LAMPIRAN A:
LIST PROGRAM

```
% Program simulasi MC-CDMA%

close all
clear all,clc
tic;
%
=========================================================
code_rate          = 2/3;              % 1/2 , 2/3, 3/4
modulasi          = 16;                % [16 atau 64] QAM
interlv             = 2;                 % [1]=Conv. Int, [2]=Block Int

panjang_GI = 1/16;
Nfft=64;
jml_subcar=48;
EbNo_max = 15;
EbNo=0:1:EbNo_max;

% ***************************** Pengirim ***************************
% Data
if modulasi == 16
    if code_rate == 1/2
         jml_itr = 2*100;
        data=randint(1,3828/2,2);
    elseif code_rate == 2/3
        jml_itr = 2*100;
        data=randint(1,5104/2,2);
    elseif code_rate == 3/4
        jml_itr = 2*100;
        data=randint(1,5742/2,2);
    end
elseif modulasi == 64
    if code_rate == 1/2
        jml_itr = 2*100;
        data=randint(1,5748/2,2);
    elseif code_rate == 2/3
        jml_itr = 2*100;
        data=randint(1,7664/2,2);
    elseif code_rate == 3/4
        jml_itr = 2*100;
        data=randint(1,8622/2,2);
    end
```

```
end
jml_itr = 2*jml_itr;

% CC Encoder
if code_rate == 1/2
   % datane = 1914 - 2874
   gen_pol=poly2trellis(7,[171 133]);
   codeword=convenc(data,gen_pol); % 3828 - 5748
elseif code_rate == 2/3
   % datane = 2552 - 3832
   gen_pol=poly2trellis(7,[171 133]);
   codeword=convenc(data,gen_pol); % 5104 - 7664
   codeword(4:4:end)=[]; % Puncture by removing every third value. % 3828 - 5748
elseif code_rate == 3/4
   % datane = 2871 - 4311
   gen_pol=poly2trellis(7,[171 133]);
   codeword=convenc(data,gen_pol); % 5742 - 8622
   codeword(3:3:end)=[]; % Puncture by removing every third value. % 3828 - 5748
end

% Interleaver
if interlv == 1 % Convolutional
   nrows = 3; slope = 2; % Interleaver parameters
   D = nrows*(nrows-1)*slope; % Delay of interleaver/deinterleaver pair
   codeword_padded = [codeword.'; zeros(D,1)]; % Pad x at the end before
interleaving.
   codeword2 = convintrlv(codeword_padded,nrows,slope); % Interleave padded
data.
   out_int = [-2*codeword2 + 1].'; % 3840 - 5760
elseif interlv == 2 % Block
   codeword2 = [codeword zeros(1,12)];
   A = 16;
   B = length(codeword2)/A;
   for n=0:length(codeword2)/B - 1
      int_sym(n+1,1:B)=codeword2(1,n*B+1:(n+1)*B);
      n=n+1;
   end
   [p q]=size(int_sym);
   out_int = -2*reshape(int_sym,1,p*q) + 1; % 3840 - 5760
end

if modulasi == 16
   % Modulasi 16QAM
   jml_bit_simbol = 4;
```

```
    pjg_data=length(out_int);
    for p = 1:pjg_data/jml_bit_simbol;
        if    out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 -1 -1 -1]
            in_map(p) = 0;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 -1 -1 1]
            in_map(p) = 1;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 -1 1 -1]
            in_map(p) = 2;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 -1 1 1]
            in_map(p) = 3;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 1 -1 -1]
            in_map(p) = 4;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 1 -1 1]
            in_map(p) = 5;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 1 1 -1]
            in_map(p) = 6;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 1 1 1]
            in_map(p) = 7;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[1 -1 -1 -1]
            in_map(p) = 8;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[1 -1 -1 1]
            in_map(p) = 9;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[1 -1 1 -1]
            in_map(p) = 10;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[1 -1 1 1]
            in_map(p) = 11;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[1 1 -1 -1]
            in_map(p) = 12;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[1 1 -1 1]
            in_map(p) = 13;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[1 1 1 -1]
            in_map(p) = 14;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[1 1 1 1]
            in_map(p) = 15;
        end
    end
    data_mod=qammod(in_map,16); % 960
elseif modulasi == 64
    % Modulasi 64QAM
    jml_bit_simbol = 6;
    pjg_data=length(out_int);
    for p = 1:pjg_data/jml_bit_simbol;
        if    out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 -1 -1 -1 -1 -1]
            in_map(p) = 0;
```

```
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 -1 -1 -1 -1  1]
    in_map(p) = 1;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 -1 -1 -1  1 -1]
    in_map(p) = 2;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 -1 -1 -1  1  1]
    in_map(p) = 3;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 -1 -1  1 -1 -1]
    in_map(p) = 4;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 -1 -1  1 -1  1]
    in_map(p) = 5;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 -1 -1  1  1 -1]
    in_map(p) = 6;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 -1 -1  1  1  1]
    in_map(p) = 7;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 -1  1 -1 -1 -1]
    in_map(p) = 8;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 -1  1 -1 -1  1]
    in_map(p) = 9;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 -1  1 -1  1 -1]
    in_map(p) = 10;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 -1  1 -1  1  1]
    in_map(p) = 11;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 -1  1  1 -1 -1]
    in_map(p) = 12;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 -1  1  1 -1  1]
    in_map(p) = 13;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 -1  1  1  1 -1]
    in_map(p) = 14;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1 -1  1  1  1  1]
    in_map(p) = 15;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1  1 -1 -1 -1 -1]
    in_map(p) = 16;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1  1 -1 -1 -1  1]
    in_map(p) = 17;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1  1 -1 -1  1 -1]
    in_map(p) = 18;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1  1 -1 -1  1  1]
    in_map(p) = 19;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1  1 -1  1 -1 -1]
    in_map(p) = 20;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1  1 -1  1 -1  1]
    in_map(p) = 21;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1  1 -1  1  1 -1]
    in_map(p) = 22;
```

```
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1  1 -1  1  1  1]
    in_map(p) = 23;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1  1  1 -1 -1 -1]
    in_map(p) = 24;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1  1  1 -1 -1  1]
    in_map(p) = 25;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1  1  1 -1  1 -1]
    in_map(p) = 26;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1  1  1 -1  1  1]
    in_map(p) = 27;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1  1  1  1 -1 -1]
    in_map(p) = 28;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1  1  1  1 -1  1]
    in_map(p) = 29;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1  1  1  1  1 -1]
    in_map(p) = 30;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[-1  1  1  1  1  1]
    in_map(p) = 31;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1 -1 -1 -1 -1 -1]
    in_map(p) = 32;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1 -1 -1 -1 -1  1]
    in_map(p) = 33;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1 -1 -1 -1  1 -1]
    in_map(p) = 34;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1 -1 -1 -1  1  1]
    in_map(p) = 35;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1 -1 -1  1 -1 -1]
    in_map(p) = 36;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1 -1 -1  1 -1  1]
    in_map(p) = 37;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1 -1 -1  1  1 -1]
    in_map(p) = 38;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1 -1 -1  1  1  1]
    in_map(p) = 39;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1 -1  1 -1 -1 -1]
    in_map(p) = 40;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1 -1  1 -1 -1  1]
    in_map(p) = 41;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1 -1  1 -1  1 -1]
    in_map(p) = 42;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1 -1  1 -1  1  1]
    in_map(p) = 43;
elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1 -1  1  1 -1 -1]
    in_map(p) = 44;
```

```
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1 -1  1  1 -1  1]
            in_map(p) = 45;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1 -1  1  1  1 -1]
            in_map(p) = 46;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1 -1  1  1  1  1]
            in_map(p) = 47;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1  1 -1 -1 -1 -1]
            in_map(p) = 48;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1  1 -1 -1 -1  1]
            in_map(p) = 49;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1  1 -1 -1  1 -1]
            in_map(p) = 50;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1  1 -1 -1  1  1]
            in_map(p) = 51;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1  1 -1  1 -1 -1]
            in_map(p) = 52;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1  1 -1  1 -1  1]
            in_map(p) = 53;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1  1 -1  1  1 -1]
            in_map(p) = 54;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1  1 -1  1  1  1]
            in_map(p) = 55;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1  1  1 -1 -1 -1]
            in_map(p) = 56;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1  1  1 -1 -1  1]
            in_map(p) = 57;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1  1  1 -1  1 -1]
            in_map(p) = 58;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1  1  1 -1  1  1]
            in_map(p) = 59;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1  1  1  1 -1 -1]
            in_map(p) = 60;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1  1  1  1 -1  1]
            in_map(p) = 61;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1  1  1  1  1 -1]
            in_map(p) = 62;
        elseif out_int(1,jml_bit_simbol*(p-1)+1:jml_bit_simbol*p)==[ 1  1  1  1  1  1]
            in_map(p) = 63;
        end
    end
    data_mod=qammod(in_map,64); % 960
end
```

```
% S to P untuk OFDM
a = length(data_mod);
b = jml_subcar;
c = a/b;
for p = 0:b-1
   out_sp(p+1,:)=data_mod(p*c+1:c*(p+1));
end

% Penambahan Pilot dan Zero Padding
dataout_pilot_zp=zeros(Nfft,c); % 12 Zero Padding ada di sini
dataout_pilot_zp(2:7,:)   = out_sp(1:6,:);
dataout_pilot_zp(8,:)     = repmat(1+i,1,c);
dataout_pilot_zp(9:21,:)  = out_sp(7:19,:);
dataout_pilot_zp(22,:)    = repmat(1-i,1,c);
dataout_pilot_zp(23:27,:) = out_sp(20:24,:);
dataout_pilot_zp(39:42,:) = out_sp(25:28,:);
dataout_pilot_zp(43,:)    = repmat(1+i,1,c);
dataout_pilot_zp(44:56,:) = out_sp(29:41,:);
dataout_pilot_zp(57,:)    = repmat(1-i,1,c);
dataout_pilot_zp(58:64,:) = out_sp(42:48,:);

% IFFT
data_ifft=ifft(dataout_pilot_zp,64);

% Penambahan Guard Interval
[g,h]=size(data_ifft);
simbol_GI=floor(panjang_GI*h);
data_ifft_insert_GI=[data_ifft(:,h-(simbol_GI-1):end) data_ifft];
[g,ukuran_frame]=size(data_ifft_insert_GI);

% Paralel to Serial
out_ifft_serial=[];
for k=1:ukuran_frame
   data_x=data_ifft_insert_GI(:,k).';
   out_ifft_serial=[out_ifft_serial data_x];
end
input_kanal=out_ifft_serial;

input_kanal_gsr=in_kanal_proses(input_kanal); % Pendelayan sinyal

% Iterasinya
for jum=1:jml_itr
```

```
    H=kanal_transmisi(data_ifft_insert_GI,Ntx*Nrx); % Pembangkitan koefisien kanal
    for z = 1:size(data_ifft_insert_GI,1)*size(data_ifft_insert_GI,2)
        HS(z)=H(z,:)*input_kanal_gsr(:,z);
    end

    % EbNo
    for energi = 1:length(EbNo)


        % Terkena AWGN
        No = noisenya(modulasi,EbNo(energi)); rata = mean(abs(HS));
        noise = rata*[randn(size(HS,1),size(HS,2)) +
i*randn(size(HS,1),size(HS,2))]*No;
        sinyal_terima1 = HS + noise;
        %sinyal_terima1=awgn(input_kanal,EbNo(energi),'measured'); % Noise
Addition

        % ************************** Penerima **************************

        % serial to paralel OFDM
        for p = 0:size(data_ifft_insert_GI,2)-1
            sinyal_terima2(p+1,:) = sinyal_terima1(1,64*p+1:64*(p+1));
        end
        sinyal_terima=sinyal_terima2.';

        % Remove Guard Interval
        sinyal_terima(:,1:simbol_GI)=[];

        % FFT
        data_fft=fft(sinyal_terima,64);

        % Estimasi Kanal
        % Pengambilan sinyal pilot
        fft_data=data_fft.';
        ambil_pilot(:,1)=fft_data(:, 8);
        ambil_pilot(:,2)=fft_data(:,22);
        ambil_pilot(:,3)=fft_data(:,43);
        ambil_pilot(:,4)=fft_data(:,57);

        % Proses estimasi & interpolasi koef kanal based on received pilot
        pilot=[1+i 1-i 1+i 1-i];
        [g,h]=size(ambil_pilot);
        for k=1:g
            koef_estimasi(k,:)=ambil_pilot(k,:)./pilot; % proses estimasi kanal
            % Mulai Proses interpolasi koefisien kanal
            x=1:h;
            y_r=real(koef_estimasi(k,:));
            xx_r=linspace(1,g,64); % 53 adalah total sc tiap subkanal
            yy_r=spline(x,y_r,xx_r);
            y_i=imag(koef_estimasi(k,:));
            xx_i=linspace(1,g,64);
            yy_i=spline(x,y_i,xx_i);
            hasil_estimasi(k,:)=yy_r+(i*yy_i);
        end

        % Kompensasi terhadap pengaruh kanal
        simbol_est=[fft_data./hasil_estimasi].';

        % Remove Pilot dan Zero Padding
        rem_pilotzp(1:6,:)   = simbol_est(2:7,:);
        rem_pilotzp(7:19,:)  = simbol_est(9:21,:);
        rem_pilotzp(20:24,:) = simbol_est(23:27,:);
        rem_pilotzp(25:28,:) = simbol_est(39:42,:);
        rem_pilotzp(29:41,:) = simbol_est(44:56,:);
        rem_pilotzp(42:48,:) = simbol_est(58:64,:);

        % P to S untuk de-OFDM
        for r = 0:jml_subcar-1
            out_ps(r*size(data_ifft,2)+1:size(data_ifft,2)*(r+1))=rem_pilotzp(r+1,:);
        end
        out_ps2 = out_ps.';

        if modulasi == 16
            % Demodulasi 16QAM
            out_demap = qamdemod(out_ps2,16);
            for q = 1:length(out_demap)
                if     out_demap(q) == 0
                    out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)= [-1 -1 -1 -1];
                elseif out_demap(q) == 1
                    out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)= [-1 -1 -1 1];
                elseif out_demap(q) == 2
                    out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)= [-1 -1 1 -1];
                elseif out_demap(q) == 3
                    out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)= [-1 -1 1 1];
                elseif out_demap(q) == 4
                    out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)= [-1 1 -1 -1];
                elseif out_demap(q) == 5
```

```
         out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)= [-1 1 -1 1];
      elseif out_demap(q) == 6
         out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)= [-1 1 1 -1];
      elseif out_demap(q) == 7
         out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)= [-1 1 1 1];
      elseif out_demap(q) == 8
         out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)= [1 -1 -1 -1];
      elseif out_demap(q) == 9
         out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)= [1 -1 -1 1];
      elseif out_demap(q) == 10
         out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)= [1 -1 1 -1];
      elseif out_demap(q) == 11
         out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)= [1 -1 1 1];
      elseif out_demap(q) == 12
         out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)= [1 1 -1 -1];
      elseif out_demap(q) == 13
         out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)= [1 1 -1 1];
      elseif out_demap(q) == 14
         out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)= [1 1 1 -1];
      elseif out_demap(q) == 15
         out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)= [1 1 1 1];
      end
   end
elseif modulasi == 64
   % Demodulasi 64QAM
   out_demap = qamdemod(out_ps2,64);
   for q = 1:length(out_demap)
      if     out_demap(q) == 0
    out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1 -1 -1 -1 -1 -1];
      elseif out_demap(q) == 1
     out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1 -1 -1 -1 -1  1];
      elseif out_demap(q) == 2
     out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1 -1 -1 -1  1 -1];
      elseif out_demap(q) == 3
     out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1 -1 -1 -1  1  1];
      elseif out_demap(q) == 4
     out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1 -1 -1  1 -1 -1];
      elseif out_demap(q) == 5
     out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1 -1 -1  1 -1  1];
      elseif out_demap(q) == 6
     out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1 -1 -1  1  1 -1];
      elseif out_demap(q) == 7
     out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1 -1 -1  1  1  1];
      elseif out_demap(q) == 8
```

```
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1 -1  1 -1 -1 -1];
 elseif out_demap(q) == 9
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1 -1  1 -1 -1  1];
 elseif out_demap(q) == 10
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1 -1  1 -1  1 -1];
 elseif out_demap(q) == 11
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1 -1  1 -1  1  1];
 elseif out_demap(q) == 12
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1 -1  1  1 -1 -1];
 elseif out_demap(q) == 13
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1 -1  1  1 -1  1];
 elseif out_demap(q) == 14
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1 -1  1  1  1 -1];
 elseif out_demap(q) == 15
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1 -1  1  1  1  1];
 elseif out_demap(q) == 16
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1  1 -1 -1 -1 -1];
 elseif out_demap(q) == 17
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1  1 -1 -1 -1  1];
 elseif out_demap(q) == 18
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1  1 -1 -1  1 -1];
 elseif out_demap(q) == 19
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1  1 -1 -1  1  1];
 elseif out_demap(q) == 20
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1  1 -1  1 -1 -1];
 elseif out_demap(q) == 21
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1  1 -1  1 -1  1];
 elseif out_demap(q) == 22
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1  1 -1  1  1 -1];
 elseif out_demap(q) == 23
 out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1  1 -1  1  1  1];
 elseif out_demap(q) == 24
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1  1  1 -1 -1 -1];
 elseif out_demap(q) == 25
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1  1  1 -1 -1  1];
 elseif out_demap(q) == 26
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1  1  1 -1  1 -1];
 elseif out_demap(q) == 27
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1  1  1 -1  1  1];
 elseif out_demap(q) == 28
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1  1  1  1 -1 -1];
 elseif out_demap(q) == 29
 out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1  1  1  1 -1  1];
 elseif out_demap(q) == 30
 out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1  1  1  1  1 -1];
 elseif out_demap(q) == 31
 out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[-1  1  1  1  1  1];
 elseif out_demap(q) == 32
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1 -1 -1 -1 -1 -1];
 elseif out_demap(q) == 33
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1 -1 -1 -1 -1  1];
 elseif out_demap(q) == 34
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1 -1 -1 -1  1 -1];
 elseif out_demap(q) == 35
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1 -1 -1 -1  1  1];
 elseif out_demap(q) == 36
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1 -1 -1  1 -1 -1];
 elseif out_demap(q) == 37
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1 -1 -1  1 -1  1];
 elseif out_demap(q) == 38
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1 -1 -1  1  1 -1];
 elseif out_demap(q) == 39
 out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1 -1 -1  1  1  1];
 elseif out_demap(q) == 40
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1 -1  1 -1 -1 -1];
 elseif out_demap(q) == 41
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1 -1  1 -1 -1  1];
 elseif out_demap(q) == 42
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1 -1  1 -1  1 -1];
 elseif out_demap(q) == 43
 out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1 -1  1 -1  1  1];
 elseif out_demap(q) == 44
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1 -1  1  1 -1 -1];
 elseif out_demap(q) == 45
 out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1 -1  1  1 -1  1];
 elseif out_demap(q) == 46
 out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1 -1  1  1  1 -1];
 elseif out_demap(q) == 47
 out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1 -1  1  1  1  1];
 elseif out_demap(q) == 48
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1  1 -1 -1 -1 -1];
 elseif out_demap(q) == 49
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1  1 -1 -1 -1  1];
 elseif out_demap(q) == 50
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1  1 -1 -1  1 -1];
 elseif out_demap(q) == 51
out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1  1 -1 -1  1  1];
 elseif out_demap(q) == 52
```

```
       out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1  1 -1  1 -1 -1];
        elseif out_demap(q) == 53
        out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1  1 -1  1 -1  1];
        elseif out_demap(q) == 54
        out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1  1 -1  1  1 -1];
        elseif out_demap(q) == 55
       out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1  1 -1  1  1  1];
        elseif out_demap(q) == 56
       out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1  1  1 -1 -1 -1];
        elseif out_demap(q) == 57
       out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1  1  1 -1 -1  1];
        elseif out_demap(q) == 58
        out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1  1  1 -1  1 -1];
        elseif out_demap(q) == 59
        out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1  1  1 -1  1  1];
        elseif out_demap(q) == 60
        out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1  1  1  1 -1 -1];
        elseif out_demap(q) == 61
       out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1  1  1  1 -1  1];
        elseif out_demap(q) == 62
        out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1  1  1  1  1 -1];
        elseif out_demap(q) == 63
       out_demod(1,jml_bit_simbol*(q-1)+1:jml_bit_simbol*q)=[ 1  1  1  1  1  1];
        end
    end
end

% DeInterleaver
if interlv == 1 % Convolutional
    P = out_demod.';
    de_int = convdeintrlv(P,nrows,slope);
    out_de_int = de_int(D+1:end); % Remove first D symbols.
    out_de_int = out_de_int.';
elseif interlv == 2 % Block
    P = out_demod;
    for n=0:length(P)/A - 1
        de_int(n+1,1:A)=P(1,n*A+1:(n+1)*A);
        n=n+1;
    end
    [t u]=size(de_int);
    out_deint = reshape(de_int,1,t*u);
    out_de_int=out_deint(1:end-12);
end
```

```
        % CC Decoder
        if code_rate == 1/2
            %out_de_int = (out_de_int-1)/(-2);
            datane_topi = vitdec(out_de_int,gen_pol,96,'trunc','unquant'); % Decode.
        elseif code_rate == 2/3
            out_decod_conv = zeros(1,2*length(data)); % Zeros represent inserted data.
            out_decod_conv(1:4:end) = out_de_int(1:3:end); % Write actual data.
            out_decod_conv(2:4:end) = out_de_int(2:3:end); % Write actual data.
            out_decod_conv(3:4:end) = out_de_int(3:3:end); % Write actual data.
            datane_topi= vitdec(out_decod_conv,gen_pol,96,'trunc','unquant'); % Decode.
        elseif code_rate == 3/4
            out_decod_conv = zeros(1,2*length(data)); % Zeros represent inserted data.
            out_decod_conv(1:3:end) = out_de_int(1:2:end); % Write actual data.
            out_decod_conv(2:3:end) = out_de_int(2:2:end); % Write actual data.
            datane_topi= vitdec(out_decod_conv,gen_pol,96,'trunc','unquant'); % Decode.
        end

        data_topi = datane_topi;

        % Perhitungan Bit Salah
        bit_error(1,energi) = sum(data~=data_topi);

        Looping_EbNo_BitError = [jum EbNo(energi) bit_error(1,energi)
cw_error(1,energi)]
        if EbNo(energi) == max(EbNo);
        end
    end
    total_bit_error(jum,:) = bit_error;
end
load data2
pe_bit = sum(total_bit_error)/(length(data)*jml_itr);
pe_bit_ici=q1+pe_bit;
figure;
semilogy(EbNo,pe_bit,'b-'); hold on
semilogy(EbNo,pe_bit_ici,'r:*'); grid
toc
waktu_jam=toc/3600;
save Rayl_1x1_16_1per2 pe_bit pe_bit_ici EbNo waktu_jam
```

```
% fungsi pembangkitan kode Gold

function goldseq=gold_seq(fbconnection1,fbconnection2)
mseq1=m_sequence(fbconnection1);
mseq2=m_sequence(fbconnection2);
N=2^length(fbconnection1)-1;
for shift_amount=0:N-1
shift_mseq2=[mseq2(shift_amount+1:N) mseq2(1:shift_amount)];
goldseq(shift_amount+1,:)=mod(mseq1+shift_mseq2,2);
end

% romberg: Romberg integration quadrature

function [q, ea,iter]=romberg(func,a,b,es,maxit,varargin)
% q = romberg (func,a,b,es,maxit, p1,p2, . . . ) :
% Romberg integration
%
% input:
% func - name of function to be integrated
% a, b = integration limits
% es = desired relative error (default = 0 . 0 0 0 0 0 1 ? )
% maxit - maximum allowable iterations ( default = 30 )
% pl,p2,... = additional parameters used by func
%
% output:
% q = integral eestimate
% ea = approximate relative error ( % )
% iter = number of iterations

if nargin < 3
   error( ' at least 3 inputs required' )
end;

if nargin < 4 | isempty (es)
   es= - 0.000001;
end;

if nargin < 5 | isempty (maxit)
   maxit=50;
end;

n=1;
I(1,1)=trap(func, a , b , n , varargin{:});
iter = 0 ;
```

```
while iter < maxit
    iter=iter+1;
    n = 2 ^ iter;
    I(iter+1,1)=trap(func, a , b , n , varargin{:});

    for k = 2 :iter+1
        j = 2 + iter - k ;
        I(j,k) = (4^(k-1)*I(j+1,k-1)-I(j,k-1))/(4^(k-1)-1);
    end;

    ea=abs((I(1,iter+1)-I(2,iter))/I(1,iter+1))* 100;
    if ea <= es
        break;
    end;
end;
q = I(1,iter+1);

% function trap
function I = trap(func, a, b, n, varargin)
% trap : composite trapezoidal rule quadrature
% I = trap(func, a, b, n, p1, p 2 , . . . ) :composite trapezoidal rule
% i n p u t :
% func = name of function to be integrated
% a, b = iintegration limits
% n - number of segments ( default = 100)
% pl , p 2 , . . . - aditional parameters used by func
% output :
% I = integral estimate

if nargin< 3
    error('at least 3 input arguments required');
end;

if ~ (b>a)
    error(' upper bound must be greater than lower');
end;

if nargin < 4 | isempty(n)
    n=100;
end;

x = a;
h = (b-a)/n;
```

```
s=func(a,varargin{:});

for i = 1 : n - 1
    x = x + h ;
    s = s + 2 * func(x,varargin{:});
end;
s = s + func(b,varargin{:});
I = (b - a) * s / ( 2 * n ) ;

% Perhitungan ICI
clc;
close all;
clear;
clc;

f=@(t) (t/sqrt(2*pi))*(exp(1)^(-0.5*t^2))
a=linspace(0,1,16);
b=a+.5;

for m=1:length(a)
    [q(m) ea(m)]=romberg(f,a(m),b(m),0.0001,30);
end;

plot(q);grid
q1=q;
save data2 q1
```