Set-up as a function

function [errorStats, errorStats\_ofdm]=qam\_OFDM\_fadechannel\_func(M,numSC,cpLen,sigma)

% M = 16; %Modulation alphabet

k = log2(M); %Bits/Symbol

% numSC = 64; %Number of OFDM subcarriers

% cpLen = 16; %OFDM Cylcic prefix length

Creation of system objects for simulation: 16-QAM Modulator/Demodulator, OFDM Modulator/Demodulator, Fade channel (for just 16-QAM and for OFDM), error rate calculator.

qamMod = comm.RectangularQAMModulator(M, "BitInput",true); %default gray-coding

qamDemod = comm.RectangularQAMDemodulator(M,"BitOutput",true);

ofdmMod = comm.OFDMModulator("FFTLength",numSC,"CyclicPrefixLength",cpLen);

ofdmDemod = comm.OFDMDemodulator("FFTLength", numSC,"CyclicPrefixLength", cpLen);

ofdmDims = info(ofdmMod);

numDC = ofdmDims.DataInputSize(1); %Number of Data Carrier symbols

frameSize = [k\*numDC 1]; %Frame size in bits

%Channel Parameters

%ofdm

% sigma = 0.1;

a\_mag\_ofdm = raylrnd(sigma, [1, numSC+cpLen]); %create the complex coefficient 'a' using a rayleigh distribution

theta\_ofdm = (2\*pi).\*rand([1, numSC+cpLen]); %create theta based on uniform distribution

expTheta\_ofdm = exp(-1j.\*theta\_ofdm);

a\_vec\_ofdm = a\_mag\_ofdm .\* expTheta\_ofdm; % enumerate the a values in a list

t\_symbol = 1e-6; % enumerate the sample periods

t\_ofdm= (t\_symbol: t\_symbol:(numSC+cpLen)\*t\_symbol);

%no ofdm

a\_mag = raylrnd(sigma, [1, frameSize(1)/4]); %create the complex coefficient 'a' using a rayleigh distribution

theta = (2\*pi).\*rand([1, frameSize(1)/4]); %create theta based on uniform distribution

expTheta = exp(-1j.\*theta);

a\_vec = a\_mag .\* expTheta; % enumerate the a values in a list

t\_symbol = 1e-6; % enumerate the sample periods

t = (t\_symbol: t\_symbol:(frameSize(1)/4)\*t\_symbol);

errorRate = comm.ErrorRate("ResetInputPort", true);

Transmission-receipt Execution and bit error calculation

dataIn = randi([0,1], frameSize); %input data creation to match ofdm frame size

qamTx = qamMod(dataIn); %16-QAM Signal generation/transmission w/out OFDM

txSig\_ofdm = ofdmMod(qamTx); %OFDM Transmission

rxSig = fadeChannel(qamTx', a\_vec); %16-QAM signal through fade channel

rxSig\_ofdm = fadeChannel(txSig\_ofdm', a\_vec\_ofdm); %OFDM Signal through fade channel

qamRx\_ofdm = ofdmDemod(rxSig\_ofdm'); %Demodulating OFDM to 16-QAM

dataOut = qamDemod(rxSig'); %receipt of 16-QAM signal

dataOut\_ofdm = qamDemod(qamRx\_ofdm); %receipt of 16-QAM (OFDM) signal

errorStats = errorRate(dataIn, dataOut, 0); %16-QAM Errors --> 1x3 array [BER, total errors, total bits]

errorStats\_ofdm = errorRate(dataIn, dataOut\_ofdm, 0); %16-QAM OFDM Errors

end