COMP40660 Assignment 3

Cian Dowd

25th March 2019

Part 1

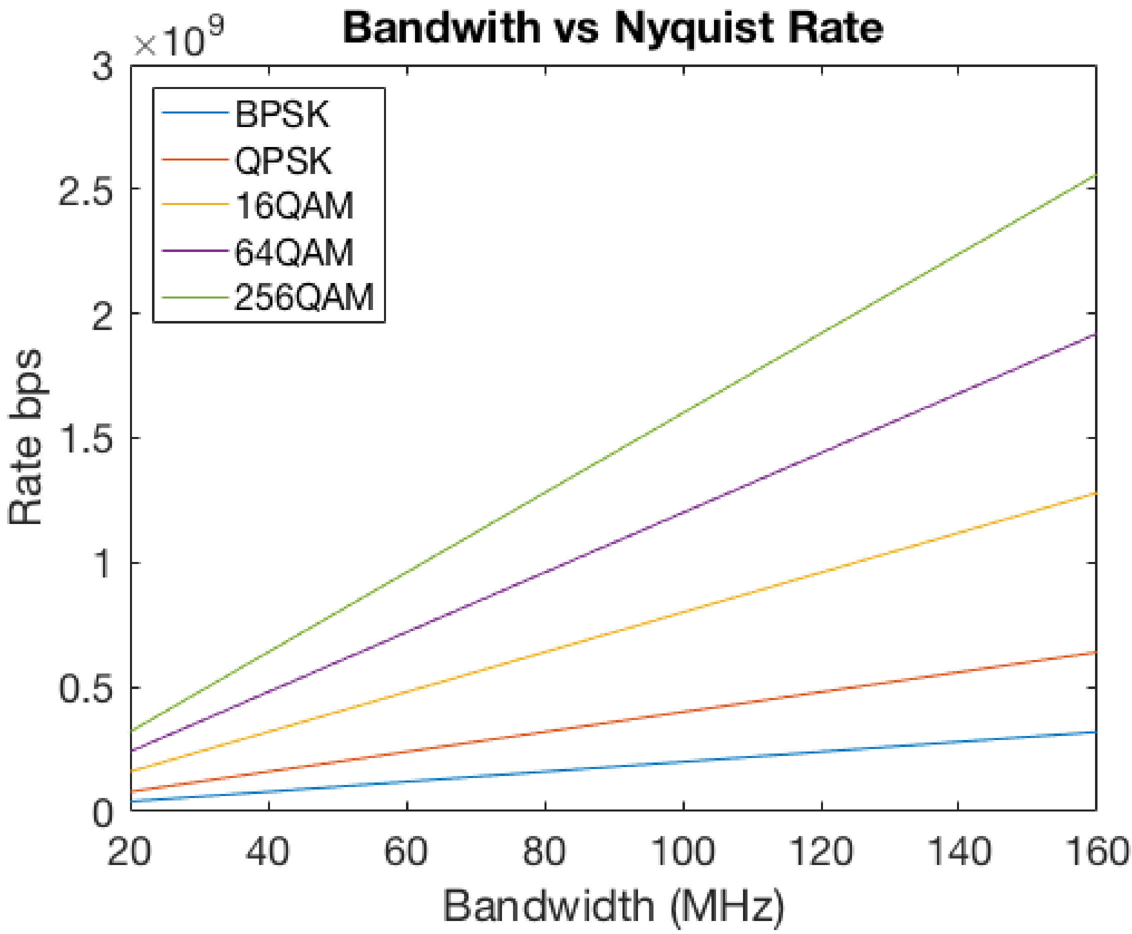

Figure 1: Nyquist Capacity versus Bandwidth

Part 2

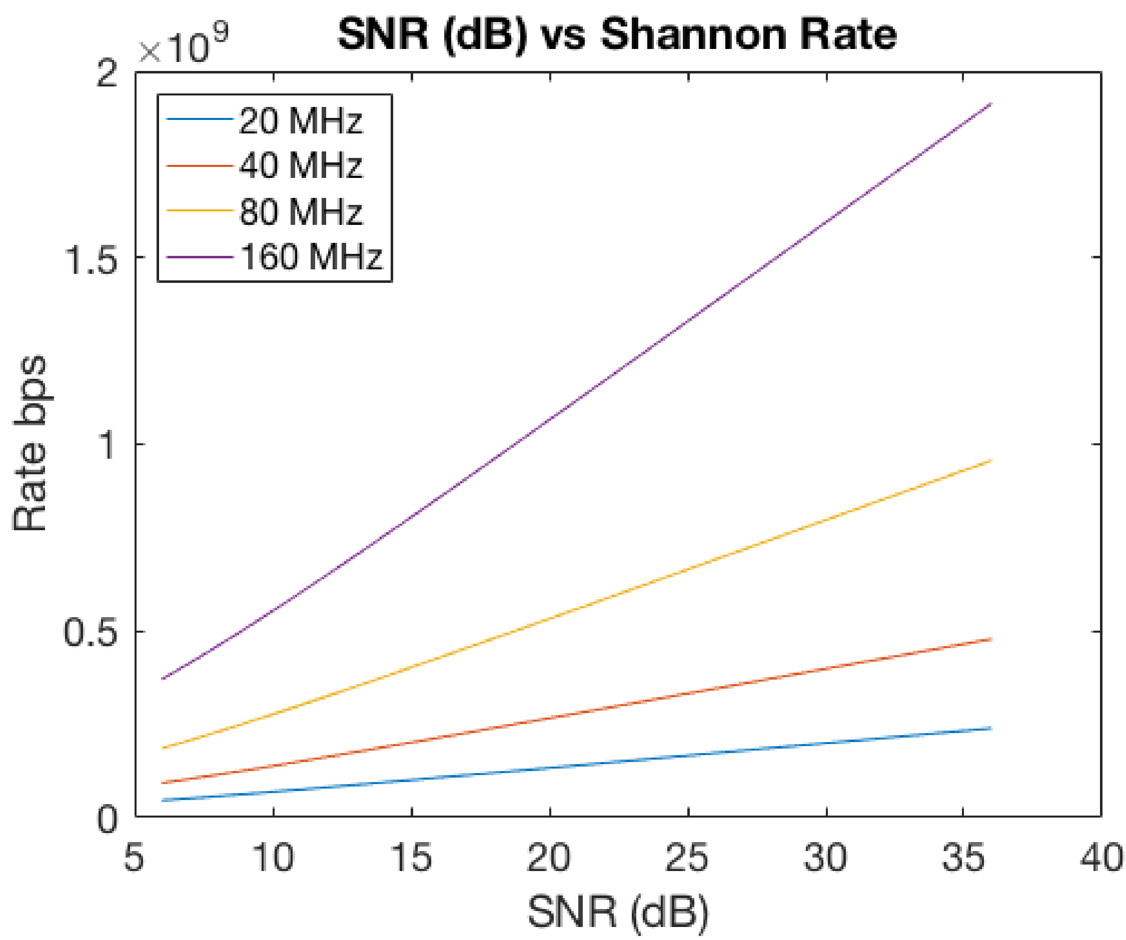

Figure 2: Shannon Capacity versus SNR (dB)

Part 3

Table 1: Suitable Modulation Formats

	20 MHz	40 MHz	80 MHz	160 MHz
5 dB	2	2	2	2
15 dB	4	4	4	4
20 dB	8	8	8	8
30 dB	16	16	16	16

Table 2: Suitable Modulation Formats

	20 MHz	40 MHz	80 MHz	160 MHz
5 dB	BPSK	BPSK	BPSK	BPSK
15 dB	QPSK	QPSK	QPSK	QPSK
20 dB	8PSK	8PSK	8PSK	8PSK
30 dB	16QAM	16QAM	16QAM	16QAM

Part 4

- Different models will have different relations to real wireless scenarios, though theoretical capacity values are typically overly optimistic. This is a result of theoretical capacity values failing to account for real scenarios with their assumptions. Some of the differences between the two models we are considering and real wireless scenarios are:

Nyquist assumes zero noise. This is inaccurate as there will always be at least some background noise and at times there will be other interferers.

Shannon assumes only additive white Gaussian Noise. This is a more realistic assumption than that of Nyquist's formula as there will always be some noise. Shannon's capacity formula becomes unrealistic in the presence of more complex noise and interference.

SNR can be affected by many factors, directly affecting either the signal strength or the strength of the noise.

Factors attenuating signal strength:

  - Free Space Loss
  - Feeder Lines
  - Filters

Factors increasing noise:

  - Background radiation in the channel
  - Interferers
  - Thermal noise in hardware

- The modulation limits affect the SNR, and therefore the likelihood of receiving the correct symbol, and the potential throughput of the system.
  For high SNR, where the receiver can make a correct decision, the higher modulation schemes allow for better throughput.
  For lower SNR, where the receiver may fail to make a correct decision, the higher modulation schemes will result in constellations which are more error prone, so lower modulation schemes will be used as a fallback.

- The percentage capacity improvement of `1024QAM` over `256QAM` is 25%.

MATLAB Code

```
function [  ] = CSAssignment3()
clear

%% Part 1
B = [20:1:160]'.*1e6; % BW options
M = [2 4  16 64 256]'; % modulation options
SNR_dB = [6:36]'; % dB options
SNR = idB(SNR_dB); % get ratio from dB

C_n = nyquist(B,M); % get Nyquist capacity

% plot figure
figure(1)
plot(B./(1e6),C_n);
legend('BPSK', 'QPSK', '16QAM', '64QAM', '256QAM','Location','NorthWest')
title('Bandwith vs Nyquist Rate')
xlabel('Bandwidth (MHz)')
ylabel('Rate bps')
set(gca,'fontsize', 18);

% print reference to console
fprintf('\nFirst Part:\nSee Figure 1\n')

%% Part 2
B = [20 40 80 160]'.*1e6; % BW options
SNR_dB = [6:36]'; % dB options
SNR = idB(SNR_dB); % get ratio from dB

C_s = shannon(B,SNR); % get Shannon capacity

% plot figure
figure(2)
plot(SNR_dB,C_s)
legend('20 MHz', '40 MHz', '80 MHz', '160 MHz','Location','NorthWest')
title('SNR (dB) vs Shannon Rate')
xlabel('SNR (dB)')
ylabel('Rate bps')
```

```
set(gca,'fontsize', 18);

% print reference to console
fprintf('\nSecond Part:\nSee Figure 2\n')

%% Part 3
B = [20 40 80 160]'.*1e6; % BW options
SNR_dB = [5 15 20 30]'; % dB options
SNR = idB(SNR_dB); % get ratio from dB

C_s = shannon(B,SNR); % get Shannon capacity
M = rev_nyquist(C_s, B); % determine modulation format

% print reference to console
fprintf('\nThird Part:\n')
fprintf('%2d %2d %2d %2d\n',M')

%% Part 4

% bps/Hz of 1024QAM and 256QAM
QAM1024 = 10;
QAM256 = 8;

pcCapacity = ((QAM1024-QAM256)/QAM256)*(100/1); % percentage improvement
    of 1024QAM over 256QAM

% print reference to console
fprintf('\nFourth Part:\nPercentage Capacity Improvement = %d%%\n',
    pcCapacity)


end

%% Functions
function [ C ] = nyquist(B, M)
C = (2 .* B) * log2(M'); % calculate the Nyquist capacity
end

function [ C ] = shannon(B, SNR)
```

```
C = log2(1 + SNR) * B'; % calculate the Shannon capacity
end

function [ ratio ] = idB(dB)
ratio = 10.^(dB./10); % turn dB into a ratio
end

function [ M ] = rev_nyquist(C_s, B)
preM = floor(2.^(C_s ./ (2 .* repmat(B,[1,4])'))); % get actual number of
    symbols desired, rounded down
M = 2.^floor(log2(preM)); % get an available (power of 2) modulation
    format
end
```