05.11.2018

# MAC Protocol Simulation in Undersea Environments

Wireless Communications, DAT610

Henrik Mjaaland and Jerzy Jedruszek
UNIVERSITY OF STAVANGER

# *Abstract*

The intention of this paper is to find a suitable MAC (media/medium access control) Protocol for undersea environments, and to prove that the chosen MAC protocol really is fitting for undersea environments by surveying literature and simulating undersea communications. The simulations were performed using CP-OFDM (Cyclic Prefix Orthogonal Frequency Division Multiplexing) and generating an UWAC (underwater acoustic channel) in MATLAB.

# Contents

# *1.0 Wireless Communication in Undersea Environments*

### *1.1 Applications*

Where the reach of wired networks stops, one must use wireless networks, usually in the form of sensors or autonomous underwater vehicles (UAVs). The data rates delivered by wired networks underwater are affected by the immense pressure that is at the bottom of the ocean, and the cables are vulnerable to many threats underwater as rock slides, earthquakes or animal attacks. Also, most of the earth is submerged in water and it is beneficial to develop underwater communication technologies.

Wireless communication has many uses for underwater environments. It is used for example in the oil industry, wireless sensor networks (gathering of scientific data and pollution monitoring), surveillance for security, and even exploration of the ocean. Wireless sensor networks have not until recently been used for home appliance, work etc. Previously it has mostly been developed and applied by the US military. Also, it allows communication from places that are out of reach for cables. For us in Norway the most important use is probably in the oil industry where it is used for UAVs that map out the ocean floor before operations.

### *1.2 Constraints*

Attaining high data rates in aquatic environments is a difficult challenge because wireless communication underwater has several limitations. Obviously, there is a reflection in the surface and bottom of the water and there is a lot of noise underwater and signals underwater has a high absorption rate. This means that underwater signals quickly attenuate and disperse, thus reducing the communication range underwater. There are constrained power resources and bandwidth in undersea environments. Also, today's mathematical representations of underwater environments are still not very advanced.

Wireless underwater communication is characterized by multipath propagation (Signals travel along multiple separate paths) and attenuation. Multipath propagation can cause ISI (inter-symbol-interference), which is when a symbol (group of multiple subcarriers in the frequency domain) interferes with a previous symbol, see Figure 1 below. An example of how multipath propagation can cause ISI is if a signal travels along multiple paths, and the symbol that is transmitted first travels along a slower path than a symbol that is transmitted afterwards and the receiver then receives the two symbols at the same time. However, since OFDM splits a carrier into subcarriers, the symbol time is expanded, which gives lower ISI.

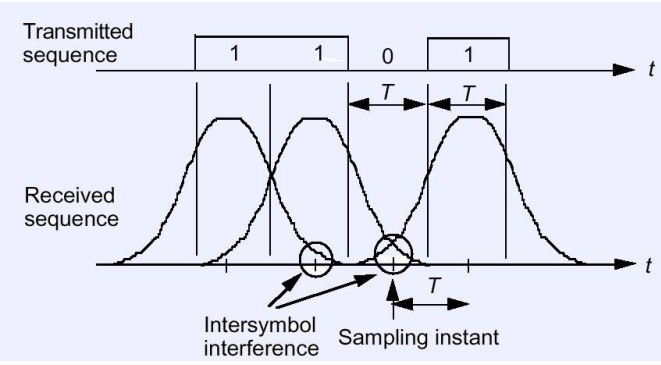


*Figure 1: Intersymbol Interference [http://technews365.info/intersymbol-interference/#prettyPhoto/0/]*

The random motions of sea surfaces and currents cause Doppler Shift. Doppler shift is the change in frequency that occurs when a receiver is moving relative to the transmitter while a signal is being transmitted[1]. Doppler shift can be avoided by using algorithms that estimate the eventual damage caused by the Doppler shift and then correcting it. Frequency offsets caused by Doppler shift can cause inter carrier interference (ICI) for OFDM (Orthogonality Frequency Division Multiplexing) MAC protocols, which is when the subcarriers lose their orthogonality. An example of ICI is shown in figure 2 below.

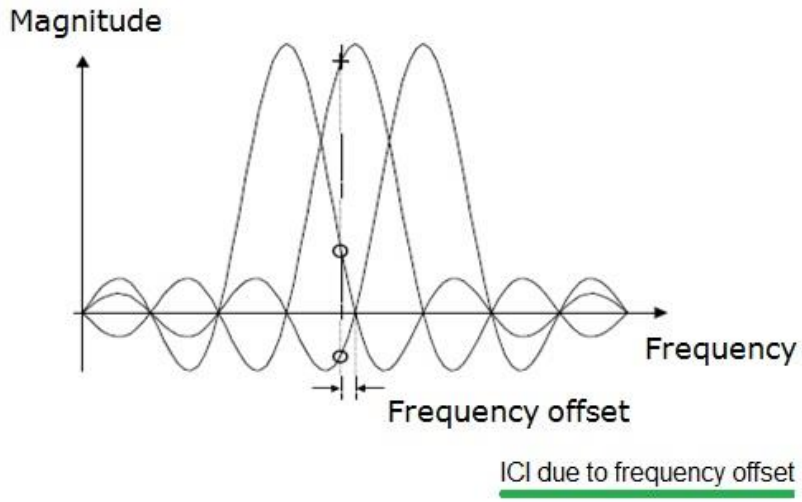


*Figure 2: Inter Carrier Interference [http://www.rfwireless-world.com/Terminology/ISI-vs-ICI.html]*

# *2.0 Signal Types*

In a *communication system*, a *transmitter* encodes a *message* to a signal, which is carried to a *receiver* by the communications *channel[2]*. For underwater environments, the selection of wave types boils down to radio frequency, optical waves or acoustic waves.

### *2.1 RF*

Radio Frequency (RF) transmits signals in the form of electromagnetic waves (radio waves), and the signals are digital. RF offers the best data rates because electromagnetic waves travel at the speed of light. However, it is not preferred in underwater environments because of high absorption, and thus high attenuation.

### *2.2 Optical Waves*

In optical networks, electromagnetic signals are encoded on light to transmit information (the signals are transmitted in the optical spectrum). As electromagnetic waves, optical waves also offer high data rates, but also high attenuation. It is also affected by scattering (when waves deviate from the trajectory).

### *2.3 Acoustic Waves*

Acoustic waves are sound waves, which has a low absorption rate. This makes acoustic waves ideal in undersea environments because acoustic waves can travel further than radio waves. Even though acoustic waves travel by the speed of sound and are slower than RF/optical waves, they can travel a lot farther and have low attenuation, and this is why we chose to use an UWAC (underwater acoustic channel) which uses acoustic waves as the name insinuates to simulate underwater wireless communication.

# *3.0 Cyclic Prefix Orthogonal Frequency Division Multiplexing (CP-OFDM)*

## *3.1 Orthogonal Frequency Division Multiplexing (OFDM)*

### *3.1.1 Presentation*

OFDM is a modulation technique that modulates a given signal over multiple orthogonal carriers (hence the name). Some of the benefits of OFDM is robustness towards multipath interference, spectrum-efficiency and simple noise filtration.

What separates OFDM from FDM is that in OFDM the carriers are closely packed, and they are orthogonal to each other. Orthogonal means that at one carrier's peak, all the other carriers are nulled out. This is why OFDM gives higher data rates than FDM, and also a higher bandwidth usage.

Multi carrier modulation methods splits up a data stream into multiple parallel streams that are then modulated on each their carrier. Guard bands are added to prevent ISI. This leads to lower spectral efficiency.

### *3.1.2 Implementation*

First, a given data stream of n bits is converted from serial to parallel format. Then the data stream is modulated using a modulation method like for example QPSK (Quadrature Phase Shift Keying) or QAM (Quadrature Amplitude Modulation). In our implementation we used QAM because it uses bandwidth efficiently. The result of the modulation is a vector. Then IFFT (Inverse Fast Fourier Transform) is calculated using the vector as input. The resulting output is an OFDM consisting of n channels that each consist of a modulated symbol. These symbols are then converted back to serial format. The OFDM is then processed further before finally being transmitted (guard interval insertion, low-pass filter to avoid aliasing etc.). On the receiver side the FFT (Fast Fourier Transform) algorithm is used. The implementation is demonstrated in figure 3 below.

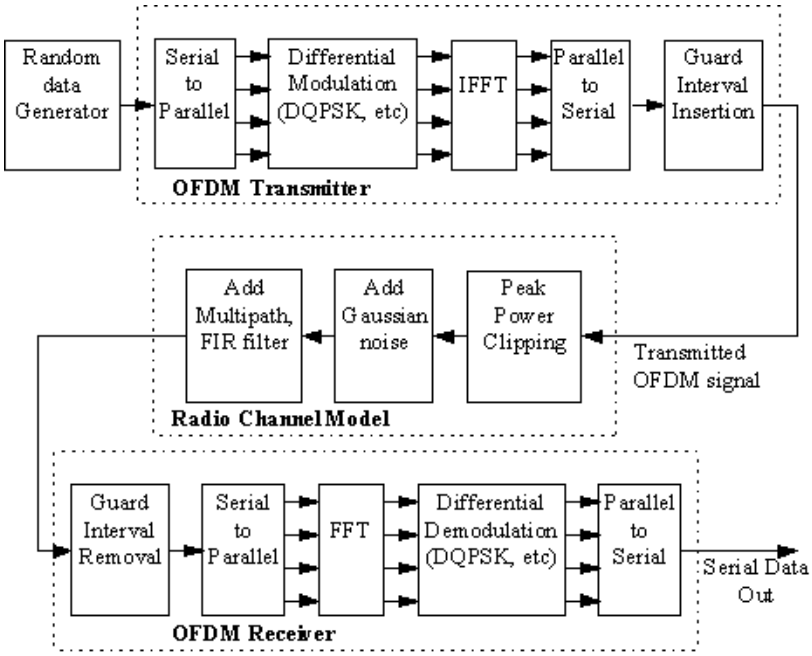


*Figure 3: OFDM Implementation [http://www.skydsp.com/publications/4thyrthesis/chapter2.htm]*

## *3.2 Cyclic Prefix (CP)*

A cyclic prefix is a copy of a part of the end of a symbol that is pre-appended to all the transmitted OFDM symbols to prevent inter-symbol-interference. An example of cyclic prefix is demonstrated in figure 4 below.

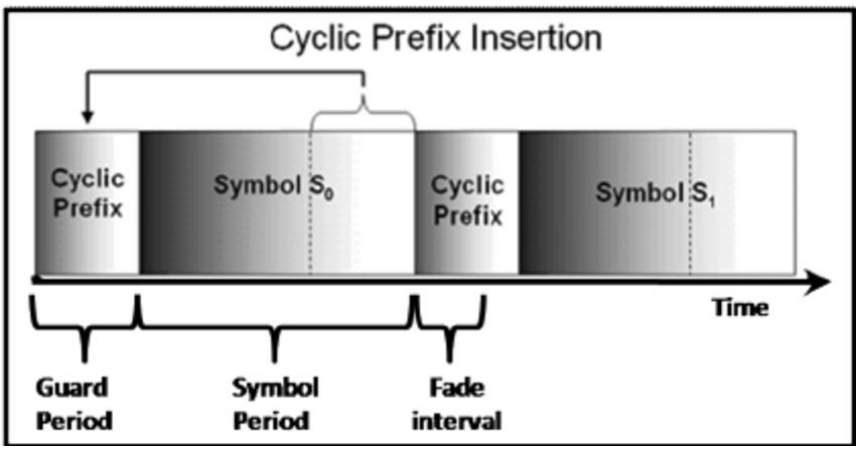


*Figure 4: Cyclic Prefix [https://www.researchgate.net/figure/Insertion-of-cyclic-prefix-in-guard-interval_fig1_220902525]*

### *3.3 Why CP-OFDM?*

There are several reasons for using OFDM in UWACs. First of all, it achieves high data rates in UWACs. It gives robust communication in large delay spread channels because it uses flat frequency narrowband channels. The computation algorithms it uses (IFFT and FFT) are computationally efficient. Also, it allows overlapping of subcarriers which gives efficient use of bandwidth.

According to Ericsson Research[3], CP-OFDM was the best candidate for NR (New Radio) which is an integral part of the 5G standardization. Ericsson Research also claims that CP-OFDM ranks best on the performance indicators that matter most – compatibility with multi-antenna technologies, high spectral efficiency, and low implementation complexity (FFT and IFFT is used for modulation and demodulation and we have a good understanding of OFDM because it has been around since 4G). Furthermore, Ericsson Research says that CP-OFDM gives low latency and that it is robust to oscillator phase noise and the Doppler Shift.

Another important reason to apply CP-OFDM for UWACs is because as stated in sub-chapter 3.2, cyclic prefix reduces ISI, which is a big problem in underwater acoustic channels. However, the reduction in ISI comes at a cost, namely a reduction in data rates. Also, the length of the cyclic prefix must be longer than the impulse response of the RRC (cosine-raised filter, which is ideal for reducing ISI) filter, otherwise there will still be ISI.

The Doppler Shift causes frequency offset which in turn causes ICI. Cyclic prefix also causes ICI because the delay spread exceeds the CP interval (guard interval). ICI caused by the latter can be reduced by correcting the sub-carrier spacing. SNR (Signal to Noise Ratio) increases as ICI is reduced.

Cons with using OFDM in UWACs on the other hand is that it is more sensitive to carrier frequency offset than single carrier systems. It also gives a high peak to average power ratio (PAPR). OFDM gives high peak-to-average power ratio (PAPR) because, in the time domain, a multicarrier signal is the sum of many narrowband signals. This means that the difference between the peak of the signal and the average value is in general bigger than for a single carrier signal, depending on the number of sub-carriers used. This leads to inefficient use of the power amplifiers.

# *4.0 Simulations and Results*

## *4.1 Choice of IDE*

For our implementations we decided it was best to use MATLAB. One of the main reasons we chose MATLAB for the experiments is because OFDM involves a lot of operations on matrices, and MATLAB is designed to operate on matrices (all MATLAB variables are two-dimensional vectors or matrices). MATLAB also has a lot of toolboxes (libraries) with built-in functions. For example, we used the 'raylrnd' function from the statistics toolbox and 'fftfilt' from the signal processing toolbox. You can open these toolboxes inside MATLAB to view the code and even edit them to as you see fit. Other benefits with using MATLAB is that you can for example execute portions of code at a time, and the 'help' function is also useful for information on MATLAB functions.

## *4.2 Test Results*

Monte Carlo simulations of CP-ODFM using 10 multi-path channels with 5 paths each and a modulation rate of 10 000 gave the following Bit Error Rate (BER) as a function of SNR (see figure 5 below). As you can see from the figure, when the SNR decreases, the bit error rate drastically increases until it stagnates (looking from right to left), and that shows how important it is to keep interferences (extrinsic noise) as low as possible.

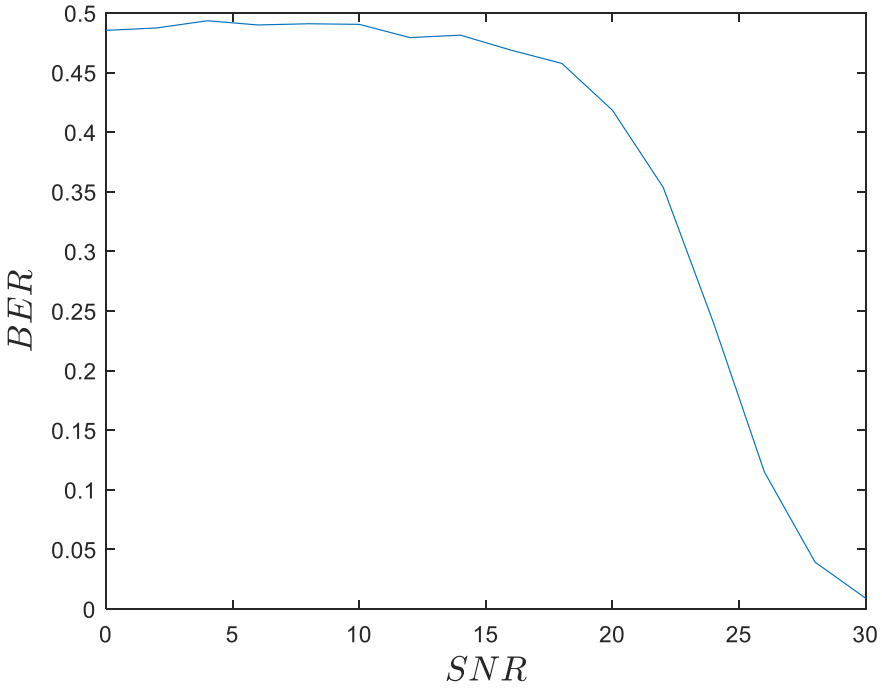


*Figure 5: Bit Error Rate (BER) as a Function of SNR*

Figure 6 displays impulse response as a function of time. The peaks in the figure is the summed signal of all the multipath components arriving at time t. The graph is sparse because the impulse response comes in pulses which is caused by multipath reflection (wireless channels are multipath channels).

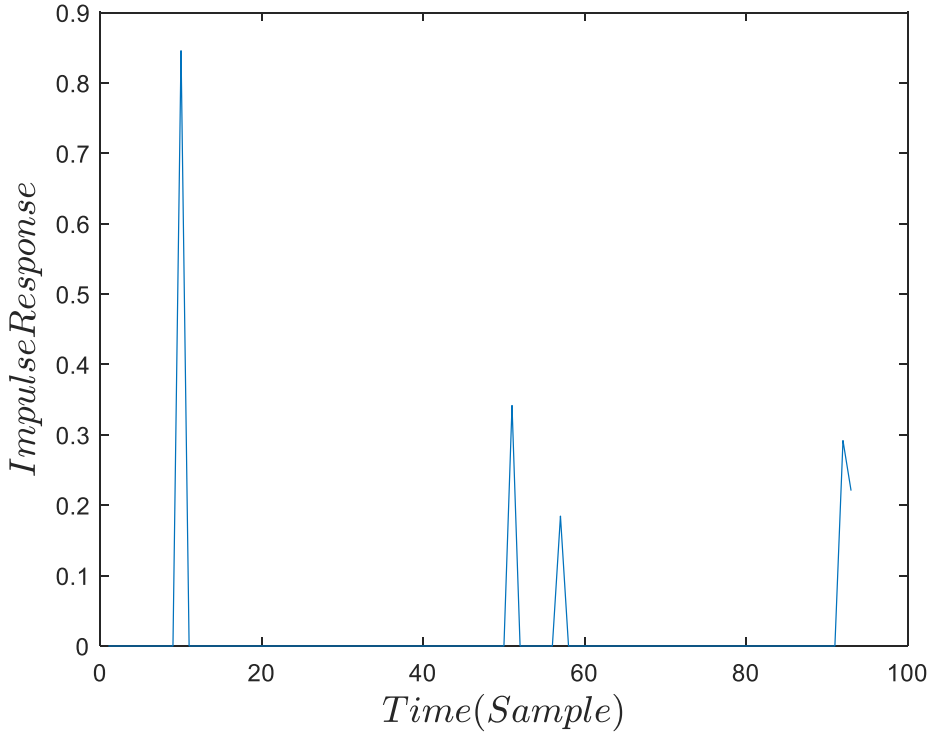


*Figure 6: Impulse Response as a Function of Time*

Figure 7 displays the gain for the 5 paths as a function of time (delay). The gain is the ratio of the spread bandwidth and the baseband bandwidth in decibels (dB). Gain can be expressed by the following formula, gain = bw/signal, where bw is the spread bandwidth. This means that the impulse response is proportional to the gain, thus we can see that figure 7 verifies the data from figure 6.

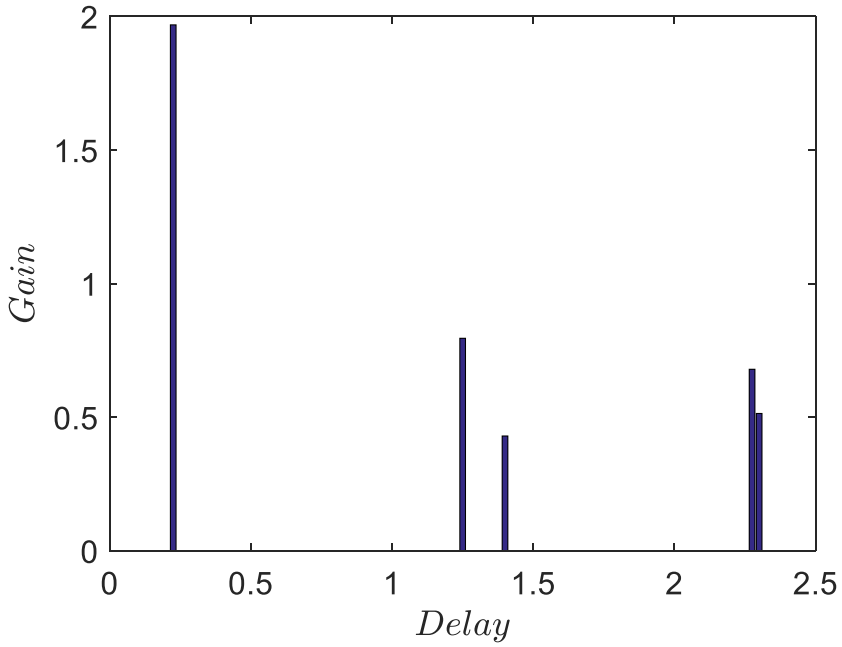


*Figure 7: Gain as a Function of Delay*

## *5.0 Future Work*

Our work can be improved by optimizing PAPR reduction or improving frequency localization (OFDM is sensitive to frequency offsets due to FFT leakage) on the transmitter side. There are many ways to reduce the PAPR. One can for example implement companding or clipping and filtering. Clipping is the easiest way, although not the best. Clipping limits a signal once it exceeds a threshold[4]. As the name insinuates, companding is a technique where signals are first compressed and then expanded. One of the best ways of reducing PAPR might be peak windowing. Peak windowing multiplies high peak values with a given window function. A window function is a function that nullifies all values outside a specific region of interest. Frequency localization can be improved by using such window functions (with smooth edges to obtain a continuous waveform and thus prevent FFT leakage).

## *6.0 Conclusion*

For this project we chose to use an UWAC (underwater acoustic channel) to simulate underwater wireless communication. The problems correlated with this type of communication are mostly resolved by Cyclic Prefix Orthogonal Frequency Division Multiplexing. The main reasons to use CP-OFDM for undersea environments is that it has high spectral efficiency, reduces ISI caused by multipath propagation (even though cyclic prefix increases ICI), while at the same time giving low BER. An UWAC is suited for simulation of underwater wireless communication because acoustic waves have a low absorption rate. In order to fix some of the remaining distortion issues we used a method called Windowing which is a method that nullifies values outside a given region of interest.

# References

*[1] Doppler Shift*. (2018, 10 25). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Doppler_effect

*[2] Signal*. (2018, 09 24). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Signal

*[3] In race to 5G CP-OFDM triumphs*. (2018, 09 24). Retrieved from ericsson: https://www.ericsson.com/research-blog/in-race-to-5g-cp-ofdm-triumphs/

[4] Clipping. (2018, 10 25). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Clipping_(signal_processing)

# *Appendix A*

### *A.1 User Manual*

We used global variables to store the channel parameters, which means that you must run UWACmain.m which generates the underwater acoustic channel and the plots in figure 6 and 7 below first, and then CpOfdm.m which simulates CP-OFDM and plots BER as a function of SNR (see figure 5 in chapter 4.2)

### *A.2 Code for 'CpOfdm.m'*

```
clear;close all;clc;


% ------ Remember to run UWACmain.m before running this file, in order to
% generate the global UWAC parameters.


global uwac_params;
disp(uwac_params);

SNR  = 0:2:30; %Signal to noise ratio in decibels
SNRlen = length(SNR); %size of SNR

bitsPerSymbol = 4; %16QAM = 2^4QAM -> 4 bits per symbol.
constellationSize = 2^bitsPerSymbol;


nGB = [16; 16]; %number of guard band subcarriers allocated to the left and right guard bands.
nFFT = 1024; %total number of subcarriers.
nBlock = nFFT - (sum(nGB)); %Subcarriers per block.


%Parameters for encoding:
K = 3; %Constraint Length
tableLength = 5*K;
x = 5;
k = 7;
codeRate = 0.5; %Convolutional Code Rate.
trellis = poly2trellis(K,[x k]);

generatedBits = bitsPerSymbol*(nBlock*codeRate);

interleaver = randi([1 1e6],1,1);

symbolRate = 24*1000;
symbolPeriod = 1/symbolRate;
BW = 5*1000;

%Convert sampled digital signal to a that of a higher sampling rate.
UpSampling = ceil(symbolRate/BW);
samplingPeriod = symbolPeriod/UpSampling;
samplingFrequency = 1/samplingPeriod;

carrierFrequency = 15000;


% Low-Pass-Filter (LPF)
filterOrder = 1000; %Maximum amount of delay to calculate output
cutoffFrequency = 0.5*samplingFrequency/UpSampling; %frequencies below pass, others attenuate.
angularFrequency = 2*pi*cutoffFrequency/samplingFrequency; %rotation rate
x = 1:filterOrder/2;
tempFilter    = angularFrequency/pi; %convolution
tempFilter2 = sin(angularFrequency*x)./(pi*x); %convolution's impulse response
LPF = [fliplr(tempFilter2) tempFilter tempFilter2]; %Low-Pass-Filter
LPF_Hamming = LPF.*hamming(filterOrder+1).'; %Hamming window (0 outside chosen interval).

if((mod(filterOrder,2) == 0))
    transientDelay = (filterOrder)/2; %time it takes response to reach half the final value.
else
    transientDelay = (1+filterOrder)/2;
```

```
end

nMC =  1; %Monte carlo iterations
BER = zeros(nMC,SNRlen); %Bit Error Rate init




for i = 1:nMC

    y = uwac_params{i,1}; %Set Channel
    %minimum CP = length of channel.
    CyclicPrefixLength = ceil(size(y,1)/UpSampling) - 1;

    bits = randi([0 1],generatedBits,1);

    RCPMat = [zeros(nFFT,CyclicPrefixLength), eye(nFFT)]; %Receiver's matrix.

    TCode = convenc(bits,trellis); %transmitters code.

    Tintrlv = randintrlv(TCode,interleaver); %interleave/randomly rearrange TCode.

    %Modulate interleaved code:
    Tmod = qammod(Tintrlv,constellationSize,'gray','InputType','bit','UnitAveragePower',true);


    symbols = [zeros(nGB(1),1); Tmod; zeros(nGB(2),1)];
    T_IFFT = sqrt(nBlock)*ifft(ifftshift(symbols)); %OFDM Modulation - IFFT

    T_CP = [T_IFFT(end-CyclicPrefixLength+1:end,:); T_IFFT].'; %prepend CP

    T_CP_Up = upsample(T_CP,UpSampling); %Convert from digital to analog

    T_Filter = fftfilt(LPF_Hamming,T_CP_Up);


    timeVector = samplingPeriod*(0:(size(T_Filter,2) - 1));
    %carrier at frequency = carrierfrequency:
    carrier = exp(2*1i*pi*carrierFrequency*timeVector);
    T_Pass = real(carrier.*T_Filter); % Convert baseband signal to passband signal
    T_Pwr = norm(T_Pass)^2/length(T_Pass); %Mean transmission power

    R_Ch = fftfilt(y(:,1),T_Pass); %receiver channel

    %receiver side

    for j = 1:SNRlen

        %AWGN (Add White Gaussian Noise)
        noiseVector  = randn(1,size(R_Ch,2)); %noise
        % Average noise power
        avgPower = norm(noiseVector)^2/length(noiseVector);
        R_Ch_AWGN = R_Ch + sqrt(((T_Pwr/avgPower))*(10^(-SNR(j)/10)))*noiseVector;

        R_Baseband = (R_Ch_AWGN.*conj(carrier)).'; %Demodulate
        R_Temp = fftfilt(LPF_Hamming,[R_Baseband; zeros(filterOrder-1,1)]);
        R_Temp2 = R_Temp(transientDelay+1:end-transientDelay);
        R_DownSample = downsample(R_Temp2,UpSampling);%Reverse Pulse Shaping

        %Estimating channel information for OFDM:
        pilotVector = conj(symbols)./(abs(symbols).^2 + 10^(-SNR(j)/10));
        estVector = fftshift(fft(RCPMat*R_DownSample,nFFT)).*pilotVector/sqrt(nFFT);
        %Equalization matrix:
        estMat = diag(conj(estVector)./(abs(estVector).^2 + 10^(-SNR(j)/10)));

        R_EstSymbTemp = estMat*fftshift(fft(RCPMat*R_DownSample,nFFT))/sqrt(nFFT);
        R_EstSymb = R_EstSymbTemp(nGB(1)+1:end-nGB(2)); %Remove null subcarries

        %Digital demodulation:
        R_Demodulated = qamdemod(R_EstSymb,constellationSize,'gray','OutputType',...
            'approxllr','UnitAveragePower',true,'NoiseVariance',avgPower);

        % Deinterleaving
        R_Deintrlv = randdeintrlv(R_Demodulated,interleaver);
```

```
        %Convolution decoding using the Viterbi algorithm
        estBits = vitdec(R_Deintrlv,trellis,tableLength,'cont','unquant');

        %Estimated Bit Error rate:
        BER(i,j) = biterr(bits(1:end-tableLength),estBits(tableLength+1:end))/(generatedBits);
    end
end

avgBER = mean(BER,1);
disp(avgBER);
disp(SNR);
figure(1);
%Signal to noise ratio and BER
plot(SNR,avgBER);
name = 'Sans Serif';
size = 15;
xlabel('$SNR$','fontname',name,'fontsize',size,'interpreter','latex');
ylabel('$BER$','fontname',name,'fontsize',size,'interpreter','latex');
```

### *A.3 Code for 'UWACmain.m'*

```
clc;clear;close all;

nCh = 100; %Number of multipath channels
nP = 5; %Number of paths per channel
modRate = 10000; %Modulation Rate
period = 1/modRate;
samplingRate = 4;
sPeriod = period/samplingRate;
cp = 32;

i = 1;
global uwac_params;
uwac_params = cell(nCh,5);

while (i <= nCh)
    [impRes, prop_delay,processing_gain,ctr,dtr] = UWAC(sPeriod,'path_count',nP);
    if ((isminphase(impRes(:,1)) ~= 0 && size(impRes(:,1),1) < (cp+1)*samplingRate))
        uwac_params{i,1} = impRes;
        uwac_params{i,2} = processing_gain;
        uwac_params{i,3} = prop_delay;
        uwac_params{i,4} = ctr;
        uwac_params{i,5} = dtr;
        disp(i);
        i = i+1;
    end
end

MCSimulations = 5;
UWACplot(MCSimulations, uwac_params);

function [] = UWACplot(numSims, uwac_params)
MCsimulations = numSims;
Hn = uwac_params{MCsimulations,1}; %Finite impulse response filter (tap vector = H0,H1,..,Hn)
processingGain  = uwac_params{MCsimulations,2};
propagationDelay = uwac_params{MCsimulations,3};

figure(1);
name = 'Sans Serif';
size = 15;
plot(Hn);
xlabel('$Time (Sample)$','fontname',name,'fontsize',size,'interpreter','latex');
ylabel('$Impulse Response$','fontname',name,'fontsize',size,'interpreter','latex');

figure(2);
%processing gain y and propagation delay x
%the gains in the plot are the average gains for all the paths):
bar(propagationDelay(:,1)*1000,processingGain);
xlabel('$Delay$','fontname',name,'fontsize',size,'interpreter','latex');
ylabel('$Gain$','fontname',name,'fontsize',size,'interpreter','latex');
set(gca,'fontname',name,'fontsize',size);
end

function [impulseResponse,lag,taps,varargout] = UWAC(period,varargin)
```

```
p = inputParser;
addParameter(p,'guardInterval',0.0246);
addParameter(p,'path_count',15);
addParameter(p,'lag_mean',0.001);
addParameter(p,'powerChange',20);
addParameter(p,'velocity',20);
addRequired(p,'period',@isnumeric);
parse(p, period, varargin{:});

period = p.Results.period;
path_count = p.Results.path_count;
lag_mean = p.Results.lag_mean;
powerChange = p.Results.powerChange;
guardInterval = p.Results.guardInterval;
velocity = p.Results.velocity;

%Doppler frequency:
c = 1500; %velocity of the waves in the medium
f = velocity/c;
f_n = length(f);
[ctr,dtr] = rat(f+1);
varargout{1} = ctr;
varargout{2} = dtr;

%Rayleigh Multipath:
pd = makedist('exponential','mu',lag_mean);
R     = random(pd,path_count,1);
Y = ceil(R/period);
lag_location = cumsum(Y);
lag = period*(lag_location-1);
a = log(10^(powerChange/10))/guardInterval;
avgPower = exp(lag*-a);
taps = raylrnd(avgPower*sqrt(2/pi));

B = repmat(f.',path_count,1);
temp = repmat(lag_location,1,f_n);
new_lag_loc = ceil(temp./(1 + B));
impulseResponse = zeros(max(max(new_lag_loc)),f_n);

for i = 1:f_n
    impulseResponse(new_lag_loc(:,i),i) = taps;
    impulseResponse(:,i) = impulseResponse(:,i)/norm(impulseResponse(:,i));
end
lag = (new_lag_loc - 1)*period;
end
```