numBits = 20000;

modOrder = 16; % for 16-QAM

bitsPerSymbol = log2(modOrder); % modOrder = 2^bitsPerSymbol

txFilt = comm.RaisedCosineTransmitFilter;

rxFilt = comm.RaisedCosineReceiveFilter;

srcBits = randi([0,1],numBits,1);

modOut = qammod(srcBits,modOrder,"InputType","bit","UnitAveragePower",true);

txFiltOut = txFilt(modOut);

spacing = zeros(7,1);

mpChan = [0.8; spacing; -0.5; spacing; 0.34]

stem(mpChan)

mpChanOut = filter(mpChan,1,txFiltOut);

if exist("mpChanOut","var") % code runs after you complete Task 3

SNR = 15; % dB

chanOut = awgn(mpChanOut,SNR,"measured");

rxFiltOut = rxFilt(chanOut);

scatterplot(rxFiltOut)

title("Receive Filter Output")

demodOut = qamdemod(rxFiltOut,modOrder,"OutputType","bit","UnitAveragePower",true);

% Calculate the BER

delayInSymbols = txFilt.FilterSpanInSymbols/2 + rxFilt.FilterSpanInSymbols/2;

delayInBits = delayInSymbols \* bitsPerSymbol;

srcAligned = srcBits(1:(end-delayInBits));

demodAligned = demodOut((delayInBits+1):end);

numBitErrors = nnz(srcAligned~=demodAligned)

BER = numBitErrors/length(srcAligned)

end

specAn = dsp.SpectrumAnalyzer("NumInputPorts",2, ...

"SpectralAverages",50,...

"ShowLegend",true);

specAn(txFiltOut,chanOut)