```
clear;
clc;

% Generate binary bipolar random sequence
L = 1000;
a = -1; % The range of the bipolar is [-1,1]
b = 1;
bits = round(a+(b-a)*rand(1,L));

% Record the position of 1 and -1 for plotting scatter
position1 = find(bits==1);
position2 = find(bits==-1);

% Let the signal traverse a AWGN channel
SNR1 = 5;
SNR2 = 15;
SNR3 = 20;
bipolar_wave_noise1 = awgn(bits, SNR1);
bipolar_wave_noise2 = awgn(bits, SNR2);
bipolar_wave_noise3 = awgn(bits, SNR3);

% Create the multipath channel h(t)
h = [0.2, 0.1, 1, 0.2, 0.1];
% Determine the number of tap and obtain the tap coefficient
N = 5;
C = force_zero(h, N);

% Signal pass through the multipath channel h(t)
rs1 = conv(bipolar_wave_noise1, h, 'same');
rs2 = conv(bipolar_wave_noise2,  h, 'same');
rs3 = conv(bipolar_wave_noise3,  h, 'same');
% Perform equalization
rs1_equalization = conv(rs1, C, 'same');
rs2_equalization = conv(rs2, C, 'same');
rs3_equalization = conv(rs3, C, 'same');

% Plot the original bipolar signal
figure(1)
set(gcf,'position', [250 200 1600 800]);
subplot(241)
plot(bits(position1), 'r.')
hold on
plot(bits(position2), 'b.')
grid on;
title("The original bipolar signal");
ylim([-3, 3]);
xlabel("time (s)")
ylabel("voltage (V)")
legend('+1V', "-1V")
% Plot the receive signal added SNR=5 AWGN before equalization
subplot(242)
plot(rs1(position1), 'r.')
```

```
hold on
plot(rs1(position2), 'b.')
grid on;
title("SNR=5 AWGN before equalization");
ylim([-3, 3]);
xlabel("time (s)")
ylabel("voltage (V)")
legend('+1V', "-1V")
% Plot the receive signal added SNR=15 AWGN before equalization
subplot(243)
plot(rs2(position1), 'r.')
hold on
plot(rs2(position2), 'b.')
grid on;
title("SNR=15 AWGN before equalization");
ylim([-3, 3]);
xlabel("time (s)")
ylabel("voltage (V)")
legend('+1V', "-1V")
% Plot the receive signal added SNR=20 AWGN before equalization and
 decision
subplot(244)
plot(rs3(position1), 'r.')
hold on
plot(rs3(position2), 'b.')
grid on;
title("SNR=20 AWGN before equalization");
ylim([-3, 3]);
xlabel("time (s)")
ylabel("voltage (V)")
legend('+1V', "-1V")
% Plot the receive signal added SNR=5 AWGN after equalization
subplot(246)
plot(rs1_equalization(position1), 'r.')
hold on
plot(rs1_equalization(position2), 'b.')
grid on;
title("SNR=5 AWGN after equalization");
ylim([-3, 3]);
xlabel("time (s)")
ylabel("voltage (V)")
legend('+1V', "-1V")
% Plot the receive signal added SNR=15 AWGN after equalization
subplot(247)
plot(rs2_equalization(position1), 'r.')
hold on
plot(rs2_equalization(position2), 'b.')
grid on;
title("SNR=15 AWGN after equalization");
ylim([-3, 3]);
xlabel("time (s)")
ylabel("voltage (V)")
legend('+1V', "-1V")
```

```
% Plot the receive signal added SNR=20 AWGN after equalization and
 decision
subplot(248)
plot(rs3_equalization(position1), 'r.')
hold on
plot(rs3_equalization(position2), 'b.')
grid on;
title("SNR=20 AWGN after equalization");
ylim([-3, 3]);
xlabel("time (s)")
ylabel("voltage (V)")
legend('+1V', "-1V")

% Upsample the signal
sps = 10;
rs1_sampled = upsample(rs1, sps);
rs2_sampled = upsample(rs2, sps);
rs3_sampled = upsample(rs3, sps);
rs1_equalization_sampled = upsample(rs1_equalization, sps);
rs2_equalization_sampled = upsample(rs2_equalization, sps);
rs3_equalization_sampled = upsample(rs3_equalization, sps);

% Use cosine roll-off waveform
alpha = 1;
span = 4;
sps = 10;
ht = rcosdesign(alpha, span, sps);
rs1_sampled = conv(rs1_sampled, ht, 'same');
rs2_sampled = conv(rs2_sampled, ht, 'same');
rs3_sampled = conv(rs3_sampled, ht, 'same');
rs1_equalization_sampled = conv(rs1_equalization_sampled, ht, 'same');
rs2_equalization_sampled = conv(rs2_equalization_sampled, ht, 'same');
rs3_equalization_sampled = conv(rs3_equalization_sampled, ht, 'same');

% Plot the eye pattern before equalization
eyediagram(rs1_sampled,sps);
title('The eye pattern of signal added noise SNR=5 before
 equalization');
eyediagram(rs2_sampled,sps);
title('The eye pattern of signal added noise SNR=15 before
 equalization');
eyediagram(rs3_sampled,sps);
title('The eye pattern of signal added noise SNR=20 before
 equalization');

% Plot the eye pattern after equalization
eyediagram(rs1_equalization_sampled,sps);
title('The eye pattern of signal added noise SNR=5 after
 equalization');
eyediagram(rs2_equalization_sampled,sps);
title('The eye pattern of signal added noise SNR=15 after
 equalization');
eyediagram(rs3_equalization_sampled,sps);
```

```
title('The eye pattern of signal added noise SNR=20 after
 equalization');

% Design zero-forcing equalizer
function [c] = force_zero(h,N)
    H = length(h);
    MID = find(h==1); % Find the center point of the h(t)
    % If h(t) is not symmetrical, zero-padding
    if(MID-1<H-MID)
        for i=1:(H-MID)-(MID-1)
            h = [0,h];
        end
    else
        for i=1:(MID-1)-(H-MID)
            h = [h,0];
        end
    end
    L = max(MID-1,H-MID);
    % Calculate the sequence of x=[(-2N),...,0,...(2N)]
    x = zeros(1,4*N+1);
    if 2*N>=L
        x((2*N+1-L:2*N+1+L))=h;
    else
        x = h((MID-2*N:MID+2*N));
    end
    % Create the matrix X=[x(0)~x(-2N);x(1)~(x-2N+1),...,x(2N)~x(0)]
    X=[];
    for i=1:2*N+1
        % Use fliplr function to flip the array
        X = [X;fliplr(x(i:2*N+i))];
    end
    % Create the target matrix y=[...0,..,0,1,0,..,0...]
    y = zeros(2*N+1,1);
    y(N+1) = 1;
    % Calculate the coefficient
    c = X^(-1)*y;
end
```

Published with MATLAB® R2021a