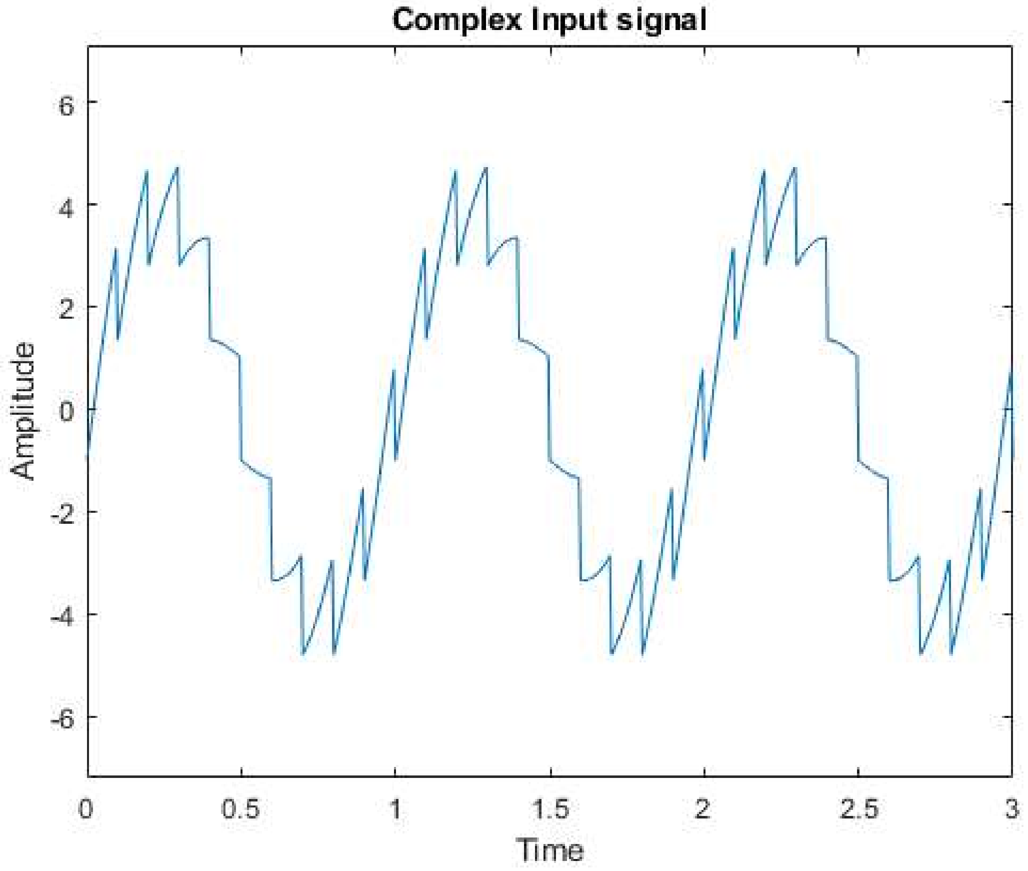
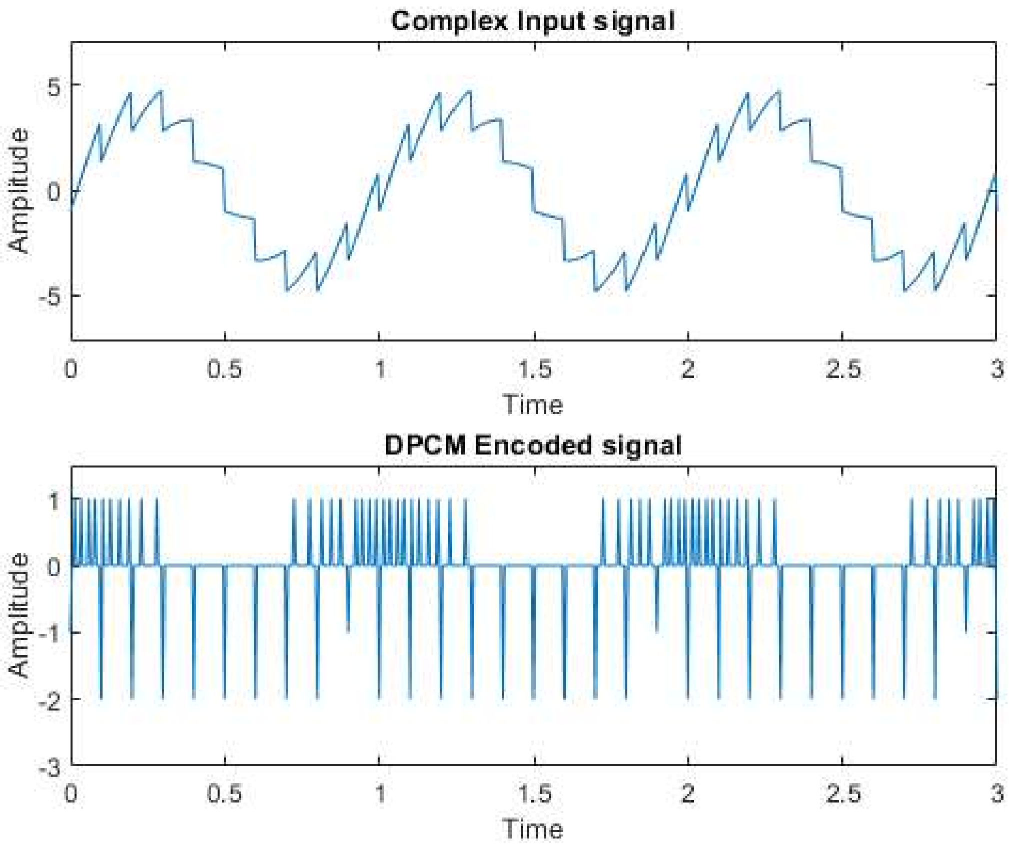
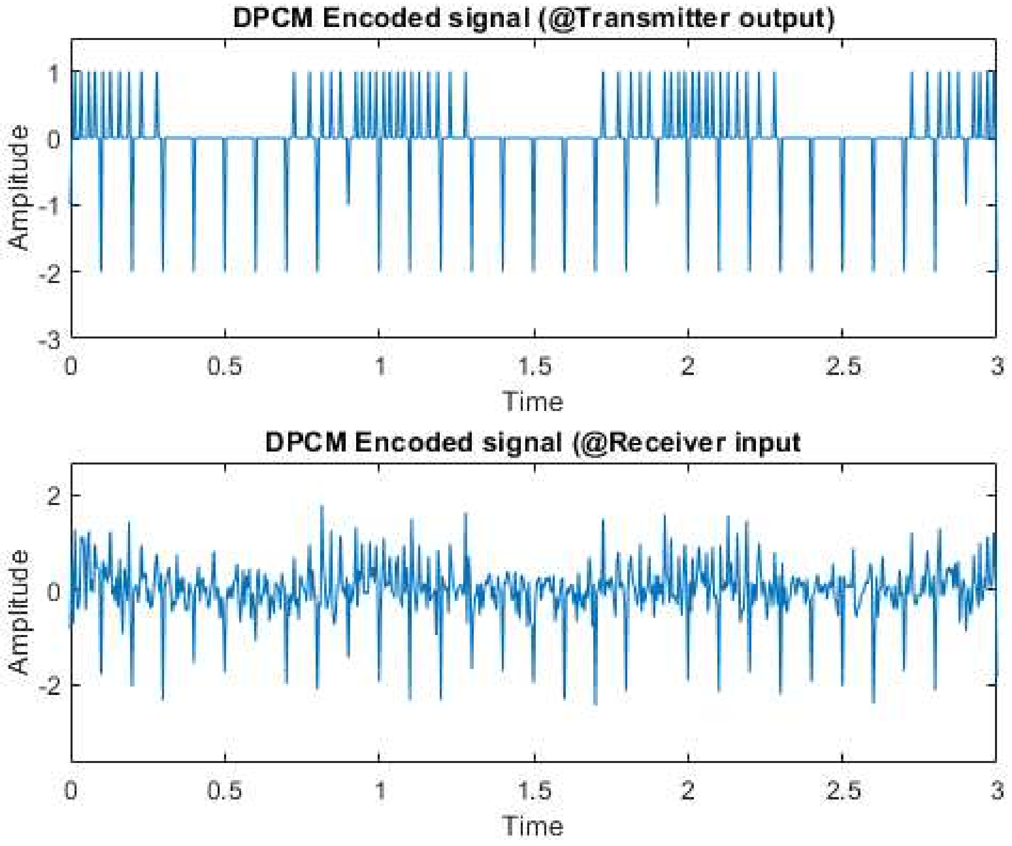
```
clc;
clear all;
close all;

%Differential Pulse COde Modulation###########
%############################################

Am1 = 4;
Am2 = 1;
fm1 = 1;
fm2 = 10;
fs = 20*fm2;
t = 0:1/fs:3;

%Input Signal;
x = Am1*sin(2*pi*fm1*t) + Am2*sawtooth(2*pi*fm2*t);

figure(); plot(t, x);
axis([min(t) max(t) (1.5*min(x)) (1.5*max(x))]);
xlabel('Time'); ylabel('Amplitude');
title('Complex Input signal');

%Transmitter Block:
for n = 1:length(x)
    if n==1                        %Same operations as below, but for 1st sample
        e(n) = x(n);
        eq(n) = round(e(n));
        xq(n) = eq(n);
    else
        e(n) = x(n) - xq(n-1);     %Difference between actual sample and predicted sample
        eq(n) = round(e(n));       %Quantizing the difference (DPCM Encoded signal)
        xq(n) = eq(n) + xq(n-1);   %Generating Prediction
    end
end

figure();
subplot(2, 1, 1); plot(t, x);
xlabel('Time'); ylabel('Amplitude');
title('Complex Input signal');
axis([min(t) max(t) (1.5*min(x)) (1.5*max(x))])

subplot(2, 1, 2); plot(t, eq);
xlabel('Time'); ylabel('Amplitude');
title('DPCM Encoded signal');
axis([min(t) max(t) (1.5*min(eq)) (1.5*max(eq))])

%Transmission Through noisy channel
eq_noisy = awgn(eq, 10);

figure();

subplot(2, 1, 1); plot(t, eq);
xlabel('Time'); ylabel('Amplitude');
title('DPCM Encoded signal (@Transmitter output)');
axis([min(t) max(t) (1.5*min(eq)) (1.5*max(eq))])
```

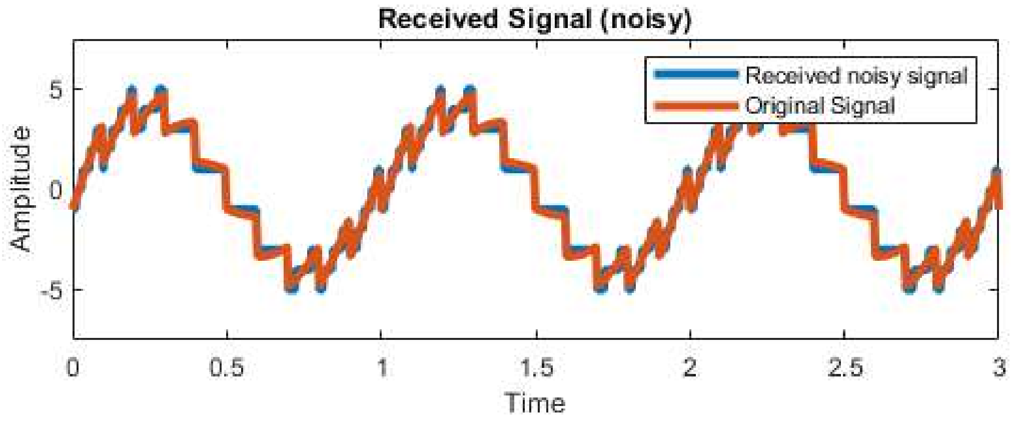
```
subplot(2, 1, 2); plot(t, eq_noisy);
xlabel('Time'); ylabel('Amplitude');
title('DPCM Encoded signal (@Receiver input');
axis([min(t) max(t) (1.5*min(eq_noisy)) (1.5*max(eq_noisy))])

%Receiver Block

%For noisy signal
for n = 1:length(eq_noisy)
    if n == 1
        recv_noisy(n) = eq(n);
    else
        recv_noisy(n) = eq(n) + recv_noisy(n-1);
    end
end

%For no noise signal
for n = 1:length(eq)
    if n == 1
        recv(n) = eq(n);
    else
        recv(n) = eq(n) + recv(n-1);
    end
end

figure();
subplot(2, 1, 1); plot(t, recv_noisy, 'LineWidth', 3);
hold on; plot(t, x, 'LineWidth', 3);
xlabel('Time'); ylabel('Amplitude');
title('Received Signal (noisy)');
legend('Received noisy signal', 'Original Signal');
axis([min(t) max(t) (1.5*min(recv_noisy)) (1.5*max(recv_noisy))])

subplot(2, 1, 2); plot(t, recv, 'LineWidth', 3);
hold on; plot(t, x, 'LineWidth', 3);
xlabel('Time'); ylabel('Amplitude');
title('Received Signal (no noise)');
legend('Received no noise signal', 'Original Signal');
axis([min(t) max(t) (1.5*min(recv)) (1.5*max(recv))])

%Filtering the signal
[num den] = butter(4, 5*fm2/fs);
filter_output_no_noise = filter(num, den, recv);
filter_output_noisy = filter(num, den, recv_noisy);

figure();
subplot(2, 1, 1); plot(t, filter_output_no_noise, 'LineWidth', 3);
hold on; plot(t, x, 'LineWidth', 3);
xlabel('Time'); ylabel('Amplitude');
title('Smoothened DPCM signal w/no noise');
legend('Smoothed no noise received signal', 'Original Signal');
axis([min(t) max(t) (1.5*min(filter_output_no_noise)) (1.5*max(filter_output_no_noise))])

subplot(2, 1, 2); plot(t, filter_output_noisy, 'LineWidth', 3);
hold on; plot(t, x, 'LineWidth', 3);
xlabel('Time'); ylabel('Amplitude');
```

```
title('Smoothened DPCM signal w/noise');
legend('Smoothed noisy received signal', 'Original Signal');
axis([min(t) max(t) (1.5*min(filter_output_noisy)) (1.5*max(filter_output_noisy))])
```

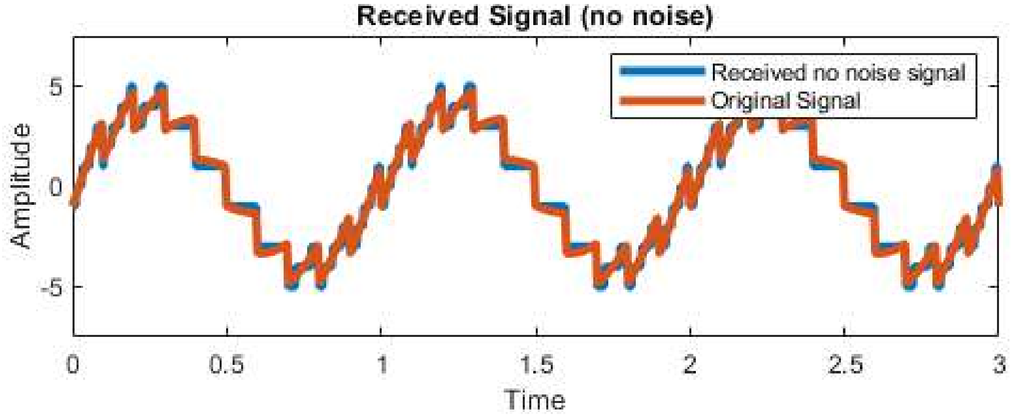



Complex Input signal
Amplitude
5
0
-5
Time
0
0.5
1
1.5
2
2.5
3
DPCM Encoded signal
Amplitude
1
0
-1
-2
-3
Time
0
0.5
1
1.5
2
2.5
3

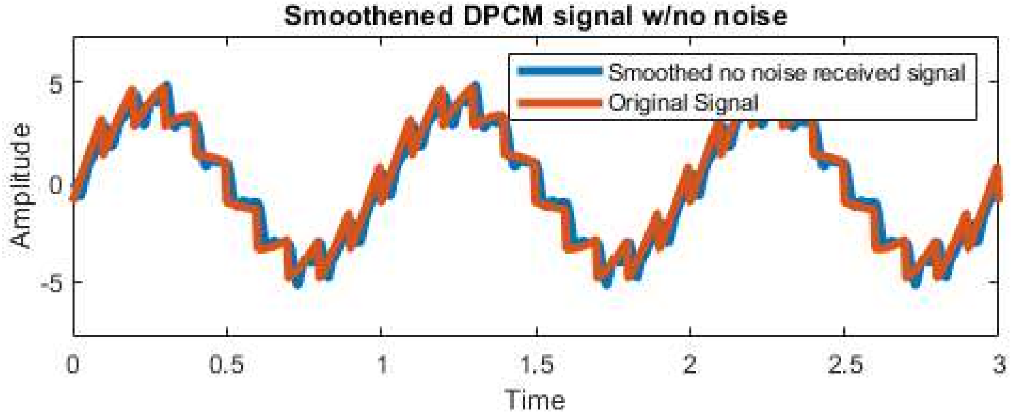

DPCM Encoded signal (@Transmitter output)
Amplitude
1
0
-1
-2
-3
Time
0
0.5
1
1.5
2
2.5
3
DPCM Encoded signal (@Receiver input
Amplitude
2
0
-2
Time
0
0.5
1
1.5
2
2.5
3



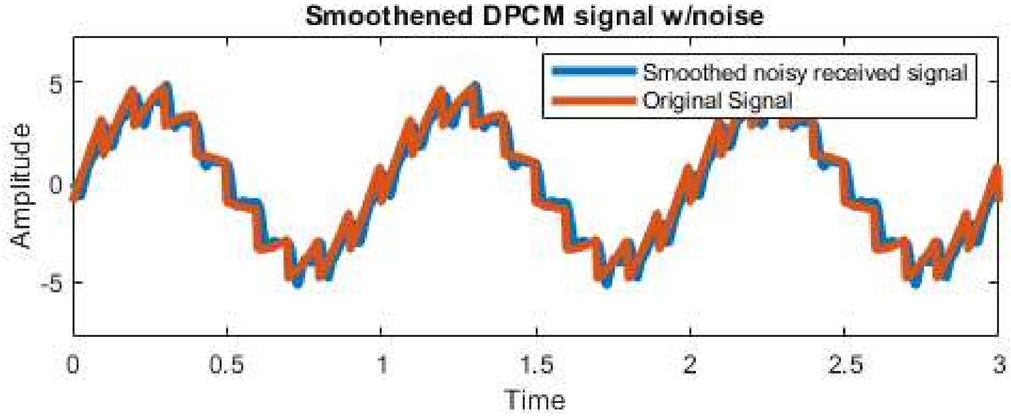
Received Signal (noisy)



Received Signal (no noise)



Smoothened DPCM signal w/no noise



Smoothened DPCM signal w/noise

Published with MATLAB® R2019b