Table of Contents


```
function [bpsk_optimum, bpsk_theory,  bpsk_unideal] = BPSK(N, data, E,
 M)
```

Transmitter

```
    s = pskmod(data,2) ; %Binary PSK
    data_seq = zeros(1,N * M) ; %Pre allocating for Date Sequence
    counter = 1; %Counter on data seq array
    for bit_counter = 1 : N
        for sym_counter = 1 : M
            data_seq(1,counter) = s(bit_counter);
            counter =counter + 1;
        end
    end
    figure(1)
    subplot(311)
    pwelch(data_seq)
    title("BPSK Power Specteral")
    grid on;
%     xlabel('Frequency (Hz)')
%     ylabel('Power (dB)')
    legend('BPSK PSD')
```

```
Not enough input arguments.

Error in BPSK (line 3)
    s = pskmod(data,2) ; %Binary PSK
```

Pwelch

```
clc;
    [pxx,f] = pwelch(data_seq,[],[],[],1000,'centered','power');
```

```
figure(2)
subplot(311)
plot(f,pow2db(pxx))
title("BPSK Power Specteral")
grid on;
xlabel('Frequency (Hz)')
ylabel('Power (dB)')
legend('PSD of BPSK')
```

Noise add by Channel

```
clc;
    n = randn(1,length(data_seq))+
 1i*randn(1,length(data_seq)); %noise
    r = sqrt(E / M) * data_seq + n ; %received Signal with Noise
    r0 = sqrt(E(120,1) / M) * data_seq + n;
```

Scatter

```
clc;
    figure(3)
    subplot(311)
    scatter(real(r(12,15:250)) , imag(r(12,15:250)),'m');
    title("BPSK Constellation")
    grid on;
    legend('BPSK Cons')
    xlabel('Real Part')
    ylabel('Imag Part')
```

Scatter Plot

```
    scatterplot(r0);
    title("BPSK Constellation")
    grid on;
    legend('BPSK Cons')
    xlabel('Real Part')
    ylabel('Imag Part')
```

Demodulation

```
    h = ones(1,M) / M ; % Moving Average
    y = zeros(size(E,1), size(r,2) + M - 1); %preallocating
    for counter = 1 : size(E,1) %E matrix 1st row
        y(counter, :) = conv(r(counter, :), h) ; %convolution on 130
 arrays
    end
```

Decision Making (Optimum point Selection)

```
clc;
```

```
    temp = zeros(size(E,1) , N ) ; %Preallocating
    for row = 1 : size(E, 1)
        for column = 1 : N
            temp(row, column) = y(row, column * M); %Optimum point
 Selection
        end
    end
```

BER Calculation

```
    y_normal = sign(real(temp)) ;
    br = pskdemod(y_normal,2) ; %Demodulation
    pe = zeros(size(E, 1), 1) ; %Preallocating for Pr of error
    clc;
    for row_counter = 1 : size(E,1)
        for column_counter = 1 : N
           if br(row_counter,column_counter) ~= data(column_counter)
               pe(row_counter) = pe(row_counter) + 1 ;
           end
        end
    end
```

Return

```
    bpsk_optimum = pe' / N;
    bpsk_theory = qfunc(sqrt(E)) ;
```

Off_Set Frequency

Demodulation

```
    clc;
    h = ones(1,M - 1) / M ; % Moving Average with 1 sample delay
    y = zeros(size(E,1), size(r,2) + M - 2); %preallocating
    for counter = 1 : size(E,1) %E matrix 1st row
        y(counter, :) = conv(r(counter, :), h) ; %convolution on 130
 arrays
    end
```

Decision Making (Optimum point Selection)

```
clc;
    temp = zeros(size(E,1) , N ) ; %Preallocating
    for row = 1 : size(E, 1)
        for column = 1 : N
            temp(row, column) = y(row, column * M - 1); %Optimum point
 Selection
        end
    end
```

BER Calculation

```
        y_normal = sign(real(temp)) ;
        br = pskdemod(y_normal,2) ; %Demodulation
        pe = zeros(size(E, 1), 1) ; %Preallocating for Pr of error
        clc;
        for row_counter = 1 : size(E,1)
            for column_counter = 1 : N
               if br(row_counter,column_counter) ~= data(column_counter)
                   pe(row_counter) = pe(row_counter) + 1 ;
               end
            end
        end
```

Return

```
        bpsk_unideal = pe' / N;

    end
```

Published with MATLAB® R2020b