BigCat Wireless - EC401
Assignment 2

Vignesh Mohan
183002181
Electronics and Communication Department
Sri Sivasubramiya Nadar College of Engineering, 603110

1. From the constellation plot shown in figure below, answer the following questions?

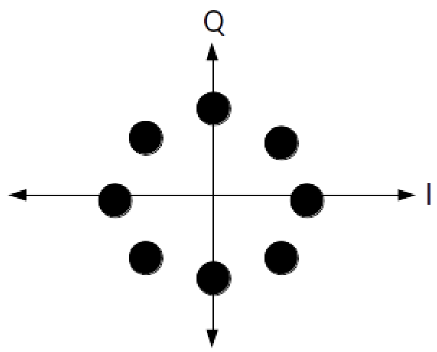

Figure 1: Constellation plot which is to be identified

1. *What type of modulation does this represent?*

   The modulation is 8-PSK

2. *How many symbols (M) are represented?*

   8 symbols

3. *How many bits per symbol (N) are used?*

   3 bits

4. *What's the bit rate when baud rate is 10,000 symbols/second?*

   30,000 bits/second (30kbps)

5. *Compare the noise performance of this modulation with 16-QAM, which is better?*

   8-PSK should in general be less susceptible to noise because the symbols would be spread further apart. Approximately, the minimum distance for constellation points for 8-PSK is $1.325\sqrt{E_b}$ which is slightly greater than for 16-QAM which is $1.265\sqrt{E_b}$.

6. *Compare the spectral efficiency of this modulation with QPSK, which is better?*

   8-PSK has better spectral efficiency since it uses $2R_b/3$ bandwidth which is less than the bandwidth R_b used by QPSK

2. Solve the IQ demodulator equation showing how I and Q components are received back.

Composite Input signal:

$$X(t) = I * cos(2\pi f_c t) - Q * sin(2\pi f_c t)$$

Inphase Component (multipled with LO signal):

$$\begin{aligned} X_i(t) &= (I * cos(2\pi f_c t) - Q * sin(2\pi f_c t)) * cos(2\pi f_c t) \\ &= \frac{I}{2} + \frac{I}{2} * cos(2\pi f_c t) - \frac{Q}{2} * sin(2\pi f_c t) \end{aligned}$$

Quadrature Component (multipled with $\pi/2$ shifted LO signal):

$$\begin{aligned} X_q(t) &= (I * cos(2\pi f_c t) - Q * sin(2\pi f_c t)) * -sin(2\pi f_c t) \\ &= \frac{Q}{2} - \frac{Q}{2} * cos(2\pi f_c t) - \frac{I}{2} * sin(2\pi f_c t) \end{aligned}$$

After LPF (sin,cos terms with f_c are cancelled):

$$\begin{aligned} X_i(t) &= \frac{I}{2} \\ X_1(t) &= \frac{Q}{2} \end{aligned}$$

3. Calculate maximum theoretical spectral efficiency for:

1. *$\pi/4$ QPSK:*

   Maximum theoretical spectral efficiency = 2 bits/s/Hz.

2. *256-QAM:*

   Maximum theoretical spectral efficiency = 8 bits/s/Hz.

4. Create a transmitter and receiver architecture.
Use the following parameters:
Input bit rate 1 Mbps
Modulation method: QPSK or 16-QAM
RRC roll-off factor 0.2
RRC output samples per symbol: 8
Carrier frequency 2.5MHz
Use AWGN channel with SNR =10dB
Equivalent settings in receiver side

```
% passband_modulation.m
close all;
bitrate = 1e06;
Fc = 2.5e06;
Fs = 8 * bitrate;

M = 4; % Modulation order
k = log2(M); % Bits/symbol
n = 500; % Transmitted bits
sps = 8; % Samples per symbol
EbNo = 10; % Eb/No (dB)

span = 8; % Filter span in symbols
rolloff = 0.20; % Rolloff factor

txfilter = comm.RaisedCosineTransmitFilter('RolloffFactor',
rolloff, 'FilterSpanInSymbols', span, ...
  'OutputSamplesPerSymbol', sps);

rxfilter = comm.RaisedCosineReceiveFilter('RolloffFactor',
rolloff, 'FilterSpanInSymbols', span, ...
  'InputSamplesPerSymbol', sps, ...
  'DecimationFactor', sps, 'Gain', 2);

ri = comm.internal.RandomIntegerGenerator('SetSize', M, ...
  'SampleTime', 1 / bitrate, 'SamplesPerFrame', n);

tx_lo = dsp.SineWave(1, Fc, 0, 'ComplexOutput', true, ...
  'SampleRate', Fs, 'SamplesPerFrame', sps * n);

rx_lo = dsp.SineWave(1, Fc, 0, 'ComplexOutput', true, ...
  'SampleRate', Fs, 'SamplesPerFrame', sps * n);

filtDelay = (txfilter.FilterSpanInSymbols + ...
  rxfilter.FilterSpanInSymbols) / 2;
errorRate = comm.ErrorRate('ReceiveDelay', filtDelay);

delay = dsp.Delay(8);
evm = comm.EVM();
```

```
biterr = 0;
totalbits = 0;
rmsEVM = 0;
tx_scope = dsp.SpectrumAnalyzer('SpectrumType', "Power Density", ...
  'SampleRate', Fs, 'FrequencyResolutionMethod', ...
  "WindowLength", 'WindowLength', 800, ...
  'PlotAsTwoSidedSpectrum', false);

rx_scope = dsp.SpectrumAnalyzer('SpectrumType', "Power", ...
  'SampleRate', Fs, 'FrequencyResolutionMethod', ...
  "WindowLength", 'WindowLength', 1024);

txSigall = [];
rxSigall = [];

tx_const_diag = scatterplot(ri());
rx_const_diag = scatterplot(ri());

for idx = 1:20
  dataIn = ri();
  modSig = pskmod(dataIn, 4, pi / 4);
  txFilterSig = txfilter(modSig);
  tx_carrier = tx_lo();
  txSig = real(txFilterSig .* tx_carrier);
  txSigall = txSig;

  SNR = EbNo + 10 * log10(k) - 10 * log10(sps);
  noisySig = awgn(txSig, SNR, 'measured');

  rx_wave = conj(rx_lo());
  rxSig = noisySig .* rx_wave;
  rxSigall = rxSig;

  rxFilterSig = rxfilter(rxSig);

  dataOut = pskdemod(rxFilterSig, 4, pi / 4);

  errStat = errorRate(dataIn, dataOut);
  biterr = biterr + errStat(2);
  totalbits = totalbits + errStat(3);

  rmsEVM = (rmsEVM + evm(delay(modSig), rxFilterSig)) / 2;

  scatterplot(modSig, 1, 0, 'y.', tx_const_diag);
  scatterplot(rxFilterSig, 1, 0, 'y.', rx_const_diag);
  tx_scope(txSigall);
  rx_scope(rxSigall);
end
```

```

fprintf('\nBit Errors = %d', biterr);
fprintf('\nBits Transmitted = %d\n', totalbits);
fprintf('\nBER = %5.2e\n', biterr / totalbits);
fprintf('\nRMS EVM = %.2f %% \n', rmsEVM);

release(tx_scope);
release(rx_scope);
```

Sample output:
Bit Errors = 157 Bits Transmitted = 104840
BER = 1.50e-03
RMS EVM = 31.68

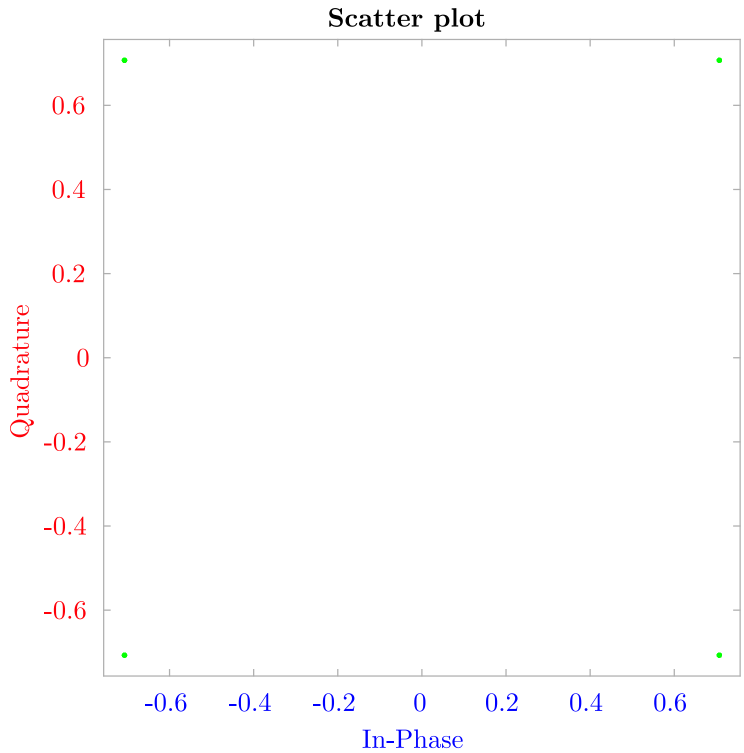

Figure 2: Transmitted constellation for QPSK

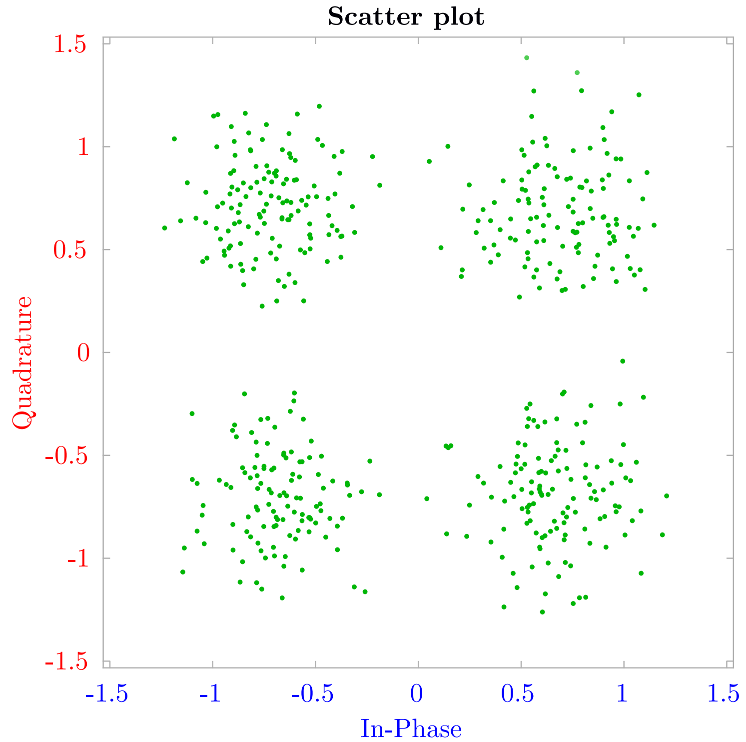

Figure 3: Received constellation for QPSK

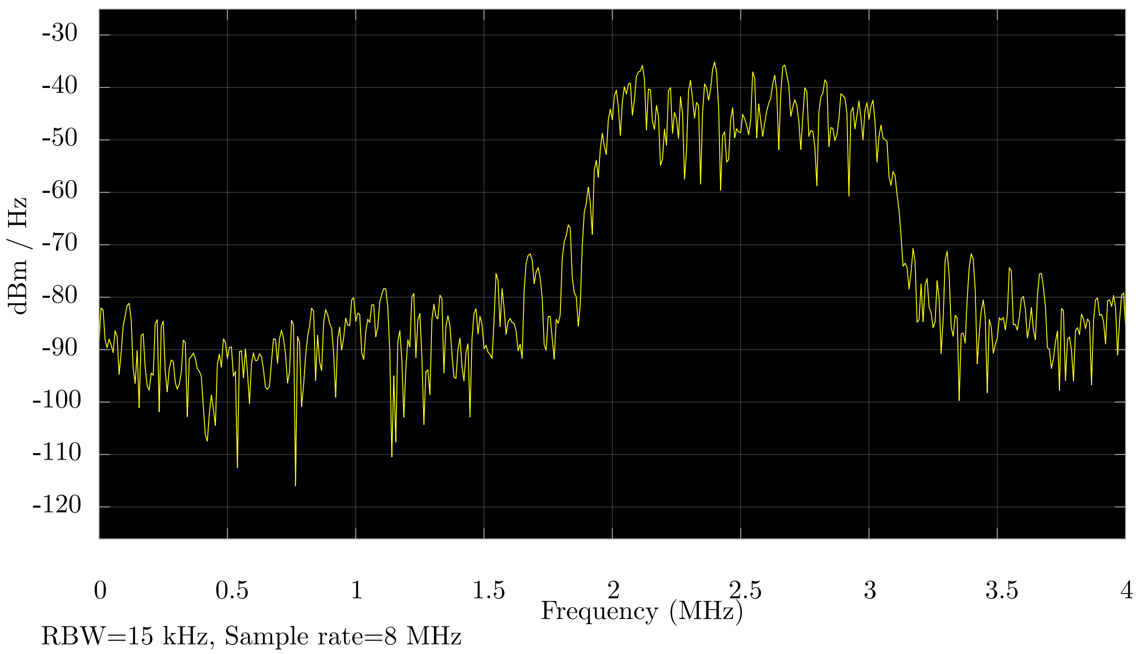

Figure 4: Transmitted spectrum for QPSK

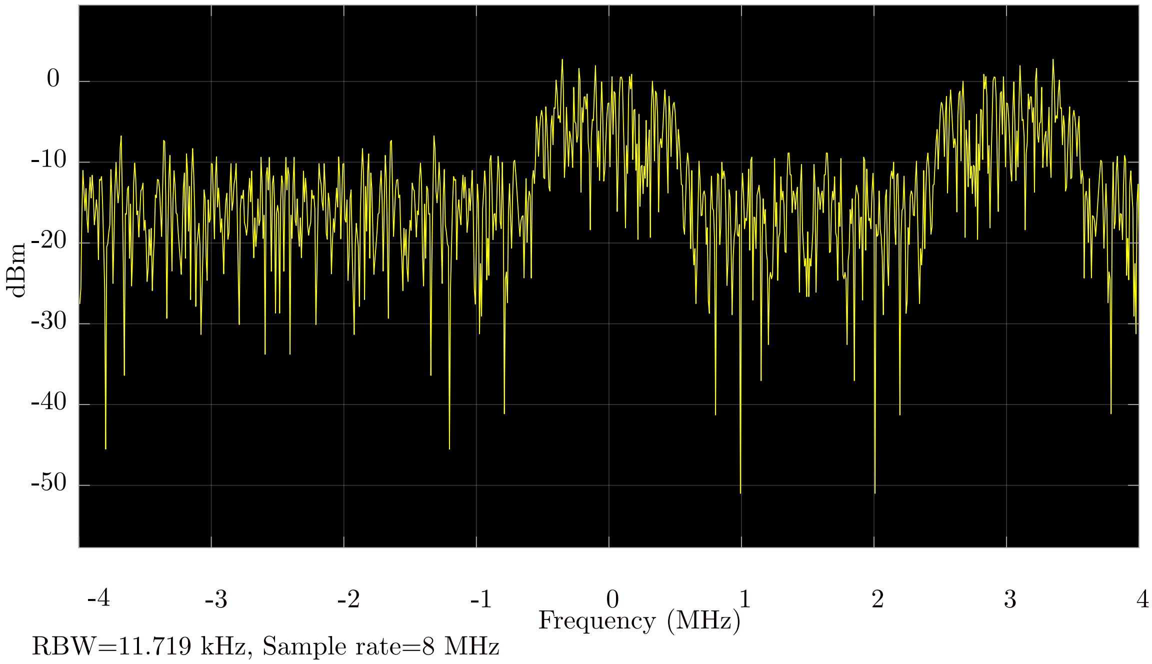

Figure 5: Received spectrum for QPSK