Introduction to digital communication

Final project

Part (2)

Line code

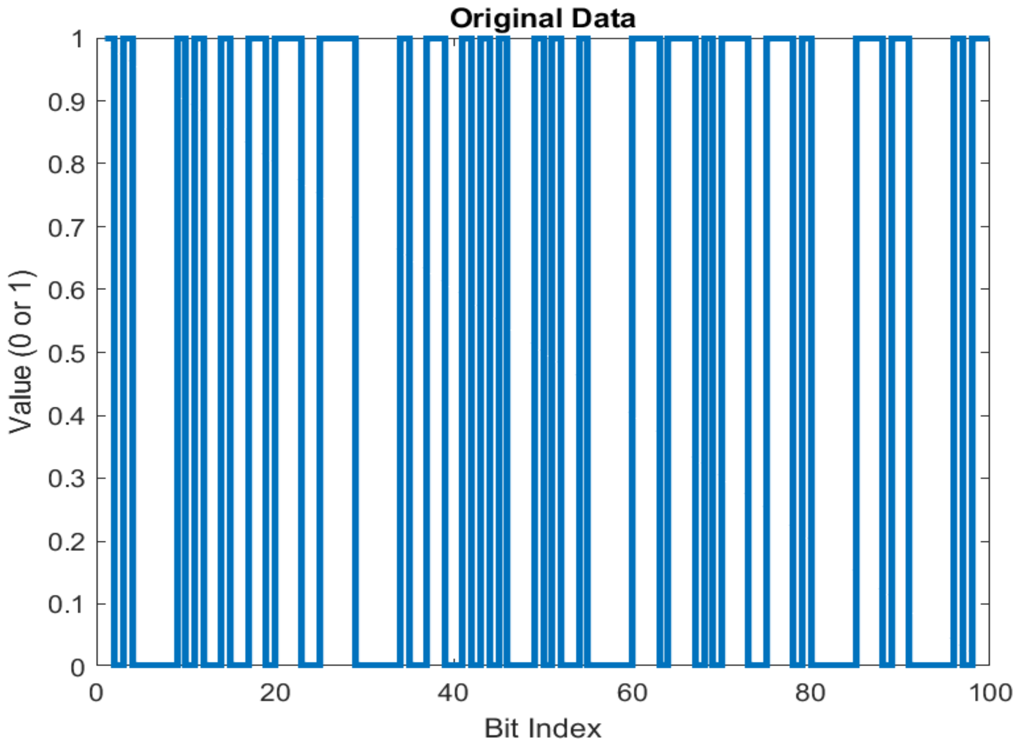
Name	ID
Asmaa Hassan Mokhtar Aboushady	19015430
Perihan Hossam Eldin Imam	20010392
Abdelraof Fathy Abdelraof Abdelrahman	20010777

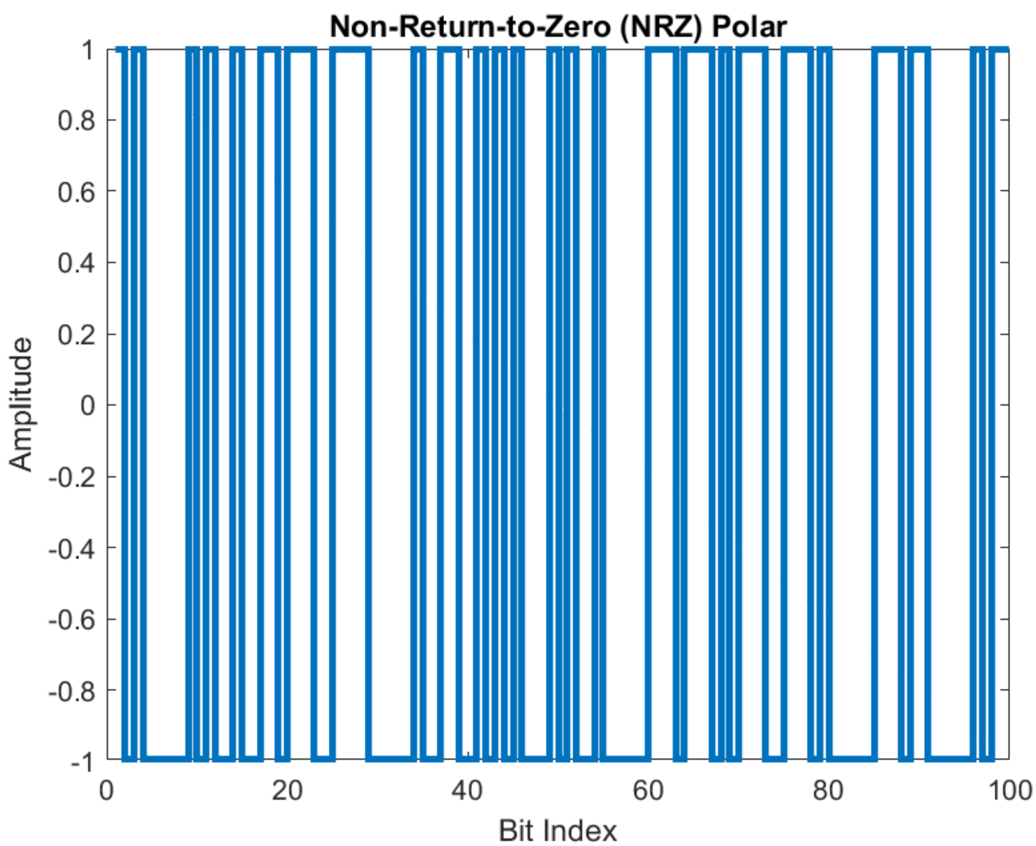
```matlab
clc;

clear;

% Number of bits to generate
num_bits = 100;

% Generate a random binary sequence
rand_bits = randi([0, 1], 1, num_bits);

% Display the original data
figure;
stairs(1:num_bits, rand_bits, 'LineWidth', 2.5); % Ensure x-axis is correct
title('Original Data');
xlabel('Bit Index');
ylabel('Value (0 or 1)');
%% Create the NRZ signal
% Initialize NRZ signal
% The signal has the same length as the number of bits
nrz_signal = zeros(1, num_bits);
for i = 1:num_bits
    if rand_bits(i) == 1
        nrz_signal(i) = 1; % Polar NRZ: "1" is positive
    else
        nrz_signal(i) = -1; % Polar NRZ: "0" is negative
    end
end

% Plot the NRZ signal
figure;
stairs(1:num_bits, nrz_signal, 'LineWidth', 2.5);
title('Non-Return-to-Zero (NRZ) Polar');
xlabel('Bit Index');
ylabel('Amplitude');

%% NRZ-I
% Initialize the NRZ-I signal
nrz_i_signal = zeros(1, num_bits); % The signal has the same length as the number of bits

% Define the initial level (start with a known state, e.g., low level)
current_level = -1;

% Create the NRZ-I signal
for i = 1:num_bits
    if rand_bits(i) == 1
        % Transition for binary "1"
        current_level = -current_level; % Toggle the level
    end
    % Maintain the same level for binary "0"
    nrz_i_signal(i) = current_level;
end

% Plot the NRZ-I signal
figure;
stairs(1:num_bits, nrz_i_signal, 'LineWidth', 2.5);
title('Non-Return-to-Zero Inverted (NRZ-I)');
xlabel('Bit Index');
ylabel('Amplitude');
```

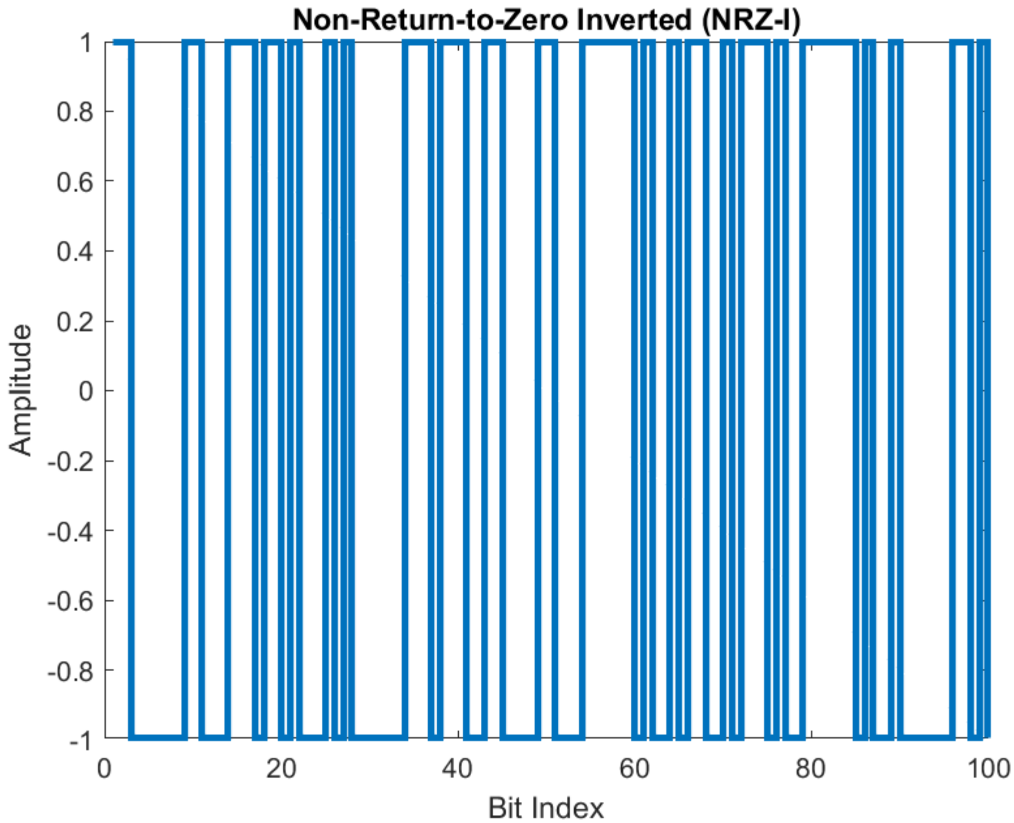
```matlab
%% Initialize Return-to-Zero (RZ) signal
RZ_signal = zeros(1, num_bits * 2); % Double the length for RZ
time = linspace(0, num_bits, num_bits * 2); % Adjust time axis to match RZ_signal length

% Create the RZ signal
for i = 1:num_bits
    % Index into the RZ_signal array
    index = (i - 1) * 2 + 1;  % Find the position in the RZ array
    if rand_bits(i) == 1
        RZ_signal(index) = 1;  % Set the first half of the bit duration
        RZ_signal(index + 1) = 0;  % Return to zero for the second half
    else
        RZ_signal(index) = 0;  % No signal for a "0"
        RZ_signal(index + 1) = 0;  % Remains zero
    end
end

% Plot the Return-to-Zero (RZ) signal
figure;
stairs(time, RZ_signal, 'LineWidth', 2.5);  % Ensure arrays are same length
title('Return-to-Zero (RZ) Signal');
xlabel('Time');
ylabel('Amplitude');


%% Create the AMI signal

% Initialize AMI signal and a variable to track the sign of "1"s
AMI_signal = zeros(1, num_bits); % Starts as all zeros
z = 1; % This variable will determine the alternating sign
for i = 1:num_bits
    if rand_bits(i) == 1 % Check if the current bit is 1
        AMI_signal(i) = z; % Assign the signal with the current sign
        z = -z; % Alternate the sign for the next "1"
    else
        AMI_signal(i) = 0; % If it's a zero, keep it zero
    end
end

% Plot the AMI signal
figure;
stairs(1:num_bits, AMI_signal, 'LineWidth', 2.5);
title('Alternate Mark Inversion (AMI)');
xlabel('Bit Index');
ylabel('Amplitude');
```

```matlab
%% create manchester
% Initialize Manchester signal
% Each bit is represented by two transitions, so the signal length is doubled
manchester_signal = zeros(1, num_bits * 2); % Each bit has two transitions
time = linspace(0, num_bits, num_bits * 2); % Time axis

% Create the Manchester signal
for i = 1:num_bits
    index = (i - 1) * 2 + 1; % Position in the Manchester array

    if rand_bits(i) == 0
        % Encode "0" with low-to-high transition
        manchester_signal(index) = 0;     % First half low
        manchester_signal(index + 1) = 1; % Second half high
    else
        % Encode "1" with high-to-low transition
        manchester_signal(index) = 1;     % First half high
        manchester_signal(index + 1) = 0; % Second half low
    end
end

% Plot the Manchester signal
figure;
stairs(time, manchester_signal, 'LineWidth', 2.5);
title('Manchester Code');
xlabel('Time');
ylabel('Amplitude');

%% multi level transmission 3
% Initialize 3-PAM signal
% Group every two bits to determine the 3-PAM value
num_symbols = floor(num_bits / 2); % Each symbol represents two bits
pam3_signal = zeros(1, num_symbols);
time = linspace(0, num_symbols, num_symbols); % Time axis

% Create the 3-PAM signal
for i = 1:num_symbols
    % Get the bits for this symbol
    bit1 = rand_bits(2*i - 1);
    bit2 = rand_bits(2*i);

    % Determine the PAM-3 level
    if bit1 == 0 && bit2 == 0
        pam3_signal(i) = -1; % Lowest level
    elseif bit1 == 0 && bit2 == 1
        pam3_signal(i) = 0; % Middle level
    elseif bit1 == 1 && bit2 == 0
        pam3_signal(i) = 1; % Highest level
    elseif bit1 == 1 && bit2 == 1
        pam3_signal(i) = 2; % Highest level (if you need this extra level)
    end
end
```

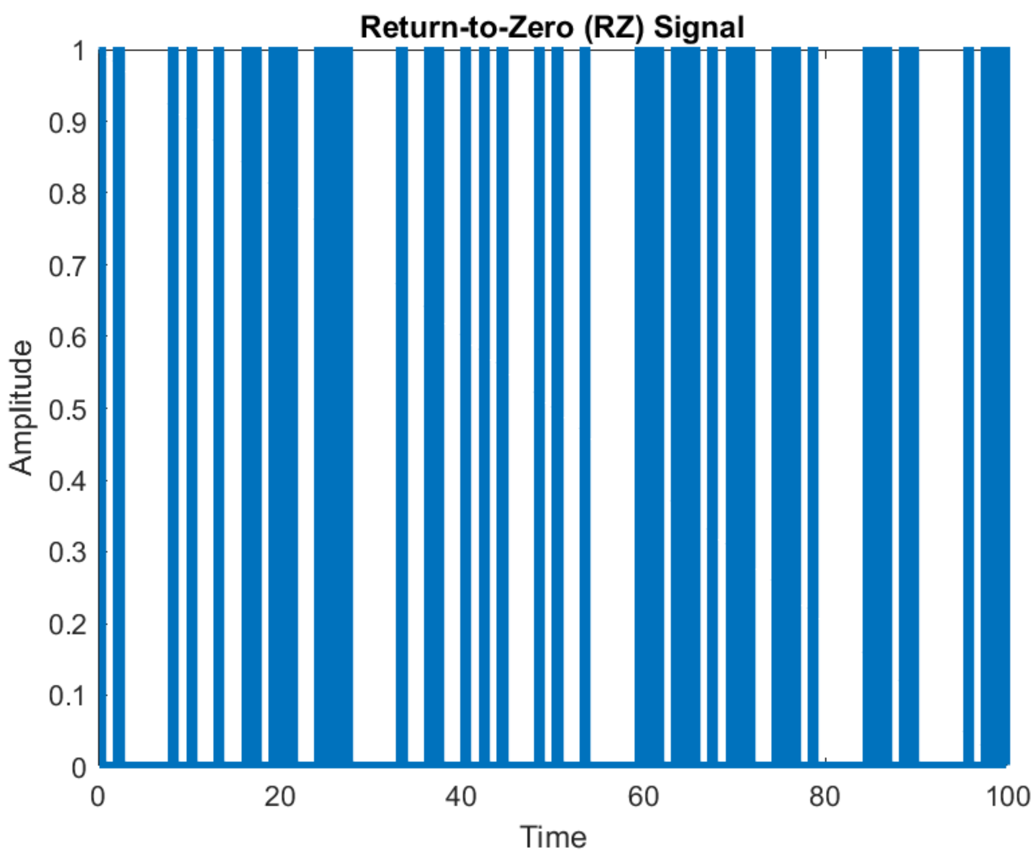
```
% Plot the 3-PAM signal
figure;
stairs(time, pam3_signal, 'LineWidth', 2.5);
title('3-Level Pulse Amplitude Modulation (3-PAM)');
xlabel('Time');
ylabel('Amplitude');

%% PSD CALCULATIONS
A=1;
f = 0:1:100;
Bit_Duration= 1/20;
PSD_NRZ = (A^2)*Bit_Duration * (sinc(f*Bit_Duration)).^2;
PSD_NRZI = ((A^2)*Bit_Duration/2) * ((sinc(f*Bit_Duration)).^2).* (1+cos(pi*f*Bit_Duration));
PSD_RZ= ((A^2)/2) * ((sinc(f*(Bit_Duration/2))).^2);
PSD_AMI=  ((A^2)*Bit_Duration/4) * ((sinc(f*(Bit_Duration/2))).^2) .* (sin (pi
*f*Bit_Duration)).^2;
PSD_MANCHESTER = (A^2)*Bit_Duration  * ((sinc(f*(Bit_Duration/2))).^2) .* (sin (pi
*f*Bit_Duration/2)).^2;
PSD_MLT3= ((A^2)*Bit_Duration/4) * ((sinc(f*(Bit_Duration/2))).^2) + ((1./ (pi*f
*Bit_Duration)).^2).*((sin(2*pi*f*(Bit_Duration))/2).^2)-
((A^2)*Bit_Duration/4).*((sinc(f.*Bit_Duration)).^2);

%% plot PSD

figure ;
plot(f, PSD_NRZ, 'b', 'LineWidth', 2.5);
title('Power Spectral Density of Return-to-Zero (RZ)');
xlabel('Frequency (Hz)');
ylabel('Power/Frequency (dB/Hz)');


figure ;
plot(f, PSD_NRZI, 'b', 'LineWidth', 2.5);
title('Power Spectral Density of Return-to-Zero (RZ)');
xlabel('Frequency (Hz)');
ylabel('Power/Frequency (dB/Hz)');

figure ;
plot(f, PSD_RZ, 'b', 'LineWidth', 2.5);
title('Power Spectral Density of Return-to-Zero (RZ)');
xlabel('Frequency (Hz)');
ylabel('Power/Frequency (dB/Hz)');

figure;
plot(f, PSD_AMI, 'r', 'LineWidth', 2.5);
title('Power Spectral Density of Alternate Mark Inversion (AMI)');
xlabel('Frequency (Hz)');
ylabel('Power/Frequency (dB/Hz)');

figure ;
plot(f, PSD_MANCHESTER, 'g', 'LineWidth', 2.5);
title('Power Spectral Density of Manchester Code');
xlabel('Frequency (Hz)');
ylabel('Power/Frequency (dB/Hz)');


figure;
plot(f, PSD_MLT3, 'c', 'LineWidth', 2.5);
title('Power Spectral Density of 3-Level PAM');
xlabel('Frequency (Hz)');
ylabel('Power/Frequency (dB/Hz)');
```


Original Data
Value (0 or 1)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
20
40
60
80
100
Bit Index

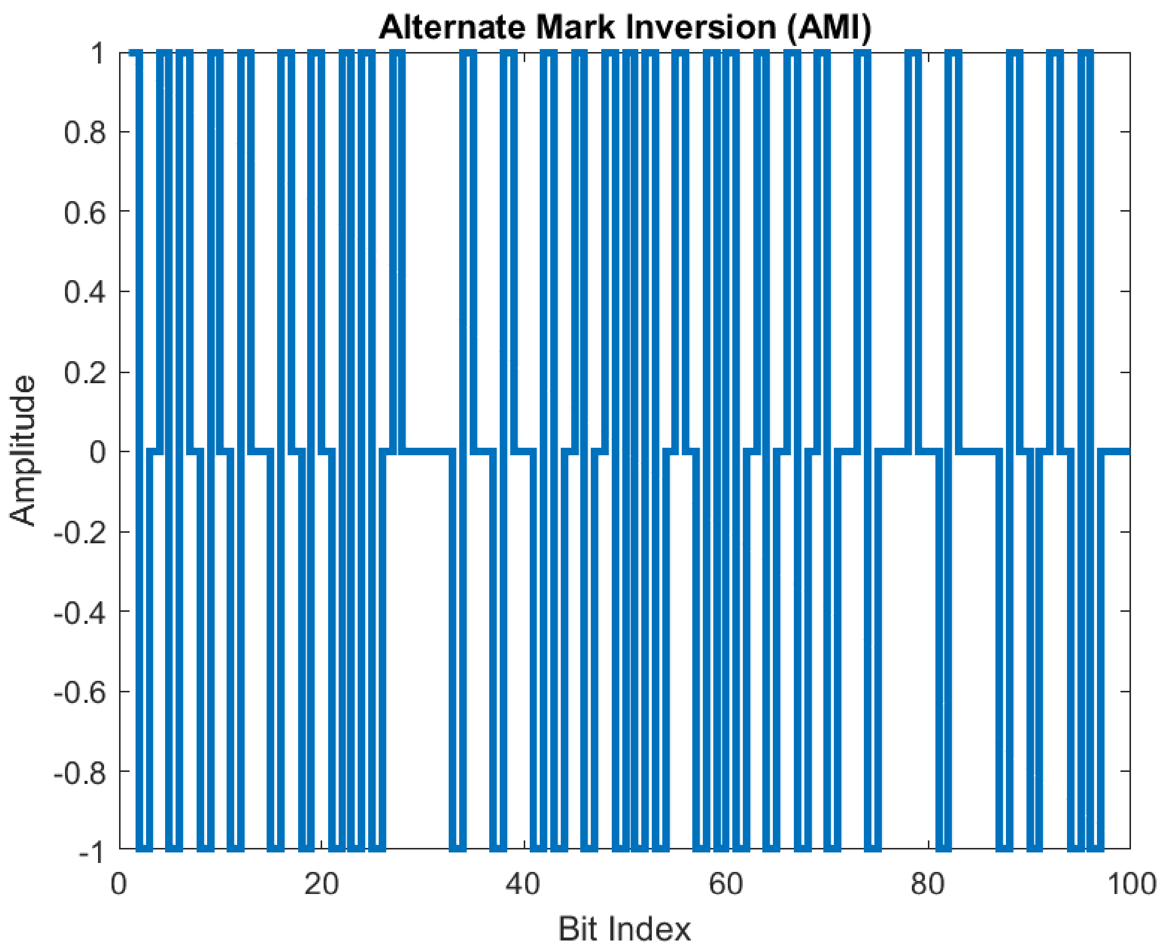

Non-Return-to-Zero (NRZ) Polar
Amplitude
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
20
40
60
80
100
Bit Index



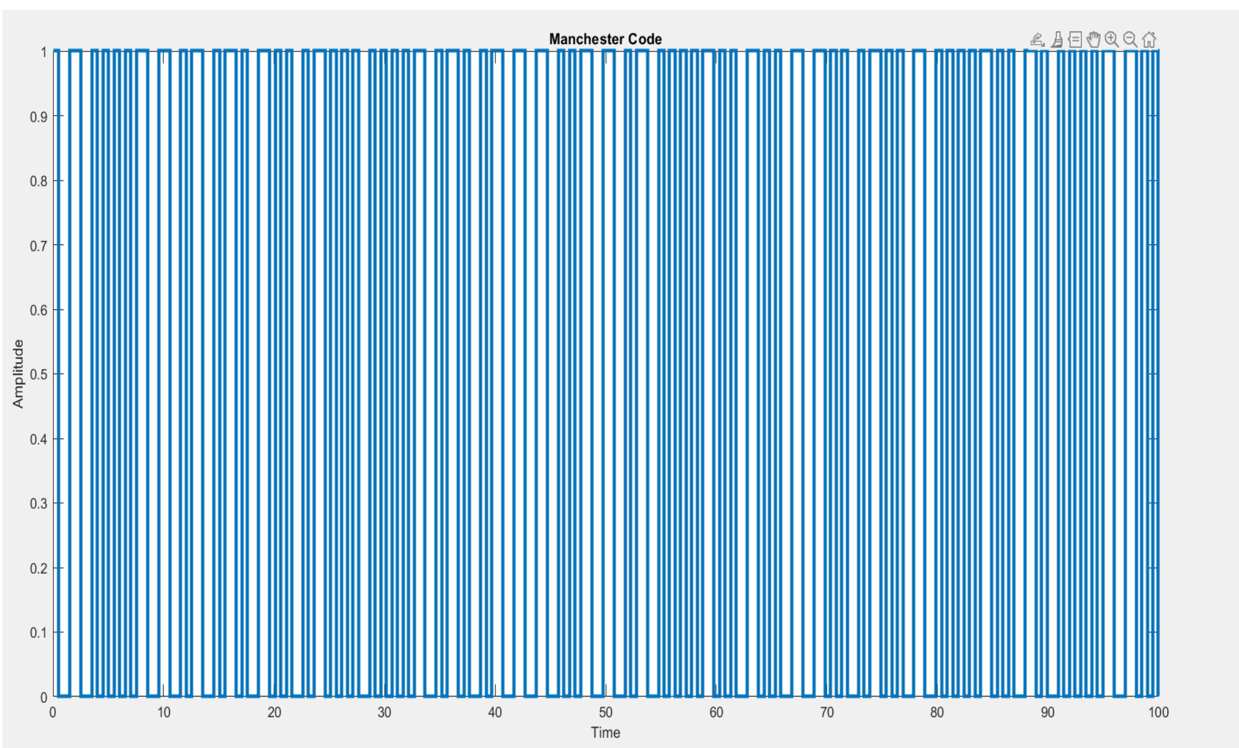
Non-Return-to-Zero Inverted (NRZ-I)
Amplitude
Bit Index
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
20
40
60
80
100



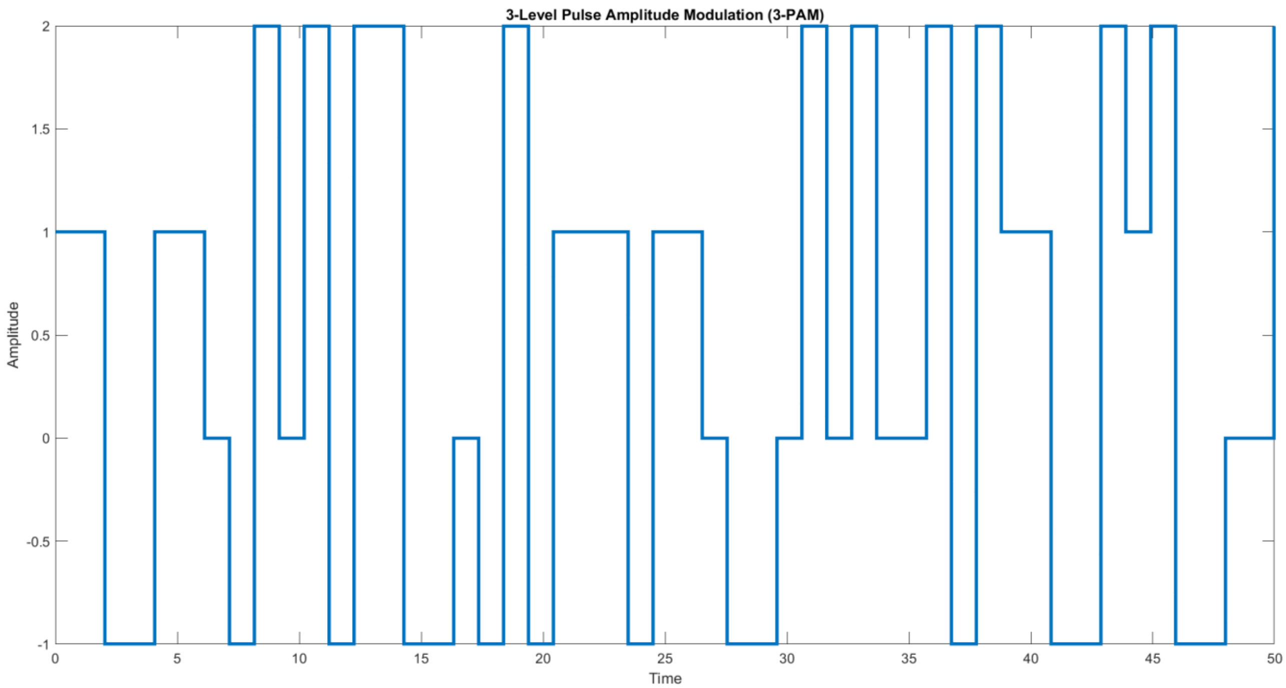
Return-to-Zero (RZ) Signal
Amplitude
Time
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
20
40
60
80
100

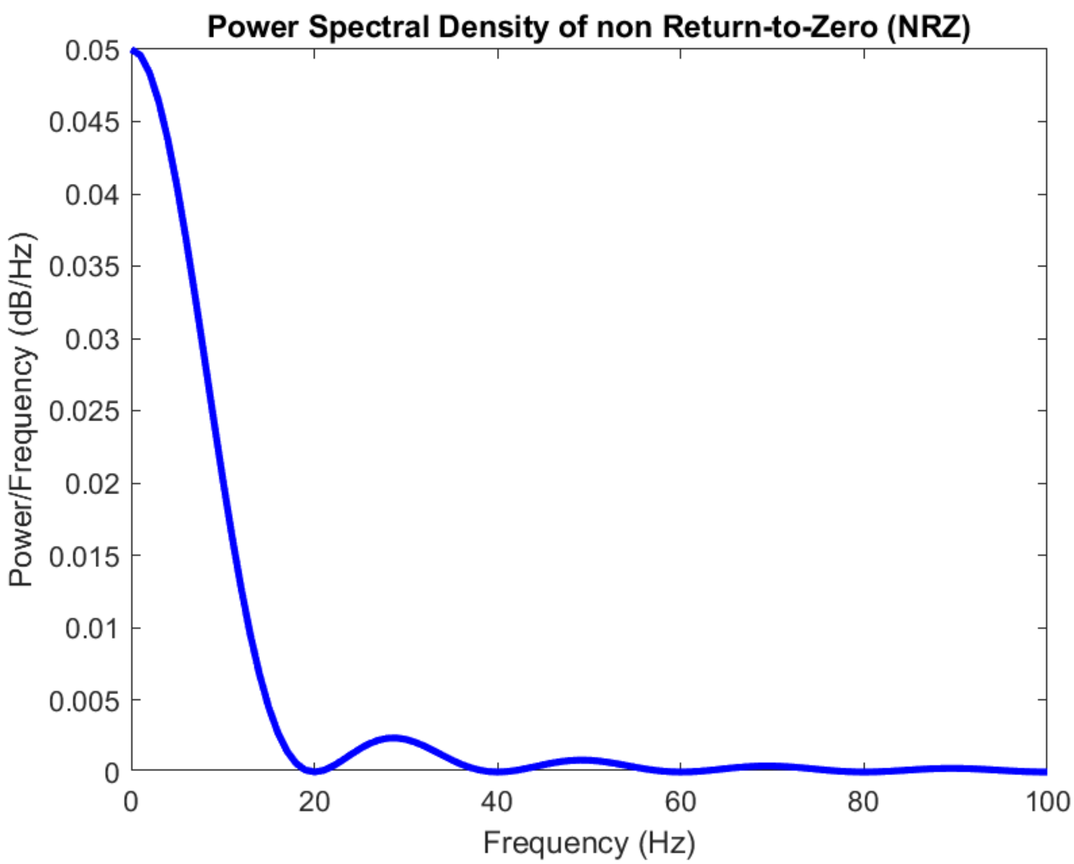


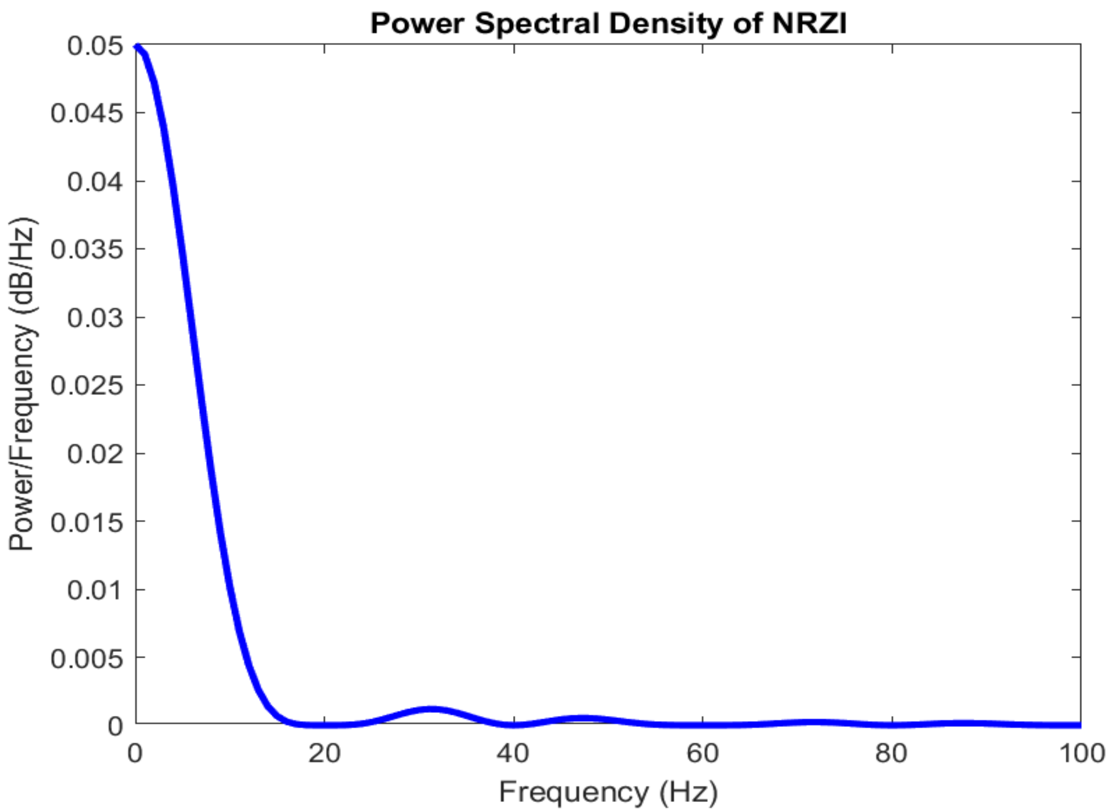
Alternate Mark Inversion (AMI)
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
Amplitude
0
20
40
60
80
100
Bit Index

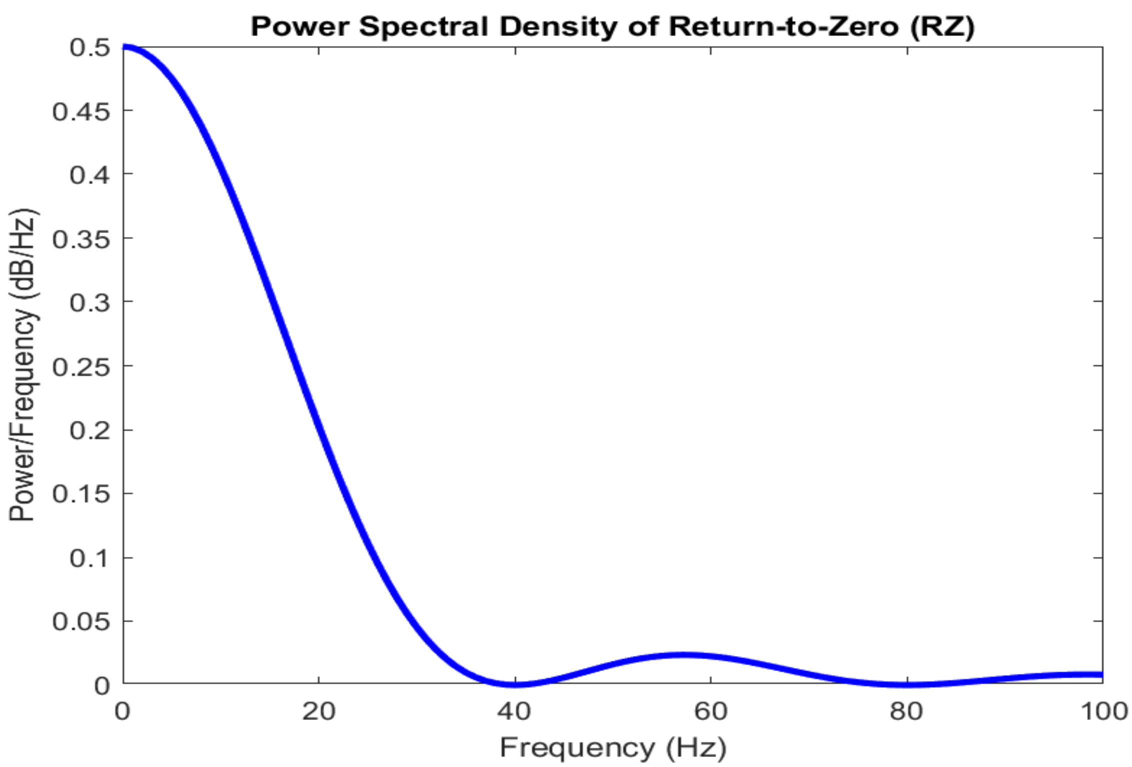


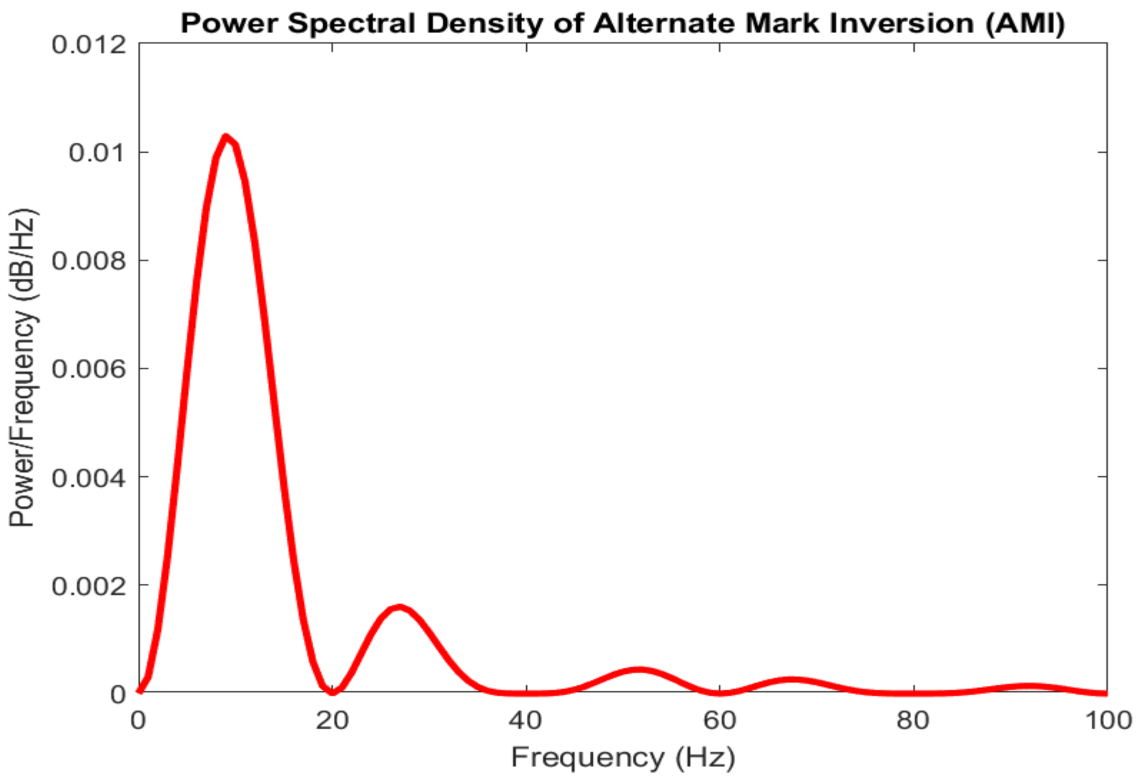
Manchester Code
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Amplitude
0
10
20
30
40
50
60
70
80
90
100
Time

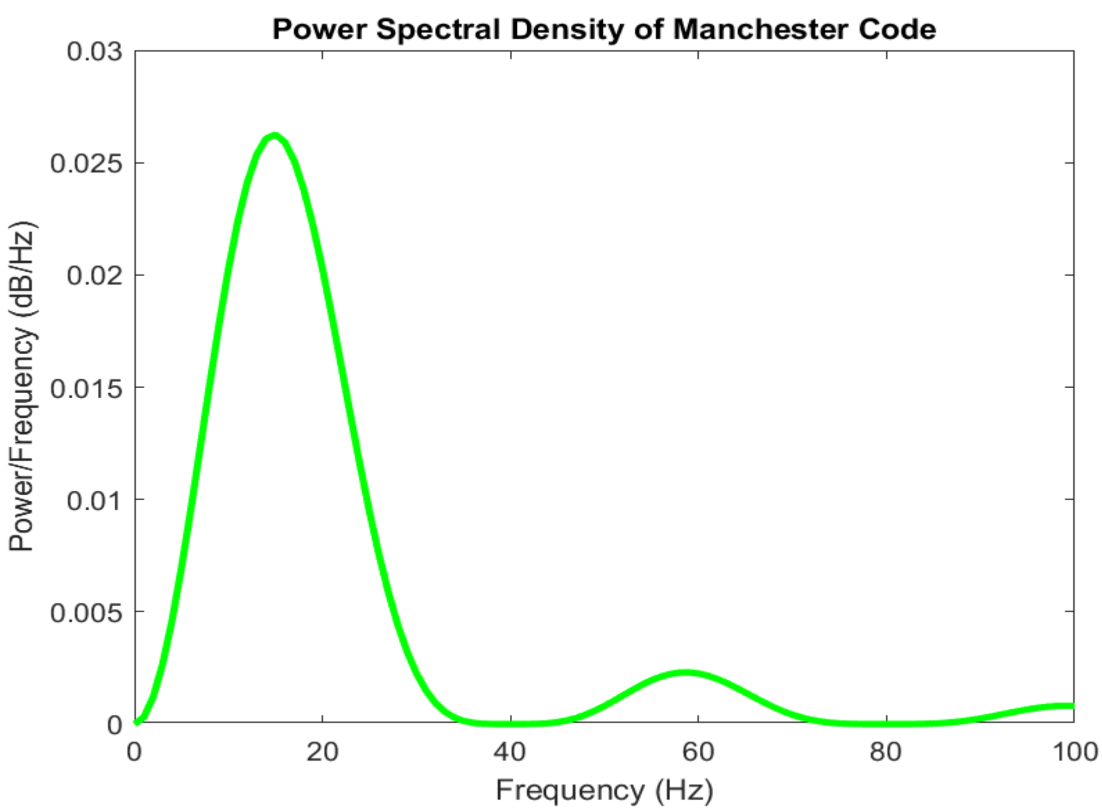

3-Level Pulse Amplitude Modulation (3-PAM)
Amplitude
Time
2
1.5
1
0.5
0
-0.5
-1
0
5
10
15
20
25
30
35
40
45
50

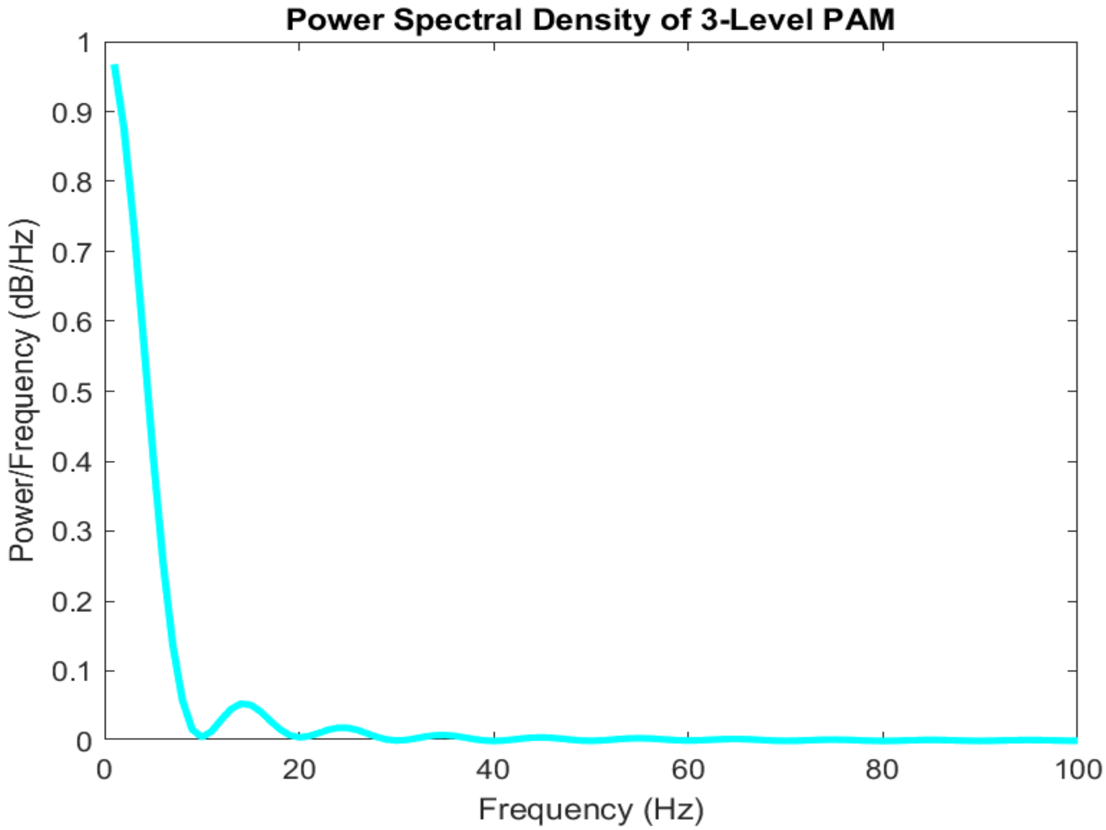

Power Spectral Density of non Return-to-Zero (NRZ)
Power/Frequency (dB/Hz)
Frequency (Hz)
0.05
0.045
0.04
0.035
0.03
0.025
0.02
0.015
0.01
0.005
0
0
20
40
60
80
100


Power Spectral Density of NRZI
Power/Frequency (dB/Hz)
0.05
0.045
0.04
0.035
0.03
0.025
0.02
0.015
0.01
0.005
0
0
20
40
60
80
100
Frequency (Hz)


Power Spectral Density of Return-to-Zero (RZ)
Power/Frequency (dB/Hz)
0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
20
40
60
80
100
Frequency (Hz)


Power Spectral Density of Alternate Mark Inversion (AMI)
Power/Frequency (dB/Hz)
0.012
0.01
0.008
0.006
0.004
0.002
0
0
20
40
60
80
100
Frequency (Hz)


Power Spectral Density of Manchester Code
Power/Frequency (dB/Hz)
0.03
0.025
0.02
0.015
0.01
0.005
0
0
20
40
60
80
100
Frequency (Hz)



Line Coding Schemes

1. Non-Return to Zero (NRZ)

- **Advantages**:
    1. Simple technique.
    2. Requires less bandwidth.
    3. Relatively immune to noise.
- **Disadvantages**:
    1. Lacks error correction.
    2. Presence of a DC component.
    3. No inherent clock signal for synchronization.

2. Non-Return to Zero Inverted (NRZI)

- **Advantages**:
    1. Supports clock recovery.
    2. Offers good synchronization.
    3. No DC component.
- **Disadvantages**:
    1. Lower bit density
    2. Limited spectral efficiency.
    3. Prone to baseline wander.

3. Return to Zero (RZ)

- **Advantages**:
    1. Better synchronization.
    2. Enables error detection.
    3. Maintains DC balance.
- **Disadvantages**:
    1. Requires high bandwidth.
    2. Increased signal power.
    3. Higher complexity.

4. Alternate Mark Inversion (AMI)

- **Advantages**:
    1. Enhanced error detection.
    2. Good bandwidth efficiency.
    3. Resistant to long runs of zeros.

- **Disadvantages**:
    1. Limited signal levels.
    2. Higher complexity.
    3. Limited data rate.

5. Manchester Coding

- **Advantages**:
    1. Self-clocking.
    2. Reliable.
    3. Supports error detection.
- **Disadvantages**:
    1. Lower data rate due to transition overhead.
    2. High power consumption.
    3. Synchronization issues.

6. Multi-Level Inverted

- **Advantages**:
    1. Improved bandwidth efficiency.
    2. Reduced transmission power.
    3. Signal rate is lower than bit rate.
- **Disadvantages**:
    1. Higher complexity due to multiple levels.
    2. More sensitive to channel distortions.

Other Common Line Codes

i) High-Density Bipolar-3 (HDB3)

- **Advantages**:
    1. Improved transmission efficiency over AMI.
    2. Good bandwidth efficiency.
- **Disadvantages**:
    1. Higher implementation complexity.
    2. Limited error detection capabilities.

ii) 4B/5B

- **Advantages**:
    1. Supports clock recovery and synchronization.
    2. Reasonably efficient data transmission rate.
- **Disadvantages**:
    1. Requires more bandwidth due to 25% overhead.
    2. Not as bandwidth efficient as some other schemes.