# MAHA BARATHI ENGINEERING COLLEGE

**NH-79, SALEM-CHENNAI HIGHWAY, A.VASUDEVANUR, CHINNASALEM (TK), KALLAKURICHI (DT) 606 201**.
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai
Accredited by NAAC and Recognized under section 2(f) & 12(B) status of UGC, New Delhi

www.mbec.ac.in | Ph: 04151-256333, 257333 | E-mail: mbec123@gmail.com

## DEPARTMENT OF ELECTRONICS AND COMMUNICATION

## ENGINEERING

### EC3501-WIRELESS COMMUNICATION LABORATORY

### III Year/ V Semester B.E ECE

### Regulation 2021

### (As Per Anna University, Chennai syllabus)

| Prepared By | Verified By | Approved By |
|---|---|---|
| Staff I/C | (HoD/ECE) | (Principal) |

## SYLLABUS

**PRACTICAL EXERCISES:** **30 PERIODS**

1. Modeling of wireless communication systems using Matlab (Two ray channel and Okumura –Hata model).
2. Modeling and simulation of Multipath fading channel.
3. Design, analyze and test Wireless standards and evaluate the performance measurements suchas BER, PER, BLER, throughput, capacity, ACLR, EVM for 4G and 5G using Matlab.
4. Modulation: Spread Spectrum – DSSS Modulation & Demodulation.
5. Wireless Channel equalization: Zero-Forcing Equalizer (ZFE), MMSE Equalizer (MMSEE), Adaptive Equalizer (ADE), Decision Feedback Equalizer (DFE).
6. Modeling and simulation of TDMA, FDMA and CDMA for wireless communication.

# INDEX

| S.NO. | DATE | TITLE OF THE EXPERIMENT | PAGE | MARK | SIGNATURE |
|---|---|---|---|---|---|
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

| Expt. No.:1 | Modeling of wireless communication systems using Matlab (Two ray channel and Okumura – Hata model) |
|---|---|
| Date: | |

**AIM**:

To model and simulate wireless communication system for two ray channel and okumura hata modelusing Matlab.

**Software Required:**

Matlab version 2014a

**THEORY:**

The two-ray ground-reflection model is a multipath radio propagation model which predicts the path losses between a transmitting antenna and a receiving antenna when they are in line of sight (LOS). Generally, the two antenna each have different height. The received signal having two components, the LOS component and the reflection component formed predominantly by a single ground reflected wave. When the distance between antennas is less than the transmitting antenna height, two waves are added constructively to yield bigger power. As distance increases, these waves add up constructively and destructively, giving regions of up-fade and down-fade. As the distance increases beyond the critical distance or first Fresnel zone, the power drops proportionally to an inverse of fourth power of an approximation to critical distance may be obtained by setting $\Delta\varphi$ to $\pi$ as the critical distance to a local maximum.

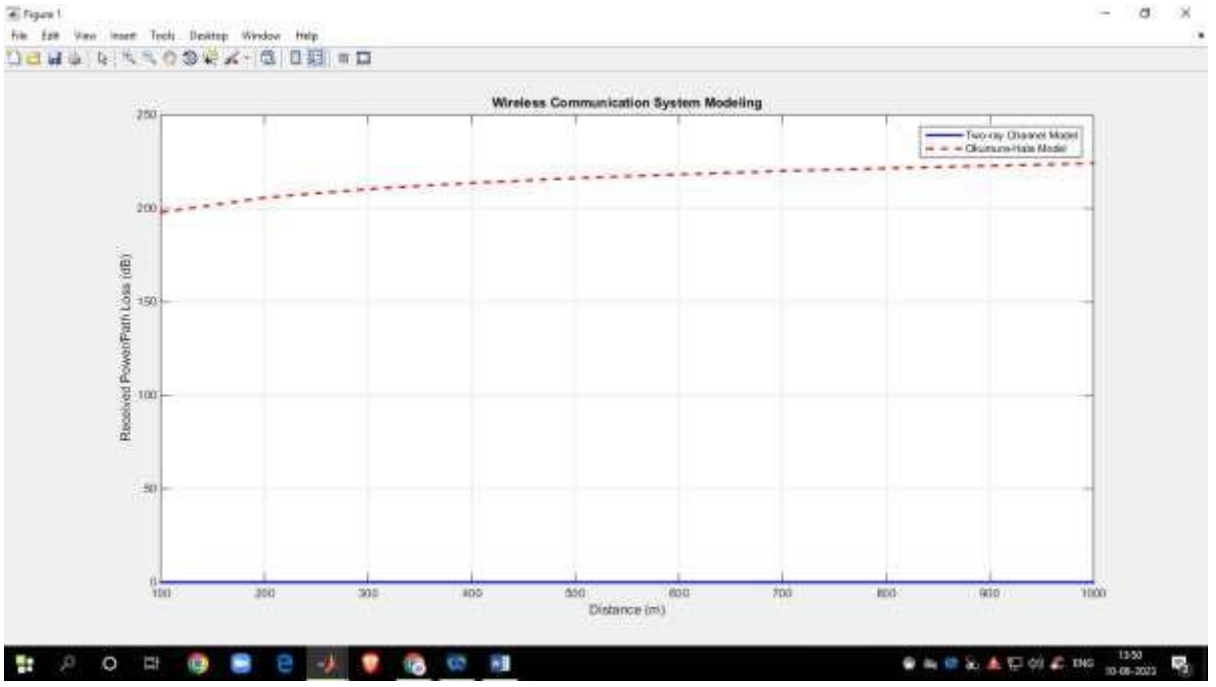
**PROCEDURE:**

1. Start the MATLAB program.
2. Open new M-file.
3. Type the program.
4. Save in current directory.
5. Compile and Run the program.
6. If any error occurs in the program correct the error and run it again.
7. For the output see command window\ Figure window.
8. Stop the program.

**CODING:**

```
>> % Wireless Communication System Modeling
% Two-ray channel model and Okumura-Hata model

% System Parameters
frequency = 900e6;       % Frequency in Hz
transmitterHeight = 50; % Transmitter height in  meters
receiverHeight = 10;  % Receiver height in meters
distance = 100:100:1000; % Distance between transmitter and receiver in meters

% Two-ray Channel Model
Pt = 1;              % Transmitted power in Watts
Gt = 1;              % Transmitter antenna gain
Gr = 1;              % Receiver antenna gain
 L = 1;              % System loss

% Calculate received power using Two-ray channel model
Pr_two_ray = Pt * (Gt * Gr * (transmitterHeight * receiverHeight)^2) / (distance^4 * L);

% Okumura-Hata Model
A = 69.55;           % Model parameter
B = 26.16;           % Model parameter
C = 13.82;% Model parameter
D = 44.9;            % Model parameter
X = 6.55;            % Model parameter
hb = 30;             % Base station height in meters
% Calculate path loss using Okumura-Hata model
PL_okumura_hata = A + B * log10(distance) + C * log10(frequency/1e6) + D - X  * log10 (hb);

% Plotting
figure;
plot(distance, Pr_two_ray, 'b-', 'LineWidth',2);
hold on;
plot(distance, PL_okumura_hata, 'r--','LineWidth', 2);
xlabel('Distance (m)');
ylabel('Received Power/Path Loss (dB)');
legend('Two-ray Channel Model', 'Okumura-Hata Model');
title('Wireless Communication System Modeling');
grid on;
```

**Output:**




**RESULT :**

Thus designing a model of wireless communication systems using Matlab(Two ray channel and Okumura –Hata model) is achieved

| Expt. No.:2 | Modeling and simulation of Multipath fading channel |
|---|---|
| Date: | |

**AIM**:

To design a Model and simulation of Multipath fading channel.

**SOFTWARE REQUIRED:**

Matlab version 2014

**THEORY:**

In wireless communication**,** fading is a phenomenon in which the strength and quality of a radio signal fluctuate over time and distance. Fading is caused by a variety of factors, including multipath propagation, atmospheric conditions, and the movement of objects in the transmission path. Fading can have a significant impact on the performance of wireless communication systems,particularly those that operate in high-frequency bands.

- Multipath delay spread is a type of small-scale fading that occurs when a transmitted signaltakes multiple paths to reach the receiver.
- The different components of the signal can arrive at the receiver at different times, causinginterference and rapid variations in signal amplitude and phase.
- Multipath delay spread can cause Inter-Symbol Interference (ISI), where symbols in thetransmitted signal overlap and interfere with each other, leading to errors in the received signal.
- The root means square (RMS) delay spread is a measure of the dispersion of the signal anddetermines the frequency-selective characteristics of the channel.
- A higher RMS delay spread indicates a more frequency-selective channel, while a lower RMSdelay spread indicates a flatter, more frequency-invariant channel.
- Multipath delay spread can be mitigated by using techniques such as equalization, diversity,and adaptive modulation.
- Equalization techniques are used to compensate for the time dispersion caused by multipathdelay spread.

- Diversity techniques are used to combine multiple signal paths to mitigate the effects of fading.
- Adaptive modulation techniques are used to adjust the modulation scheme and data rate based on the channel conditions, allowing the system to adapt to changes in the channel and maintaina reliable communication link.

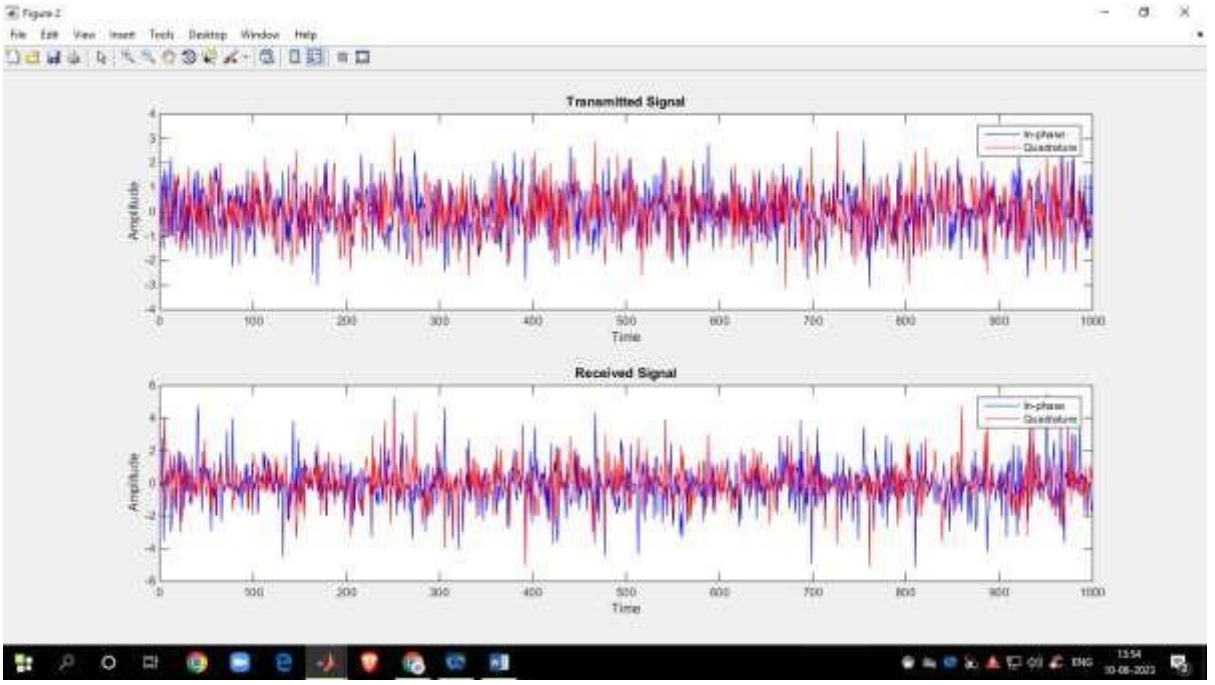
**PROCEDURE:**

1. Start the MATLAB program.
2. Open new M-file
3. Type the program
4. Save in current directory
5. Compile and Run the program
6. If any error occurs in the program correct the error and run it again
7. For the output see command window\ Figure window
8. Stop the program.

**CODING:**

```
% Simulation parameters

numSamples = 1000; % Number of samples
numPaths = 3;      % Number of multipath paths
fadePower = 0.5; % Fading power

% Generate Rayleigh fading channel coefficients

h = sqrt(fadePower/2)*(randn(numPaths, numSamples) + 1i*randn(numPaths, numSamples));

% Generate transmitted signal

txSignal = randn(1, numSamples) + 1i*randn(1, numSamples);

% Simulate multipath fading channel
rxSignal = zeros(1, numSamples);
for path = 1:numPaths
rxSignal = rxSignal + h(path, :) .* txSignal;
end

% Plot the transmitted and received signals
t = 1:numSamples;
figure;
subplot(2,1,1);
plot(t, real(txSignal), 'b', t, imag(txSignal), 'r');
title('Transmitted Signal');
legend('In-phase', 'Quadrature');
xlabel('Time');
ylabel('Amplitude');
subplot(2,1,2);
plot(t, real(rxSignal), 'b', t, imag(rxSignal), 'r');
title('Received Signal');
legend('In-phase', 'Quadrature');
xlabel('Time');
ylabel('Amplitude');
```

**OUTPUT** :




**RESULT:**

Thus the designing of a Model and simulation of Multipath fading channel has been achieved.

| Expt. No.:3 | Design, analyze and test Wireless standards and evaluate the performance measurements such as BER, PER, BLER, throughput,capacity, ACLR, EVM for 4G and 5G using Matlab |
|---|---|
| Date: | |

**AIM** :

Design, analyze and test Wireless standards and evaluate the performance measurements such asBER, PER, BLER, throughput, capacity, ACLR, EVM for 4G and 5G using Matlab.

**CODING:**

```
% Define simulation parameters

numBits = 1e6; % Number of bits to transmit
EbNo_dB = 10; % Eb/No in dB
% Generate QPSK symbols
txSymbols = randi([0 3], 1, numBits);
modulatedSymbols = pskmod(txSymbols, 4, pi/4);
% Add noise to the symbols
EbNo =10^(EbNo_dB/10);noiseVar = 1 / (2 * EbNo);
noise = sqrt(noiseVar) * randn(size(modulatedSymbols));
rxSymbols = modulatedSymbols + noise;
% Apply Rayleigh fading channel
Fade Channel =
rayleighchan(1/1000,30);
fadedSymbols = filter(fadeChannel, rxSymbols);
% Demodulate received symbols
demodulatedSymbols = pskdemod(fadedSymbols, 4, pi/4);
% Calculate Bit Error Rate (BER)
numErrors = sum(txSymbols ~=demodulatedSymbols);
ber = numErrors / numBits;
% Display results
fprintf('Bit Error Rate (BER): %.4f\n', ber);
```

**OUTPUT:**

Bit Error Rate (BER): 0.7512

**RESULT:**

Thus designing, analyzing and testing Wireless standards and evaluating the performance measurements such as BER, PER, BLER, throughput, capacity, ACLR, EVM for 4G and 5G using Matlab hasbeen achieved.

| Expt. No.:4 | Modulation: Spread Spectrum – DSSS Modulation & Demodulation |
|---|---|
| Date: | |

**AIM:**

To design modulation: Spread Spectrum – DSSS Modulation & Demodulation.

**SOFTWARE REQUIRED:**

Matlab version 2014

**THEORY:**

Direct-sequence spread spectrum in Wireless Networks is a technique that transmits a data signal over a range of frequencies, spreading it uniformly across the allocated spectrum. Direct-sequence spread spectrum is used to ensure that a particular frequency band (and its corresponding range of frequencies) is kept free from interference. This technique can be related to escaping the problem of co-channel interference (like two different wireless networks transmitting on the same frequency band) and cross-talk interference. Direct-sequence spread spectrum can also be used as an alternative approach to orthogonal frequency division multiplexing, where the baseband signal is encoded and transmitted across a quantity of fixed, predetermined channels. In this situation, each channel may carry different information, data signals, or time slots for different applications within the same network. Direct-sequence spread spectrum has also been used to transmit data that is encrypted and, in some processes, it is used to transmit non-data signals like power signaling or control signals.

Direct Sequence Spread Spectrum (DSSS) is a communication system that divides the bandwidthof a radio channel into wide frequency bands and transmits these signals over separate frequencies. In this frequency-hopping process, each signal is assigned a different orthogonal sequence of frequencies.All other radios in the range must gain each signal sequentially and then transmit it, which significantly reduces the risk of interference from outside sources or jamming. The time required for this process is proportional to the number of frequencies used for transmission.

When security agencies need to be ready to communicate secretly, DSSS can be implemented so that their transmissions cannot be spied upon by other parties who are monitoring broadcasts on a shorter wavelength or through tapping devices.

**PROCEDURE:**

1. Start the MATLAB program.
2. Open new M-file
3. Type the program
4. Save in current directory
5. Compile and Run the program
6. If any error occurs in the program correct the error and run it again
7. For the output see command window\ Figure window
8. Stop the program.

**CODING:**

```
% DSSS Modulation and Demodulation Example
% Parameters
data = [1 0 1 0 1 1 0 0]; % Original data signal
spreadingCode = [1 1 0 1]; % Spreading code
spreadingFactor = length(spreadingCode);
% DSSS Modulation
modulatedSignal = [];
for i = 1:length(data)
chips = repmat(data(i), 1, spreadingFactor) .* spreadingCode;
modulatedSignal = [modulatedSignal chips];
end
% DSSS Demodulation
demodulatedSignal = [];
for i = 1:length(modulatedSignal)/spreadingFactor
chips = modulatedSignal((i-1)*spreadingFactor+1:i*spreadingFactor);
chipSum = sum(chips);
    if chipSum>= spreadingFactor/2
  demodulatedSignal = [demodulatedSignal 1];
 else
  demodulatedSignal = [demodulatedSignal 0];
end
end
% Display Results disp('Original
Data:'); disp(data);
disp('Demodulated Data:');
disp(demodulatedSignal);
```

**OUTPUT:**

Original Data:

1 0 1 0 1 1 0 0

Demodulated Data:

1 0 1 0 1 1 0 0

**RESULT:**

Thus designing modulation: Spread Spectrum – DSSS Modulation & Demodulation has been achieved.

| Expt. No.:5 | Wireless Channel equalization: Zero-Forcing Equalizer (ZFE), MMSE Equalizer (MMSEE), |
|---|---|
| Date: | |

**AIM:**

To design a wireless Channel equalization: Zero-Forcing Equalizer (ZFE), MMS Equalizer (MMSEE), Adaptive Equalizer (ADE),Decision Feedback Equalizer (DFE).

**APPARATUS REQUIRED :**

PC with MATLAB Software

**THEORY:**

If the modulation bandwidth exceeds the coherent bandwidth of the radio channel ISI occur and modulation pulses are spread in time. Equalization compensates for ISI(Inter symbol interference)created by multipath within time dispersive channels. An equalizer within a receivercompensates for the average range of expected channel amplitude and delay characteristics.Equalizers can be classified as linear equalizer, non linear equalizer, zero forcing equalizer, Adaptive equalizer,(MMSE)Minimum mean square error equalizer and decision feedback equalizer.In zero forcing equalization the equalizer $g_k$ attempts to completely inverse the channelby forcing $c_k * g_k = \delta(k-ko)$.

**PROCEDURE:**

1. Start the MATLAB program.
2. Open new M-file
3. Type the program
4. Save in current directory
5. Compile and Run the program
6. If any error occurs in the program correct the error and run it again
7. For the output see command window\ Figure window
8. 8.Stop the program.

**ALGORITHM**

Simulate the link by following these steps:

1.Generate the number of bits and Eb/No value

2.Modulate BPSK Signal

**CODING :**

```
% Zero-Forcing Equalizer (ZFE) MATLAB code
% Define the channel impulse
responseh = [0.1 0.3 0.4 0.2];
% Generate random transmitted
symbolsN = 100; % Number of
symbols symbols = randi([0, 1], 1, N);
% Convolve transmitted symbols with the channel impulse response
received_signal = conv(symbols, h);
% Add AWGN (Additive White Gaussian Noise) to the received signal
snr_dB = 20;
% Signal-to-Noise Ratio (SNR) in dB
received_signal = awgn(received_signal, snr_dB, 'measured');
% Zero-Forcing Equalizer
% Define the length of the equalizer tap
L = length(h);
% Initialize the equalizer taps
equalizer_taps = zeros(1, L);
% Loop through each received symbol and perform equalization
equalized_symbols = zeros(1, N);
for n = 1:N
% Extract the received symbols for equalizationreceived_symbols =
received_signal(n:n+L-1);
% Perform zero-forcing equalization
equalized_symbols(n) = equalizer_taps * received_symbols';
% Update the equalizer taps using the least squares algorithm
```

```
error = symbols(n) - equalized_symbols(n);
equalizer_taps = equalizer_taps + error * received_symbols / (received_symbols *
received_symbols');
end
% Print the original symbols and equalized symbols
disp('Original Symbols:');
disp(symbols); disp('Equalized
Symbols:');
disp(equalized_symbols);
```

**OUTPUT:**

```
Original Symbols:
 Columns 1 through 19
   1   1   0   1   1   0   0   1   1   1   0   1   1   0   1   0   0   1   1

 Columns 20 through 38
   1   1   0   1   1   1   1   1   0   1   0   1   0   0   0   0   1   1   0

 Columns 39 through 57
   1   0   0   0   1   1   0   0   0   1   1   1   0   1   1   0   0   0   1

 Columns 58 through 76
   0   1   0   1   0   1   1   1   1   1   0   0   0   1   0   1   0   1   0

 Columns 77 through 95
   0   0   1   0   0   1   1   1   1   0   1   1   0   1   0   0   1   1   1

 Columns 96 through 100
   0   1   0   0   0
```

Equalized Symbols:

Columns 1 through 11

0 1.0242 1.0351 -0.0148 0.8118 0.8423 0.2395 0.3127 0.8637 0.5667 1.0034

Columns 12 through 22

0.3164 0.8987 0.7162 -0.0194 0.8262 0.2108 0.3684 1.3409 0.6328 0.5942 0.7986

Columns 23 through 33

0.3903 1.3034 0.9963 0.6816 0.6242 0.6419 0.1078 0.9584 -0.0282 0.4643 -0.0959

Columns 34 through 44

0.3857 0.3709 0.1746 1.1529 0.6859 -0.3254 0.6316 -0.1321 0.2851 0.6131 0.9881

Columns 45 through 55

0.1328 -0.3112 0.5753 0.4748 1.4226 0.8176 0.5202 0.2300 0.9991 0.4921 -0.2495

Columns 56 through 66

0.2145 0.5610 1.0497 -0.3251 1.0165 -0.0410 1.1669 0.3767 1.3984 0.8522 0.7683

Columns 67 through 77

0.6932 0.4118 -0.0997 0.1789 0.1747 1.2491 0.0166 1.0660 -0.0451 0.5827 -0.1786

Columns 78 through 88

0.2406 0.4407 0.5875 -0.0514 0.5994 1.4474 0.8587 0.6711 0.4184 0.5040 1.2422

Columns 89 through 99

0.4668 -0.0972 0.6936 -0.1060 0.7651 1.3313 0.6154 0.7091 0.0191 0.5241 -0.1900

Column 100

0.0171

---

**2. MMSE CODE**

```
% Parameters
M = 4;                  % Number of transmitted symbols
N = 1000;               % Number of received
symbols SNRdB = 10;             % Signal-to-
Noise Ratio in dBpilotSymbols = [1 -1 1 -1]; %
Known pilot symbols
% Generate random symbols
transmittedSymbols = randi([0 M-1], 1, N);
% Modulation
modulatedSymbols = qammod(transmittedSymbols, M);
% Channel
channel = [0.8 -0.4 0.2 -0.1];
% Example channel coefficients
channel Output = filter(channel, 1, modulatedSymbols);
% Add noise
SNR = 10^(SNRdB/10);
noiseVar = 1/(2*SNR);
noise = sqrt(noiseVar) * randn(1, length(channelOutput));
receivedSignal = channel Output + noise;
% Channel estimation using pilot symbols
pilotIndices = randperm(N, length(pilotSymbols));
pilotSignal = receivedSignal(pilotIndices);
estimated Channel = conv(pilotSignal, conj(pilotSymbols(end:-1:1)));
```

```
estimatedChannel = estimatedChannel(end-length(channel)+1:end);
% MMSE equalization
Equalizer Coefficients = conj(estimated Channel) ./ (abs(estimated Channel).^2 +
noiseVar);
equalized Symbols = filter(equalizer Coefficients, 1, receivedSignal);
% Demodulation
Demodulated Symbols = qamde mod(equalizedSymbols, M);
% Calculate bit error rate
bitErrors = sum(transmitted Symbols ~= demodulatedSymbols);
bitErrorRate = bitErrors / N;
disp(['Bit Error Rate: ' num2str(bitErrorRate)]);
```

**output:**

Bit Error Rate: 0.787

---

ADE:

```
% Parameters
channel_length = 10; % Length of the channel impulse
responsesnr_db = 20; % Signal-to-noise ratio in dB
num_symbols = 1000;  % Number of symbols to
transmitmu = 0.01;       % LMS step size
% Generate random symbols
data_symbols = randi([0, 1], 1, num_symbols);
% Modulate symbols (e.g., BPSK modulation)
modulated_symbols = 2 * data_symbols - 1;
% Create the channel impulse response
channel = (randn(1, channel_length) + 1i * randn(1, channel_length)) / sqrt(2);
% Convolve the modulated symbols with the channel
received_symbols = filter(channel, 1, modulated_symbols);
```

```
% Add noise to the received signal
noise_power = 10^(-snr_db / 10);
noise = sqrt(noise_power) * (randn(1, length(received_symbols)) + 1i * randn(1,
length(received_symbols)));
received_symbols_noisy = received_symbols + noise;
% Adaptive equalizer using the LMS algorithm
equalizer_length = channel_length;
 % Set the equalizer length to match the channel length
equalizer = zeros(1, equalizer_length);
output_signal = zeros(1, length(received_symbols_noisy));
for i = equalizer_length:length(received_symbols_noisy)
% Extract the received symbols for the current equalizer window
received_window = received_symbols_noisy(i:-1:i-equalizer_length+1);
% Compute the equalizer output
output_signal(i) = equalizer * received_window.';
% Compute the error
error = modulated_symbols(i) - output_signal(i);
% Update the equalizer coefficients
equalizer = equalizer + mu * conj(error) * received_window;
end
% Demodulate the equalized symbols (decision-directed)demodulated_symbols =
real(output_signal) > 0;
% Calculate the bit error rate (BER)
ber = sum(data_symbols ~= demodulated_symbols) / num_symbols;
disp(['Bit Error Rate (BER): ', num2str(ber)]);
```

**output:**

Bit Error Rate (BER): 0.519

**RESULT:**

Thus designing a wireless Channel equalization: Zero-Forcing Equalizer (ZFE), MMS Equalizer(MMSE), Adaptive Equalizer (ADE),Decision Feedback Equalizer(DFE) has been achieved.

| Expt. No.:6 | Modeling and simulation of TDMA, FDMA and CDMA for wireless communication |
|---|---|
| Date: | |

**AIM**:

To model and simulate TDMA, FDMA and CDMA for wireless communication.

**SOFTWARE REQUIRED:**

Matlab version 2014

**THEORY:**

Access methods are multiplexing techniques that provide communications services to multiple usersin a single-bandwidth wired or wireless medium. There are five basic access or multiplexing methods: frequency division multiple access (FDMA), time division multiple access (TDMA), codedivision multiple access (CDMA), orthogonal frequency division multiple access (OFDMA), and spatial division multiple access (SDMA). Each one of these takes advantage of multiplexing methods, dividing the bandwidth of the signal into different sub-bands, which are then assigned to different users in order to allow multiple users to share a single channel. Multiplexing is a communications technique that multiplexes, or combines, multiple signals into a single signal. The reverse process is called demultiplexing.

In FDMA, the time slots are assigned to the users in a sequential fashion. In TDMA, the time slotsare assigned to the users in a random fashion. In CDMA, the time slots are assigned to the users based on their code sequences.

FDMA:

FDMA is the process of dividing one channel or bandwidth into multiple individual bands, each foruse by a single user . Each individual band or channel is wide enough to accommodate the signal spectra of the transmissions to be propagated. The data to be transmitted is modulated on to each subcarrier, and all of them are linearly mixed together.

TDMA:

TDMA is a digital technique that divides a single channel or band into time slots. Each time slot isused to transmit one byte or another digital segment of each signal in sequential serial data format.This technique works well with slow voice data signals, but it’s also useful for compressed video and other high-speed data.

CDMA:

CDMA is another pure digital technique. It is also known as spread spectrum because it takes thedigitized version of an analog signal and spreads it out over a wider bandwidth at a lower power level. This method is called direct sequence spread spectrum (DSSS) as well *as* The digitized and compressed voice signal in serial data form is spread by processing it in an XOR circuit along with achipping signal at a much higher frequency.

**PROCEDURE:**

1. Start the MATLAB program.
2. Open new M-file
3. Type the program
4. Save in current directory
5. Compile and Run the program
6. If any error occurs in the program correct the error and run it again
7. For the output see command window\ Figure window8.Stop

the program.

**CODING:**

1.TDMA

```
% Step 1: Define System
ParametersnumUsers = 4;
timeSlotDuration = 1; % seconds
totalTimeSlots = 10;
channelGain = 0.8;
% Step 2: Generate User Traffic
userData = randi([0, 1], numUsers, totalTimeSlots);
% Step 3: Create Time Slots
timeSlots = linspace(0, timeSlotDuration*totalTimeSlots, totalTimeSlots);
% Step 4: Allocate Time Slots to Users
userSlots = mod(0:totalTimeSlots-1, numUsers) + 1;
% Step 5: Simulate Transmission
```

```
receivedData = zeros(numUsers, totalTimeSlots);
for slot = 1:totalTimeSlots
for user = 1:numUsers
if userSlots(slot) == user
% Simulate transmission for the current user in the time slot transmitted Data =
userData(user, slot);
% Simulate channel effects
receivedData(user, slot) = transmittedData * channelGain;
end
end
end
```

**% Step 6**: Evaluate Performance Metrics (e.g., BER)

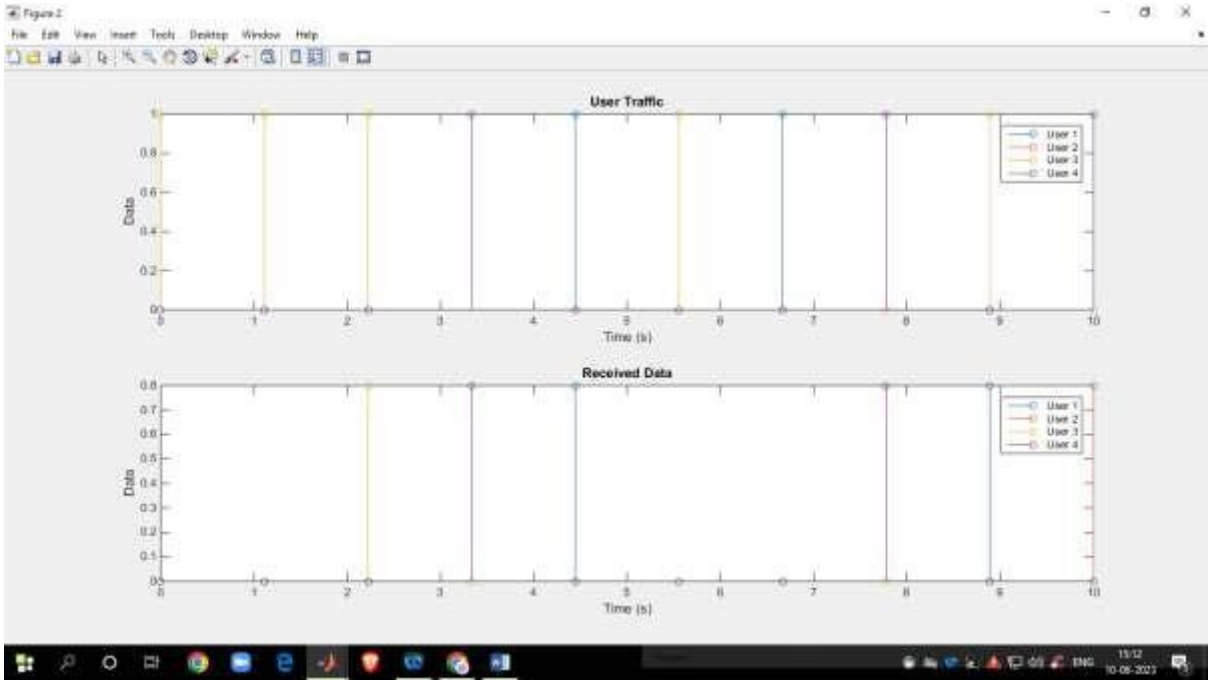
```
bitErrorRate = sum(sum(xor(receivedData, userData))) / (numUsers * totalTimeSlots);
```

**% Step 7**: Visualize Resultsfigure;

```
subplot(2, 1, 1);
stem(timeSlots, userData');
title('User Traffic');
xlabel('Time (s)');
ylabel('Data');
legend('User 1', 'User 2', 'User 3', 'User 4');
subplot(2, 1, 2);
stem(timeSlots, receivedData');
title('Received Data');
xlabel('Time (s)');
ylabel('Data');
legend('User 1', 'User 2', 'User 3', 'User 4');
disp(['Bit Error Rate: ', num2str(bitErrorRate)]);
```

**Output :**

Bit Error Rate: 0.375




---

2.FDMA:

```
% System parameters
totalBandwidth = 10e6; % Total available bandwidth (Hz)
numUsers = 5; % Number of users
carrierFrequency = 1e6; % Carrier frequency (Hz)
userBandwidth = totalBandwidth / numUsers; % Bandwidth allocated to each user (Hz)
% Time parameters
samplingFrequency = 100e6; % Sampling frequency (Hz)
timeDuration = 1e-3; % Simulation duration (s)
time = 0:1/samplingFrequency:timeDuration;
% Generate user signals
userSignals = zeros(numUsers, length(time));
for i = 1:numUsers
userFrequency = carrierFrequency + (i-1) * userBandwidth; % Frequency of user signal
```

```
userSignals(i, :) = sin(2*pi*userFrequency*time);
end
% Create the FDMA signal
fdmaSignal = sum(userSignals, 1);
% Add noise to the FDMA signal
snr = 10; % Signal-to-Noise Ratio (in dB)
noisySignal = awgn(fdmaSignal, snr, 'measured');
% Perform signal demodulation
demodulatedSignals = zeros(numUsers, length(time));
for i = 1:numUsers
user Frequency = carrierFrequency + (i-1) * userBandwidth; % Frequency of user signal
demodulatedSignals(i, :) = noisySignal .* sin(2*pi*userFrequency*time);
end
% Plot the original user signals and the demodulated signalsfigure;
subplot(numUsers+1, 1, 1);
plot(time, fdmaSignal);
 title('FDMA Signal');
xlabel('Time (s)');
ylabel('Amplitude');
for i = 1:numUsers subplot(numUsers+1, 1, i+1);
 plot(time, demodulatedSignals(i, :));
title(['Demodulated Signal - User ', num2str(i)]);
xlabel('Time (s)');
ylabel('Amplitude');
end
```

Output:


---

3.CDMA:

```
% CDMA Simulation Parameters
numBits = 1000;   % Number of bits per user

chipRate = 1e6;   % Chip rate (chips per second)

snr = 10;         % Signal-to-Noise Ratio (dB)

% Generate random data bits for User 1 and User 2

user1Bits = randi([0, 1], 1, numBits);

user2Bits = randi([0, 1], 1, numBits);
% BPSK Modulation
user1Symbols = 2 * user1Bits - 1;    % Map 0s to -1 and 1s to 1

user2Symbols = 2 * user2Bits - 1;

% Chip-level Spreading (using a simple chip sequence)

chipSequence = [1, -1, 1, 1, -1, 1, -1, -1];        % Chip sequence for spreading

user1SpreadSymbols = kron(user1Symbols, chipSequence);
```

```
user2SpreadSymbols = kron(user2Symbols, chipSequence);
% Add AWGN (Additive White Gaussian Noise)noiseVar = 10^(-snr/10);
% Noise variance
user1NoisySymbols = user1SpreadSymbols + sqrt(noiseVar/2) * randn(1,
length(user1SpreadSymbols));
user2NoisySymbols = user2SpreadSymbols + sqrt(noiseVar/2) * randn(1,
length(user2SpreadSymbols));
% Matched Filtering (correlation with chip sequence)
user1FilteredSymbols = filter(fliplr(chipSequence), 1, user1NoisySymbols);
user2FilteredSymbols = filter(fliplr(chipSequence), 1, user2NoisySymbols);
% Symbol Detection (using correlation with chip sequence)
user1DetectedBits =user1FilteredSymbols(1:length(user1Symbols)) > 0;
user2DetectedBits = user2FilteredSymbols(1:length(user2Symbols)) > 0;
% Bit Error Rate (BER) Calculation
berUser1 = sum(user1DetectedBits ~= user1Bits) / numBits;
berUser2 = sum(user2DetectedBits ~= user2Bits) / numBits;
% Display results
disp(['User 1 BER: ', num2str(berUser1)]);
disp(['User 2 BER: ', num2str(berUser2)]);
```

**Output:**

User 1 BER: 0.523

User 2 BER: 0.535

**Result :**

Thus modeling and simulation of TDMA, FDMA and CDMA forwireless communication has been achieved.

**VIVA QUESTIONS :**

1. What is Okumura-Hata model in wireless communication?
2. What is the difference between Hata model and Okumura model?
3. Is the Hata model used for signal strength prediction?
4. What is wireless channel model?
5. What are the main wireless channels?
6. Which wireless channel is better?
7. What is multipath fading channel?
8. What are the effects of multipath fading?
9. What are multipath channels?
10. What are the causes of multipath?
11. What is the advantage of multipath?
12. How do you reduce multipath effects?
13. What are the disadvantages of multipath
14. What is the working principle of zero-forcing equalizer?
15. Why is zero forcing used?
16. What is the need of an equalizer?
17. What is the MMSE channel equalization?
18. What are channel equalization methods?
19. What is the function of the equalizer?
20. What is adaptive equalization of channel?
21. 8. What is DFE equalizer?
22. What is decision feedback equalizer technique?
23. Why is equalizer used?
24. What is the ACLR measurement in Matlab?
25. What is the ACLR requirement?
26. How to generate 5G signal in MATLAB?
27. What is the use of 5G toolbox in Matlab?
28. What is the full form of ACLR?
29. How do 5G antennas work?
30. What is the use of 5G toolbox in Matlab?
31. Why do we need spread spectrum?
32. What is spread spectrum modulation?
33. What are the types of spread spectrum?
34. What is the spread spectrum technique?
35. What is the formula for spectral spread?
36. What is the full form of DSSS?
37. What is the spreading factor of a spread spectrum?
38. What are the advantages of DSSS
39. What is the basic principle of TDMA?
40. What is TDMA used for?
41. Why GSM is called TDMA?
42. What is the basic principle of FDMA?
43. What is the frequency range of FDMA?
44. What are the applications of FDMA?
45. What is the principle of CDMA?
46. What technology is used in CDMA?
47. What is the noise power of a CDMA system?