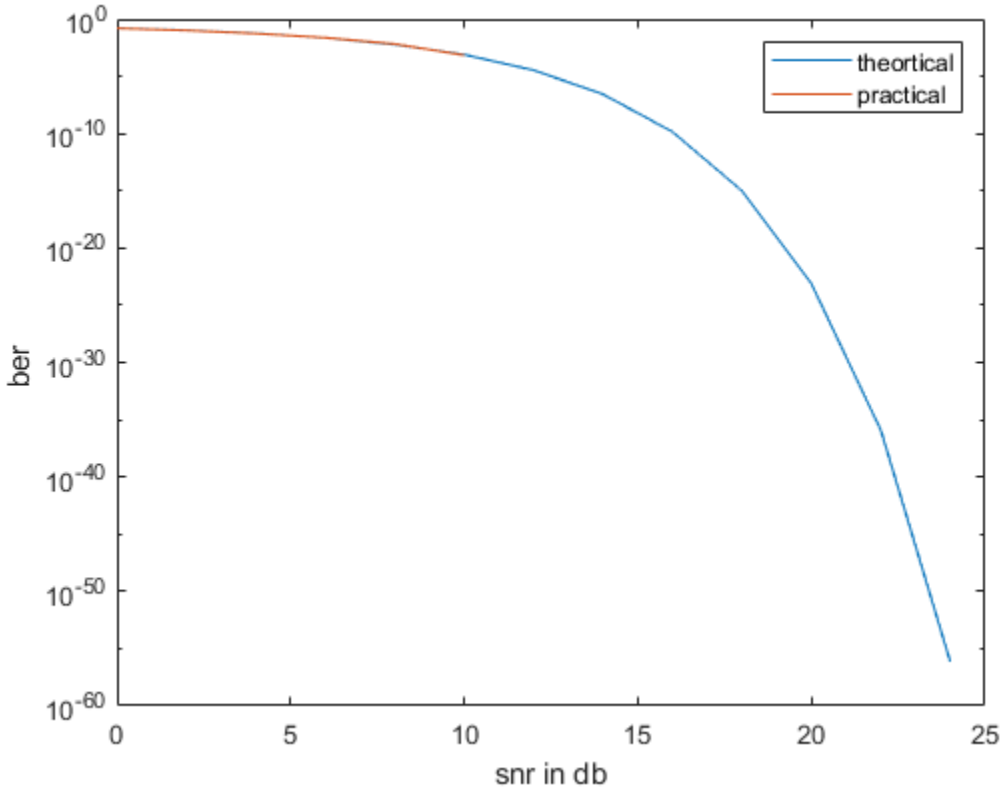
```
clc;
clear all;
close all;
N=10000;

r = randi([0,1],N,1);

for i=1:N                       %converted to bpsk

        if(r(i)==0)
            r(i) = -1;
        else
             r(i) = 1;
        end


end

n = randn(N,1);                  %noise signal


snr_db = 0:2:24;
kk=1;% snr in db

for k = 1: length(snr_db)

snr_linear = 10.^(snr_db(k)/10);           % converted snr to linear
sigma = 1./(snr_linear).^(1/2);                % find sigma

y = r+ sigma.*n;                  % find y = bpsk_signal + sigma*noise


% convert y sequence into bpsk take threshold value = 0
% z is your constructed signal
% y is output signal with noise
for j=1:N
   if(y(j)<0)
            z(j,1) = -1;
        else
            z(j,1) = 1;
        end

end

% check bit by bit that r and z is same or not

count_error=0;
for jj=1:N
    if(r(jj)~=z(jj))
        count_error=count_error+1;
%     else
%         count_error;
```

```
    end
end

% calculate theortical ber  Q((snr_linear)^(1/2)


ber_th(k) = qfunc((snr_linear).^(1/2));
ber_prac(k) = count_error/N;
%k=k+1;
end
%snr_linear1 = 10.^(snr_db/10)  ;

semilogy(snr_db, ber_th);
xlabel("snr in db");
ylabel("ber ");

hold on
semilogy(snr_db, ber_prac);
xlabel("snr in db");
ylabel("ber ");

legend('theortical','practical')
```



Published with MATLAB® R2018b