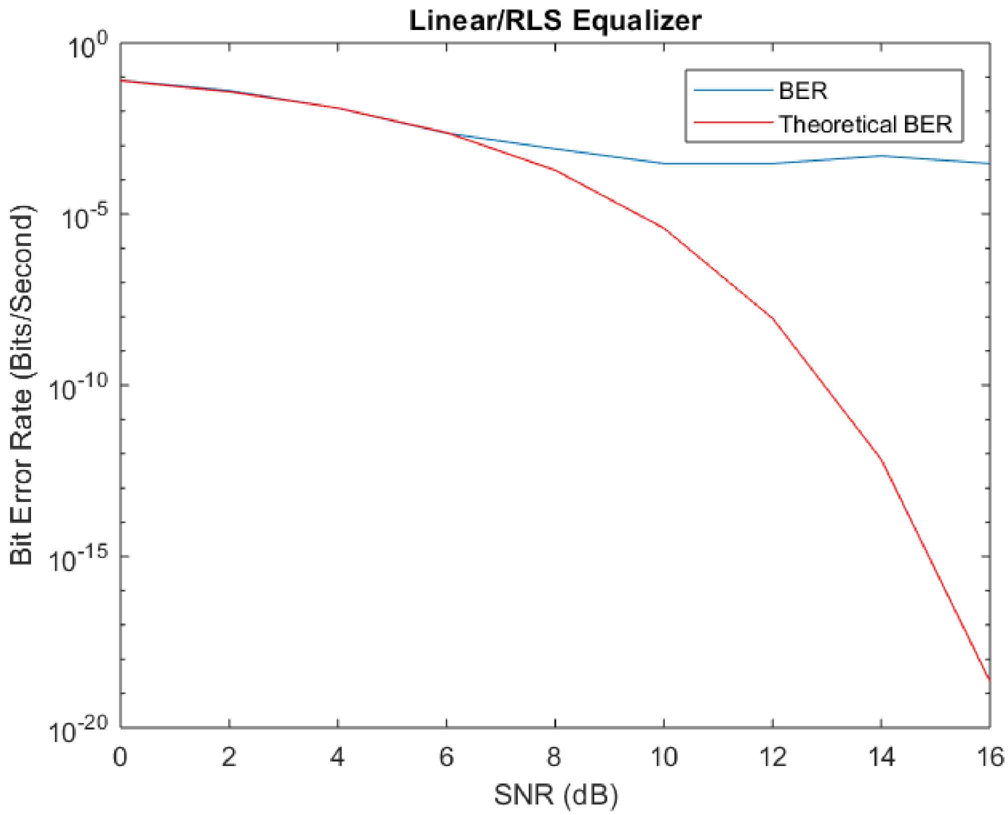
```
constellation = scatterplot(txChan,1,trainlen,'bx');     % Filtered
 signal constellation

hold on;
scatterplot(txEq,1,trainlen,'g.',constellation);     % Equalized signal
 constellation
scatterplot(Equalizer.SigConst,1,0,'k*',constellation); % Ideal signal
 constellation
legend('Filtered signal','Equalized signal',...
    'Ideal signal');
hold off;
```


Scatter plot
Filtered signal
Equalized signal
Ideal signal
Quadrature
In-Phase
1.5
1
0.5
0
-0.5
-1
-1.5

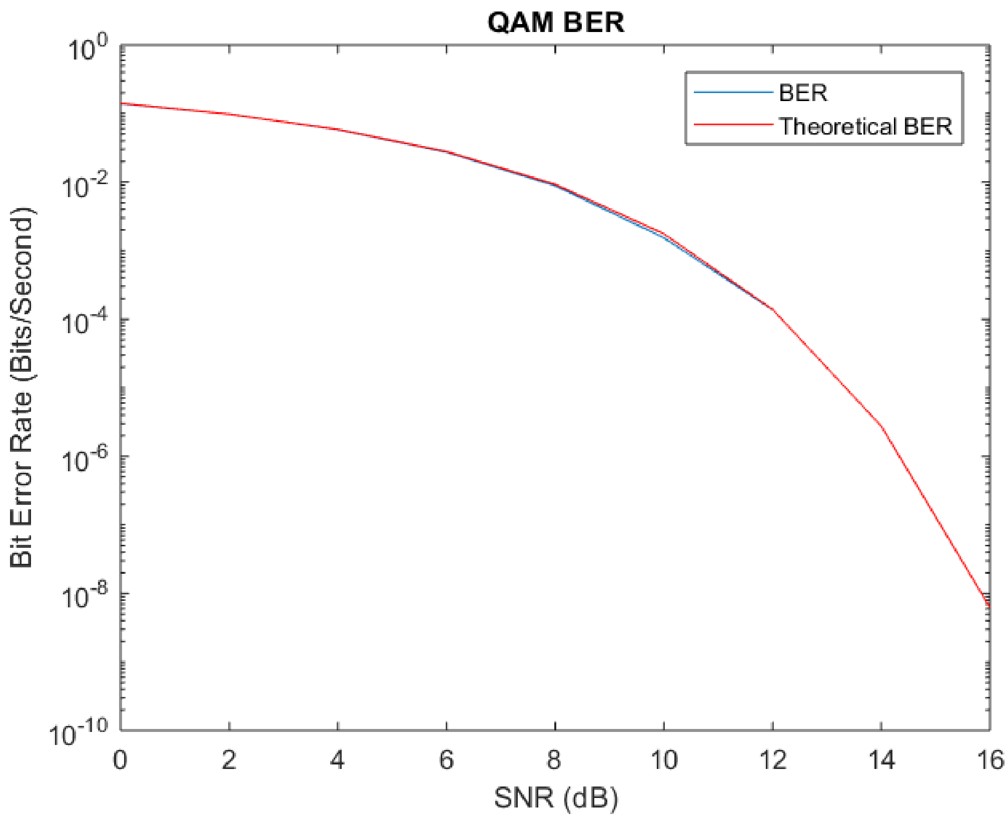
Other equalizer configurations

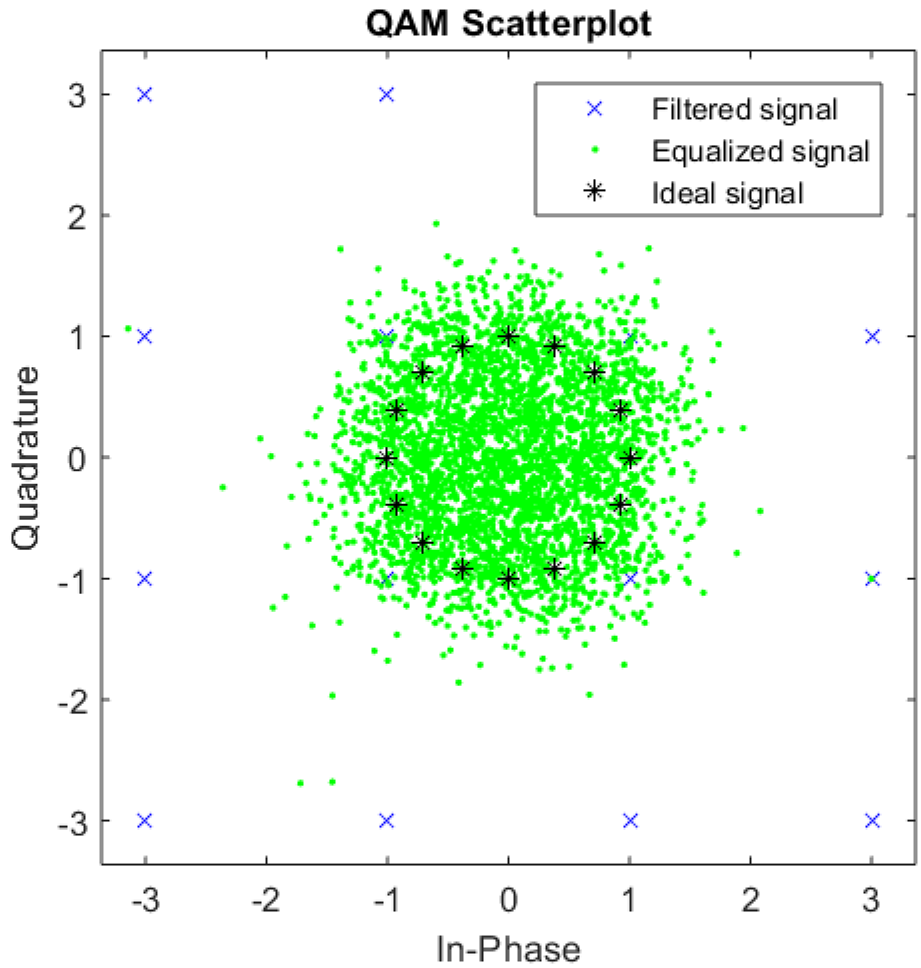



Decision Feedback / RLS
BER
Theoretical BER
Bit Error Rate (Bits/Second)
SNR (dB)
10^0
10^-5
10^-10
10^-15
10^-20
0
2
4
6
8
10
12
14
16


Linear/RLS Equalizer
BER
Theoretical BER
Bit Error Rate (Bits/Second)
SNR (dB)
10^0
10^-5
10^-10
10^-15
10^-20
0
2
4
6
8
10
12
14
16

QAM figures


QAM BER
BER
Theoretical BER
Bit Error Rate (Bits/Second)
SNR (dB)
10^0
10^-2
10^-4
10^-6
10^-8
10^-10
0
2
4
6
8
10
12
14
16



Published with MATLAB® R2017b