EXPERIMENT NO: 1

5G-COMPLIANT WAVEFORM GENERATION AND TESTING

AIM:

To generate and test a 5G-compliant waveform using MATLAB.

SOFTWARE USED:

MATLAB

PROCEDURE:

1. Set parameters for the carrier frequency, sample rate, number of samples, and signal-to-noise ratio (SNR).
2. Generate random binary data for QPSK modulation.
3. Perform QPSK modulation by mapping bits to symbols.
4. Create a time vector based on the sample rate and number of samples.
5. Combine in-phase (I) and quadrature (Q) components to form the QPSK signal.
6. Generate the carrier signal using a complex exponential function.
7. Modulate the QPSK symbols with the carrier signal to obtain the transmitted signal.
8. Add AWGN (Additive White Gaussian Noise) to simulate real-world channel effects.
9. Demodulate the received signal by removing the carrier.
10. Extract the phase information from the received signal.
11. Convert the demodulated symbols back to binary data based on phase values.
12. Plot the following:

    Transmitted signal (I-component)

    Received signal with noise

    Comparison of transmitted and decoded data

MATLAB code

```
clc;
clear all;
close all;

% Parameters
carrierFrequency = 3.5e9; % Carrier frequency in Hz (3.5 GHz for sub-6GHz 5G)
sampleRate = 30.72e6; % Sample rate in Hz
numSamples = 1024; % Number of samples in the waveform
snr = 20; % Signal-to-noise ratio in dB

% Generate a simple 5G waveform (QPSK modulation)
data = randi([0, 1], 2, numSamples); % Random bits for QPSK modulation
qpskSymbols = 2 * data - 1; % Map bits to QPSK symbols (-1, 1)

% Create a time vector
time = (0:numSamples - 1) / sampleRate;

% Modulate the QPSK symbols
qpskSignal = qpskSymbols(1, :) + 1j * qpskSymbols(2, :);

% Generate the carrier signal
carrierSignal = exp(1j * 2 * pi * carrierFrequency * time);

% Generate the transmitted signal
transmittedSignal = qpskSignal .* carrierSignal;

% Add noise to the transmitted signal
noisySignal = awgn(transmittedSignal, snr, 'measured');

% Receiver: downconvert by removing the carrier
```

```
receivedSignal = noisySignal ./ carrierSignal;

% Demodulate the received signal
demodulatedSymbols = angle(receivedSignal);

% Decode the demodulated symbols back to bits
decodedData = demodulatedSymbols > 0;

% Plot the results
subplot(3, 1, 1);
plot(time, real(transmittedSignal));
title('Transmitted Signal (I Component)');
xlabel('Time (s)');
ylabel('Amplitude');

subplot(3, 1, 2);
plot(time, real(noisySignal));
title('Received Signal with Noise');
xlabel('Time (s)');
ylabel('Amplitude');

subplot(3, 1, 3);
stem(data(:), 'rx');
hold on;
stem(decodedData(:), 'bo');
title('Transmitted and Decoded Data');
xlabel('Sample Index');
ylabel('Bit Value');
legend('Transmitted Data', 'Decoded Data');
```

1. Initialization

clc;

clear all;

close all;

- clc: Clears the command window.
- clear all: Removes all variables from the workspace.
- close all: Closes all open figure windows.

---

2. Parameters

carrierFrequency = 3.5e9; % Carrier frequency in Hz (3.5 GHz for sub-6GHz 5G)

sampleRate = 30.72e6; % Sample rate in Hz

numSamples = 1024; % Number of samples in the waveform

snr = 20; % Signal-to-noise ratio in dB

- Defines key parameters: carrier frequency, sample rate, number of samples, and SNR.
- These values are standard for simulating a simple 5G sub-6 GHz system.

---

3. QPSK Modulation

data = randi([0, 1], 2, numSamples); % Random bits for QPSK modulation

qpskSymbols = 2 * data - 1; % Map bits to QPSK symbols (-1, 1)

- **QPSK modulation** uses 2 bits per symbol.
- randi([0, 1], 2, numSamples) generates random bits (0 or 1) in a 2x1024 matrix.
- 2 * data - 1 maps bits from [0,1][0, 1][0,1] to QPSK symbols [−1,1][-1, 1][−1,1].

For example:

- Bit pair [0, 0] → symbol (-1, -1)
- Bit pair [1, 0] → symbol (1, -1)

🔀 How QPSK works

- QPSK uses **two orthogonal components:**
  - **In-phase component (I):** Represents the real part of the signal.
  - **Quadrature component (Q):** Represents the imaginary part of the signal.

The QPSK signal can be expressed as:

$$s(t) = A\cos(2\pi f_c t) + B\sin(2\pi f_c t)$$

where:

- A and B are the amplitudes determined by the bit pairs.
- f_c is the carrier frequency.

The possible combinations of bits and their corresponding symbols are:

Bit Pair	In-phase (I)	Quadrature (Q)	Symbol (I + jQ)	Phase (Radians)
(0,0)	-1	-1	-1 - j	$-\frac{3\pi}{4}$
(0,1)	-1	+1	-1 + j	$\frac{3\pi}{4}$
(1,0)	+1	-1	1 - j	$-\frac{\pi}{4}$
(1,1)	+1	+1	1 + j	$\frac{\pi}{4}$

Each symbol corresponds to a unique point on the **constellation diagram** — a plot of the I and Q components.

---

4. Time Vector

time = (0:numSamples - 1) / sampleRate;

- his generates a sequence of integers starting from 0 up to numSamples - 1.
- For example, if numSamples = 1024:
- 0, 1, 2, ..., 1023
- This represents the sample indices.
- (0:numSamples - 1) / sample Rate
- Each sample index is divided by the sample Rate to convert the indices into time values.
- The time vector will have units of seconds.

Why divide by sample Rate?

- The sample rate (in Hz) tells us how many samples are taken per second.
- Time for each sample is given by:
- $t = n/fs$

- where:
- t = time at sample
- fs = sample rate
- Example:
- If:
- numSamples = 5;
- sampleRate = 10; % 10 Hz
- Then:
- time = (0:5 - 1) / 10;
- This produces:
- time = [0, 0.1, 0.2, 0.3, 0.4]
- Sample 0 happens at 0 seconds.
- Sample 1 happens at 0.1 seconds.
- Sample 4 happens at 0.4 seconds.
- Purpose:
- The time vector is essential for plotting signals over time and modulating carrier waves like:
- $\cos(2\pi f t)$

---

5. Modulation

qpskSignal = qpskSymbols(1, :) + 1j * qpskSymbols(2, :);

- Combines the two bit streams into **complex QPSK symbols**:

QPSK Signal=I+jQQPSK\ Signal = I + jQQPSK Signal=I+jQ

where III and QQQ are the two bit streams.

---

6. Carrier Signal

matlab

CopyEdit

carrierSignal = exp(1j * 2 * pi * carrierFrequency * time);

- Creates a **complex exponential carrier wave**:

ej2πfte^{j2\pi ft}ej2πft

- Used for **upconversion** (modulating the baseband QPSK signal onto the carrier frequency).

---

7. Transmit the Signal

```
transmittedSignal = qpskSignal .* carrierSignal;
```

- Modulates the QPSK signal onto the carrier frequency.

---

8. Add Noise

```
noisySignal = awgn(transmittedSignal, snr, 'measured');
```

- Adds **Additive White Gaussian Noise (AWGN)** with the specified SNR (20 dB).

---

9. Receiver (Downconversion)

```
receivedSignal = noisySignal ./ carrierSignal;
```

- **Downconverts** the received signal by dividing out the carrier wave.
- This brings the signal back to the baseband (removes the high-frequency carrier component).

---

10. Demodulation

```
demodulatedSymbols = angle(receivedSignal);
```

- Extracts the **phase angle** of the received signal.
- QPSK demodulation works by checking the phase to decide the bit values.

---

11. Decode Bits

```
decodedData = demodulatedSymbols > 0;
```

- **Decodes bits** by checking if the phase is positive or negative:
    - Positive phase → bit 1
    - Negative phase → bit 0

---

12. Plot the Results

```
subplot(3, 1, 1);

plot(time, real(transmittedSignal));
```

```
title('Transmitted Signal (I Component)');
xlabel('Time (s)');
ylabel('Amplitude');
```

- Plots the **in-phase (real part)** of the transmitted signal.

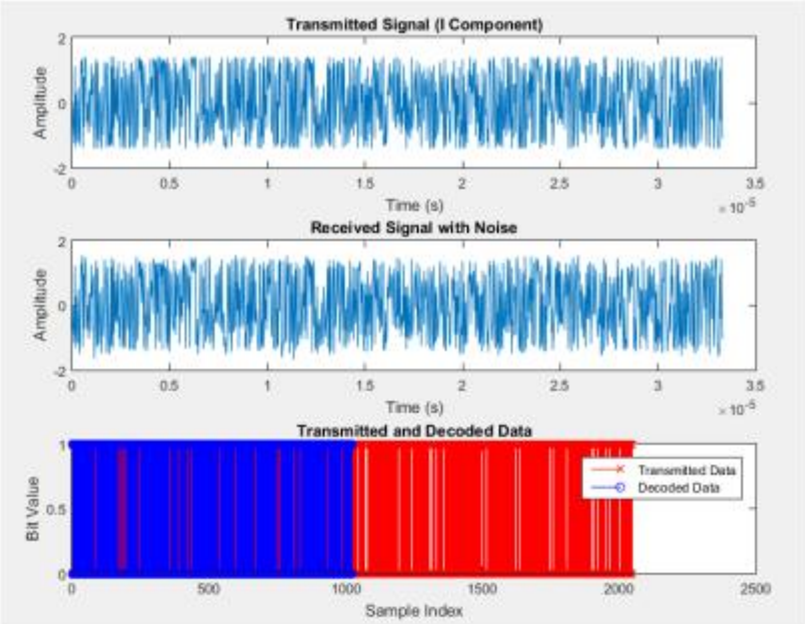
```
subplot(3, 1, 2);
plot(time, real(noisySignal));
title('Received Signal with Noise');
xlabel('Time (s)');
ylabel('Amplitude');
```

- Plots the real part of the **received (noisy) signal**.

```
subplot(3, 1, 3);
stem(data(:), 'rx');
hold on;
stem(decodedData(:), 'bo');
title('Transmitted and Decoded Data');
xlabel('Sample Index');
ylabel('Bit Value');
legend('Transmitted Data', 'Decoded Data');
```

- **Red crosses (rx):** transmitted bits
- **Blue circles (bo):** decoded bits
- **Compares original and received bits** to check for errors visually

Output:



Result:

Thus the 5G-Compliant Waveform Generation and Testing in MATLAB was successfully executed.