

BỘ THÔNG TIN & TRUYỀN THÔNG
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG



BÁO CÁO THỰC HÀNH
MÔN THÔNG TIN DI ĐỘNG

Giảng viên : Nguyễn Viết Đảm
Họ và tên : HÀ Xuân Huy
Mã sinh viên : B19DCVT172
Nhóm : 06 <lớp cô hoa>
Tổ : 01

MỤC LỤC

Sim_MA06: Trục quan hóa nguyên lý hoạt động hệ thống truyền dẫn OFDM trên cơ sở thực hiện IFFT/FFT và chèn/khử CP	03
Sim_MA07: Mô hình hóa và mô phỏng hiệu năng BER cho hệ thống truyền dẫn BPSK-OFDM dùng mã kênh trong môi trường kênh AWGN.....	21
Sim_MA08: Mô hình hóa và mô phỏng hiệu năng SER cho hệ thống truyền dẫn OFDM trong môi trường kênh AWGN và kênh pha đa đường	30
Sim_FWC05: Mô hình hóa và mô phỏng hệ thống đa anten SVD MIMO.....	37
Sim_FWC06: Mô hình hóa và mô phỏng dung lượng của hệ thống SVD MIMO	43
Sim_FWC07: Mô hình hóa và mô phỏng dung lượng của hệ thống MIMO tương quan.....	55

Sim_MA06: Trục quan hóa nguyên lý hoạt động hệ thống truyền dẫn OFDM trên cơ sở thực hiện IFFT/FFT và chèn/khử CP

1, Mục đích và nội dung

a, Mục đích:

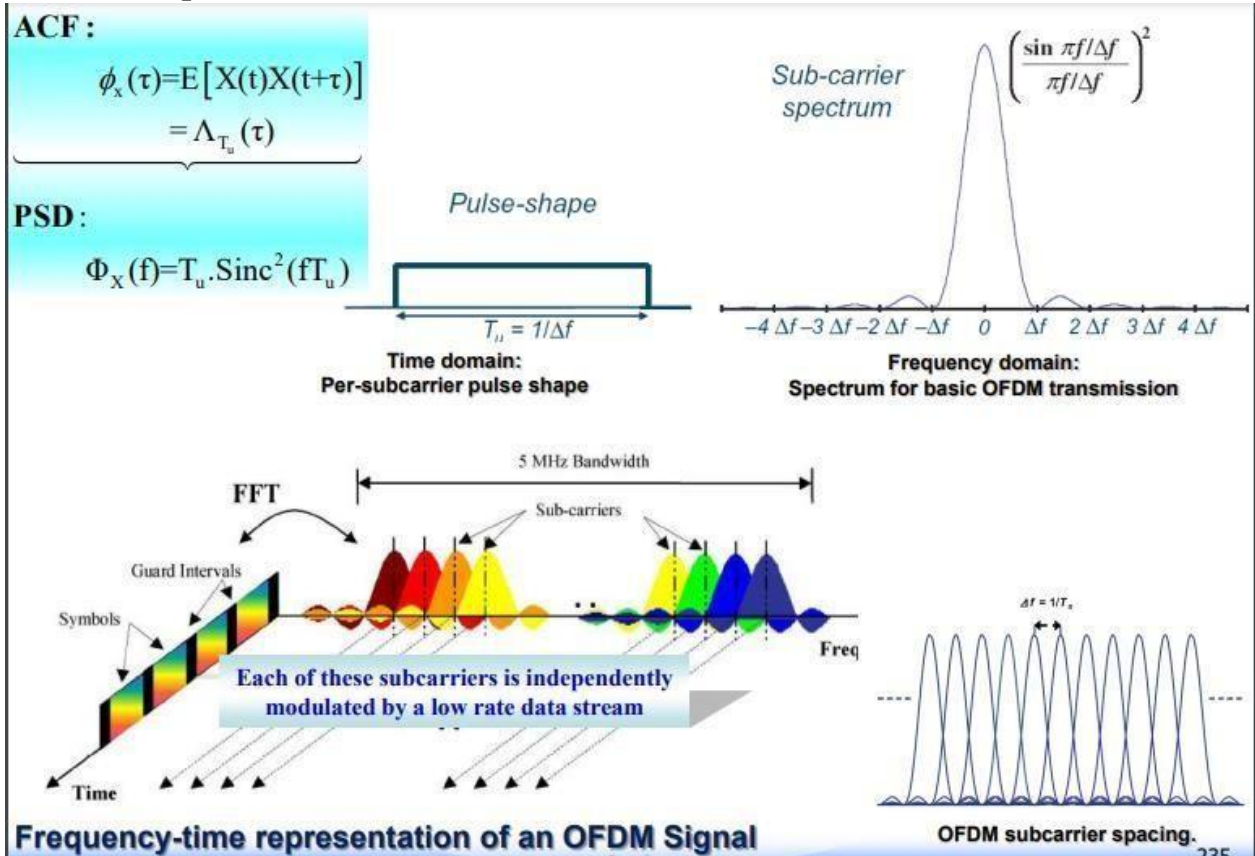
- Làm sáng quá trình xây dựng mô hình và nguyên lý hoạt động của hệ thống truyền dẫn OFDM trên cơ sở xử lý IFFT/FFT và chèn/khử CP.
- Trục quan hóa nguyên lý hoạt động trên cơ sở mô tả và mô phỏng các tín hiệu điển hình trên Matlab.

b, Nội dung:

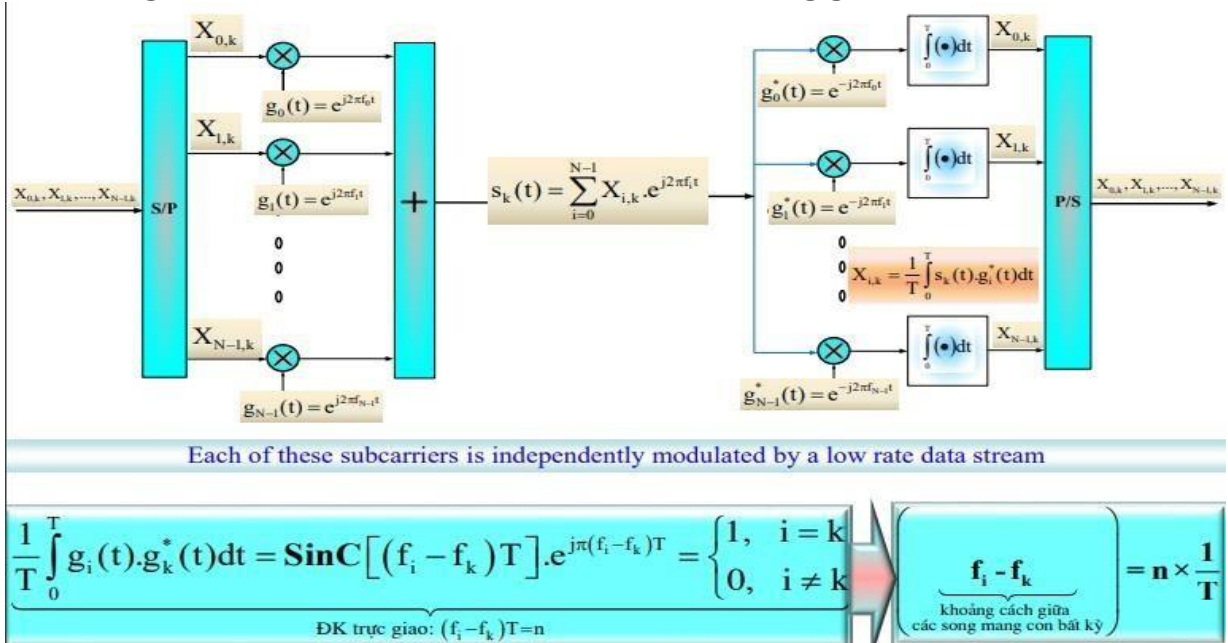
- Khái niệm cơ bản:
 - Tín hiệu và phổ tần của tín hiệu băng tần cơ sở.
 - Tín hiệu và phổ tần của tín hiệu thông dải/điều chế và dịch phổ tần tín hiệu.
 - Truyền dẫn đơn sóng mang/đa sóng mang, MCM/FDM.
 - FDM và OFDM.
- Xây dựng và trình bày nguyên lý hoạt động điều chế/giải điều chế OFDM trên cơ sở không gian tín hiệu.
 - Mô hình hóa quá trình truyền thông tín hiệu trên cơ sở không gian tín hiệu.
 - Mô hình hóa quá trình điều chế/giải điều chế tín hiệu OFDM trên cơ sở không gian tín hiệu: Nguyên lý hoạt động quá trình điều chế/giải điều chế.
 - Tín hiệu và hệ thống trong miền thời gian.
 - Tín hiệu và hệ thống trong miền tần số
 - Thực hiện điều chế/giải điều chế tín hiệu OFDM bằng thuật toán IFFT/FFT.
 - Matlab hóa và mô phỏng hệ thống OFDM trên cơ sở thuật toán IFFT/FFT.
- Các tham số đặc trưng của tín hiệu OFDM
 - Tham số tín hiệu OFDM trong miền thời gian.
 - Tham số tín hiệu OFDM trong miền tần số.
 - Lựa chọn các tham số OFDM trên cơ sở các tham số của kênh vô tuyến.
- Truyền dẫn tín hiệu OFDM
 - Truyền dẫn tín hiệu OFDM trong băng tần cơ sở.
 - Matlab hóa để tính toán biểu diễn tín hiệu OFDM trong băng tần cơ sở.
 - Truyền dẫn tín hiệu OFDM trong băng tần vô tuyến;
 - Matlab hóa để tính toán biểu diễn tín hiệu OFDM trong băng tần vô tuyến.
- Trục quan hóa nguyên lý hoạt động trên cơ sở mô tả và mô phỏng các tín hiệu đặc trưng của sơ đồ (mô hình) trên Matlab.

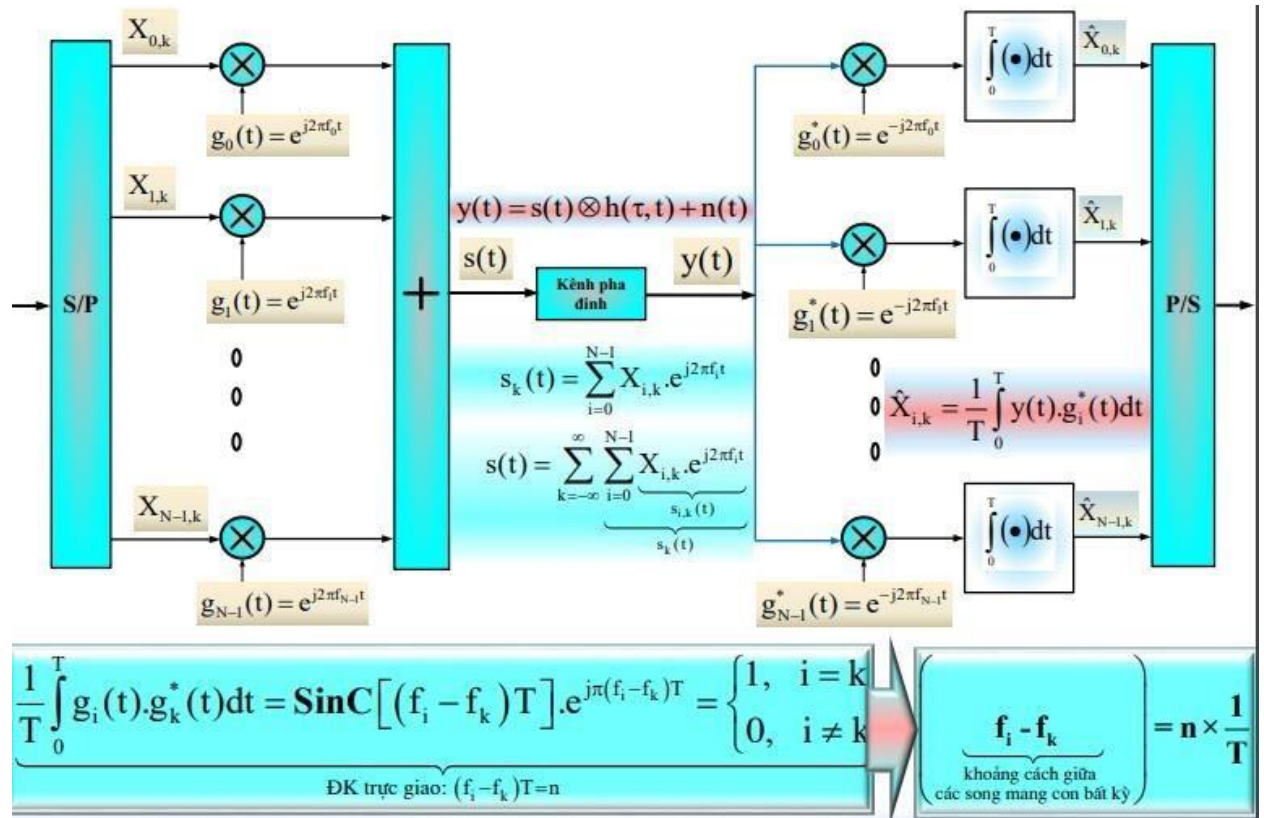
2, Cơ sở lý thuyết

- Tín hiệu và phổ tần tín hiệu OFDM



- Điều chế/giải điều chế tín hiệu OFDM trên cơ sở không gian tín hiệu





- Tính chất trực giao giữa các sóng mang con
Nếu các sóng mang con là:

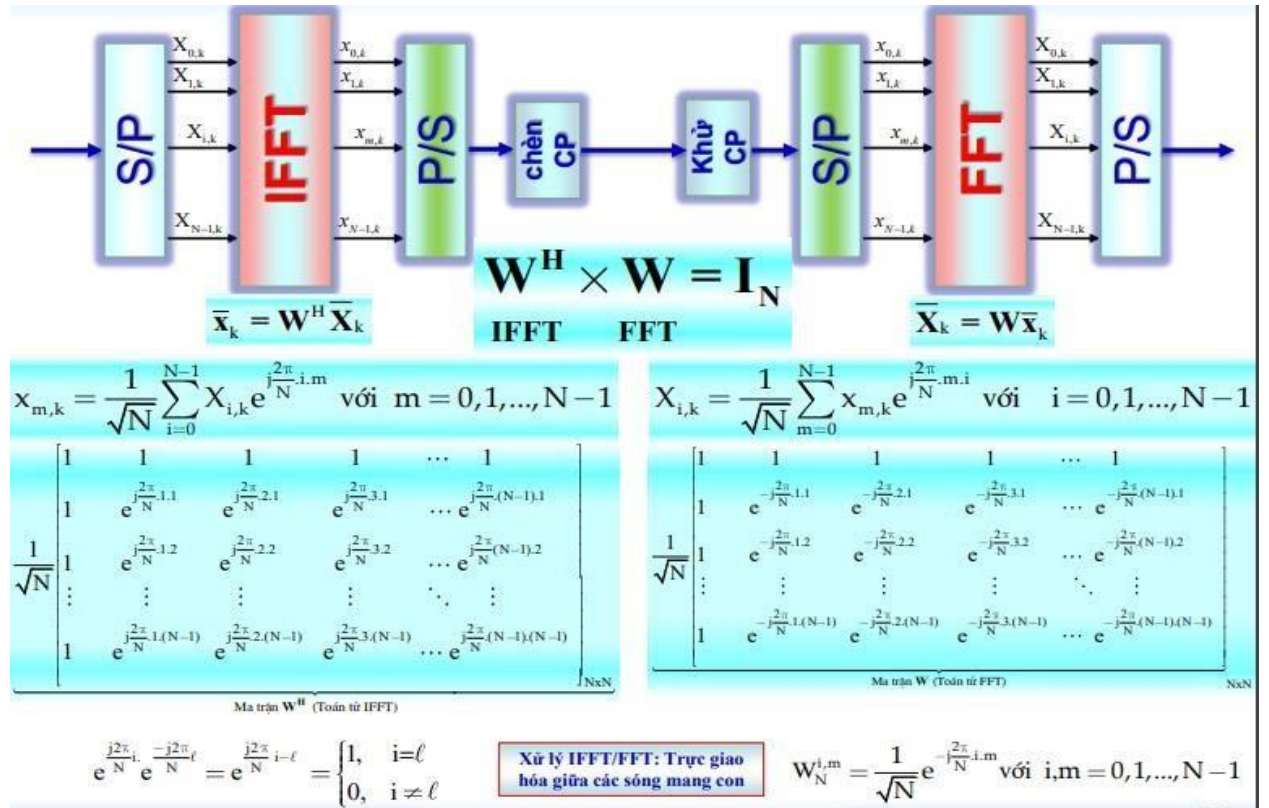
$$g_i(t) = \begin{cases} e^{j2\pi f_i t}, & t \in [0, T] \\ 0, & t \notin [0, T] \end{cases}$$

$$g_\ell^*(t) = \begin{cases} e^{-j2\pi f_\ell t}, & t \in [0, T] \\ 0, & t \notin [0, T] \end{cases}$$

Chứng minh:

$$\frac{1}{T} \int_{t_0}^{t_0+T} g_i(t) \cdot g_\ell^*(t) dt = \begin{cases} 1, & \text{if } f_i = f_\ell \\ 0, & \text{if } f_i = f_\ell + n \cdot \frac{1}{T} \end{cases}$$

- Mô hình hóa hệ thống truyền dẫn OFDM trên cơ sở thực hiện IFFT/FFT và chèn/khử CP



$$\bar{\mathbf{x}}_k = \mathbf{W}^H \bar{\mathbf{X}}_k = \left[X_{0,k}, X_{1,k}, \dots, X_{N-1,k} \right]_{N \times 1}^T$$

$$\mathbf{W}^H = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & e^{j\frac{2\pi}{N} 1.1} & e^{j\frac{2\pi}{N} 2.1} & e^{j\frac{2\pi}{N} 3.1} & \dots & e^{j\frac{2\pi}{N} (N-1).1} \\ 1 & e^{j\frac{2\pi}{N} 1.2} & e^{j\frac{2\pi}{N} 2.2} & e^{j\frac{2\pi}{N} 3.2} & \dots & e^{j\frac{2\pi}{N} (N-1).2} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & e^{j\frac{2\pi}{N} 1.(N-1)} & e^{j\frac{2\pi}{N} 2.(N-1)} & e^{j\frac{2\pi}{N} 3.(N-1)} & \dots & e^{j\frac{2\pi}{N} (N-1).(N-1)} \end{bmatrix}_{N \times N}$$

$\bar{\mathbf{X}}_k = [X_{0,k}, X_{1,k}, \dots, X_{i,k}, \dots, X_{N-1,k}]^T$: vector thể hiện N mẫu trong miền tần số $\{X_{i,k}\}, i=0, \dots, N-1$

$\bar{\mathbf{x}}_k = [X_{0,k}, X_{1,k}, \dots, X_{m,k}, \dots, X_{N-1,k}]^T$: vector thể hiện N mẫu trong miền thời gian $\{x_{m,k}\}, m=0, \dots, N-1$

$(.)^T$: phép chuyển vị

Xử lý IDFT (IFFT)

Hệ tuyến tính

$$\underbrace{\bar{\mathbf{X}}_k}_{\substack{\text{Đầu ra IFFT} \\ \text{(tập các mẫu miền thời gian)}}} = \underbrace{\mathbf{W}^H}_{\substack{\text{Toán tử IFFT} \\ \text{(Ma trận thực hiện IFFT)}}} \times \underbrace{\bar{\mathbf{X}}_k}_{\substack{\text{Đầu vào IFFT} \\ \text{(tập các mẫu miền tần số)}}} = [x_{0,k}, x_{1,k}, \dots, x_{N-1,k}]^T$$

$$= \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & e^{j\frac{2\pi}{N} \cdot 1 \cdot 1} & e^{j\frac{2\pi}{N} \cdot 2 \cdot 1} & e^{j\frac{2\pi}{N} \cdot 3 \cdot 1} & \dots & e^{j\frac{2\pi}{N} \cdot (N-1) \cdot 1} \\ 1 & e^{j\frac{2\pi}{N} \cdot 1 \cdot 2} & e^{j\frac{2\pi}{N} \cdot 2 \cdot 2} & e^{j\frac{2\pi}{N} \cdot 3 \cdot 2} & \dots & e^{j\frac{2\pi}{N} \cdot (N-1) \cdot 2} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & e^{j\frac{2\pi}{N} \cdot 1 \cdot (N-1)} & e^{j\frac{2\pi}{N} \cdot 2 \cdot (N-1)} & e^{j\frac{2\pi}{N} \cdot 3 \cdot (N-1)} & \dots & e^{j\frac{2\pi}{N} \cdot (N-1) \cdot (N-1)} \end{bmatrix}_{N \times N} \times \begin{bmatrix} X_{0,k} \\ X_{1,k} \\ \vdots \\ X_{i,k} \\ \vdots \\ X_{N-2,k} \\ X_{N-1,k} \end{bmatrix}_{N \times 1} = \begin{bmatrix} x_{0,k} \\ x_{1,k} \\ \vdots \\ x_{m,k} \\ \vdots \\ x_{N-2,k} \\ x_{N-1,k} \end{bmatrix}_{N \times 1}$$

Ma trận thực hiện IFFT

Tập các mẫu trong miền tần số đầu vào IFFT

Tập các mẫu trong miền thời gian đầu ra IFFT

$$x_{m,k} = \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} X_{i,k} e^{j\frac{2\pi}{N} \cdot i \cdot m} \quad \text{với } m = 0, 1, \dots, N-1$$

Xử lý DFT (FFT)

$$\bar{\mathbf{X}}_k = \mathbf{W} \bar{\mathbf{x}}_k$$

$$W_N^{i,m} = e^{-j\frac{2\pi}{N} \cdot i \cdot m}$$

$$\mathbf{W} = \begin{bmatrix} W_N^{00} & \dots & W_N^{0(N-1)} \\ \vdots & \ddots & \vdots \\ W_N^{(N-1)0} & \dots & W_N^{(N-1)(N-1)} \end{bmatrix} = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & e^{-j\frac{2\pi}{N} \cdot 1 \cdot 1} & e^{-j\frac{2\pi}{N} \cdot 2 \cdot 1} & e^{-j\frac{2\pi}{N} \cdot 3 \cdot 1} & \dots & e^{-j\frac{2\pi}{N} \cdot (N-1) \cdot 1} \\ 1 & e^{-j\frac{2\pi}{N} \cdot 1 \cdot 2} & e^{-j\frac{2\pi}{N} \cdot 2 \cdot 2} & e^{-j\frac{2\pi}{N} \cdot 3 \cdot 2} & \dots & e^{-j\frac{2\pi}{N} \cdot (N-1) \cdot 2} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & e^{-j\frac{2\pi}{N} \cdot 1 \cdot (N-1)} & e^{-j\frac{2\pi}{N} \cdot 2 \cdot (N-1)} & e^{-j\frac{2\pi}{N} \cdot 3 \cdot (N-1)} & \dots & e^{-j\frac{2\pi}{N} \cdot (N-1) \cdot (N-1)} \end{bmatrix}$$

$$W_N^{i,m} = \frac{1}{\sqrt{N}} e^{-j\frac{2\pi}{N} \cdot i \cdot m} \quad \text{với } i, m = 0, 1, \dots, N-1$$

$$\mathbf{W}\mathbf{W}^H = \mathbf{I}_N \Leftrightarrow \mathbf{W} \text{ và } \mathbf{W}^H \text{ thỏa mãn ĐK đơn nhất (Unitary)}$$

$$\text{FFT: } X_{i,k} = \sum_{m=0}^{N-1} x_{m,k} W_N^{i,m} = \sum_{m=0}^{N-1} x_{m,k} e^{-j\frac{2\pi}{N} \cdot m \cdot i}$$

Xử lý IDFT/DFT (IFFT/FFT):

$$\begin{aligned}
 \mathbf{W}^H \times \mathbf{W} &= \mathbf{I}_N \\
 \begin{matrix} \text{IFFT} & \text{FFT} \end{matrix} \\
 &= \frac{1}{N} \begin{bmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & e^{j\frac{2\pi}{N} \cdot 1 \cdot 1} & e^{j\frac{2\pi}{N} \cdot 2 \cdot 1} & e^{j\frac{2\pi}{N} \cdot 3 \cdot 1} & \dots & e^{j\frac{2\pi}{N} \cdot (N-1) \cdot 1} \\ 1 & e^{j\frac{2\pi}{N} \cdot 1 \cdot 2} & e^{j\frac{2\pi}{N} \cdot 2 \cdot 2} & e^{j\frac{2\pi}{N} \cdot 3 \cdot 2} & \dots & e^{j\frac{2\pi}{N} \cdot (N-1) \cdot 2} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & e^{j\frac{2\pi}{N} \cdot 1 \cdot (N-1)} & e^{j\frac{2\pi}{N} \cdot 2 \cdot (N-1)} & e^{j\frac{2\pi}{N} \cdot 3 \cdot (N-1)} & \dots & e^{j\frac{2\pi}{N} \cdot (N-1) \cdot (N-1)} \end{bmatrix}_{N \times N} \\
 &\quad \text{Ma trận } \mathbf{W}^H \text{ (Toán tử IFFT)} \\
 &\quad \times \begin{bmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & e^{-j\frac{2\pi}{N} \cdot 1 \cdot 1} & e^{-j\frac{2\pi}{N} \cdot 2 \cdot 1} & e^{-j\frac{2\pi}{N} \cdot 3 \cdot 1} & \dots & e^{-j\frac{2\pi}{N} \cdot (N-1) \cdot 1} \\ 1 & e^{-j\frac{2\pi}{N} \cdot 1 \cdot 2} & e^{-j\frac{2\pi}{N} \cdot 2 \cdot 2} & e^{-j\frac{2\pi}{N} \cdot 3 \cdot 2} & \dots & e^{-j\frac{2\pi}{N} \cdot (N-1) \cdot 2} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & e^{-j\frac{2\pi}{N} \cdot 1 \cdot (N-1)} & e^{-j\frac{2\pi}{N} \cdot 2 \cdot (N-1)} & e^{-j\frac{2\pi}{N} \cdot 3 \cdot (N-1)} & \dots & e^{-j\frac{2\pi}{N} \cdot (N-1) \cdot (N-1)} \end{bmatrix}_{N \times N} \\
 &\quad \text{Ma trận } \mathbf{W} \text{ (Toán tử FFT)} \\
 &= \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & 0 & \dots & 0 & 1 \end{bmatrix}_{N \times N} = \mathbf{I}_N;
 \end{aligned}$$

$$e^{j\frac{2\pi}{N} \cdot i} \cdot e^{-j\frac{2\pi}{N} \cdot \ell} = e^{j\frac{2\pi}{N} \cdot i - \ell} = \begin{cases} 1, & i = \ell \\ 0, & i \neq \ell \end{cases}$$

Xử lý IFFT/FFT: Trục giao hóa các sóng mang con

$$\mathbf{W}_N^{i,m} = \frac{1}{\sqrt{N}} e^{-j\frac{2\pi}{N} \cdot i \cdot m} \text{ với } i, m = 0, 1, \dots, N-1$$

- Chèn CP/khử CP

Chèn CP/khử CP

Chèn CP:

- + Mục đích: Đối phó ISI do pha đình đa đường
 - + Độ dài ký hiệu OFDM: $T = T_{\text{FFT}} + T_{\text{CP}}$
 - + T_{CP} thường được chọn bằng trễ trỗi cực đại của đa đường
- => Tổng số mẫu đầu ra bộ chèn CP là $(N+V)$ mẫu.

$$\begin{aligned}
 \bar{\mathbf{s}}_k &= \mathbf{C}_p \bar{\mathbf{x}}_k \longrightarrow \bar{\mathbf{s}}_k = \mathbf{C}_p \bar{\mathbf{x}}_k = [s_{-V}, s_{-V+1}, \dots, s_{-1}, s_0, s_1, \dots, s_{N-1}]^T \\
 \bar{\mathbf{x}}_k &= [x_{0,k}, x_{1,k}, \dots, x_{N-1,k}]^T \\
 \mathbf{C}_p &= \begin{bmatrix} \mathbf{0}_{V \times (N-V)} & \mathbf{I}_V \\ & \mathbf{I}_N \end{bmatrix} \\
 &= \begin{bmatrix} \overbrace{x_{N-V,k}, x_{N-V+1,k}, \dots, x_{N-1,k}}^{(N+V) \text{ mẫu dữ liệu}} \quad \underbrace{x_{0,k}, x_{1,k}, \dots, x_{N-1,k}}_{\text{Số liệu gốc (N mẫu)}} \end{bmatrix}^T \\
 &= \begin{bmatrix} \mathbf{0}_{V \times (N-V)} & \mathbf{I}_V \\ & \mathbf{I}_N \end{bmatrix}_{(N+V) \times N} \times \begin{bmatrix} x_{0,k} \\ \vdots \\ x_{N-1,k} \end{bmatrix}_{N \times 1}
 \end{aligned}$$

Tổng số mẫu đầu ra CP là $(N+V)$ mẫu trong ký hiệu OFDM thứ k

3, Code matlab

a, Sim_MA_06_PSD_OFDM

```
%=====
=====
%===== Sim_MA_06_PSD_OFDM
=====
%=====
=====

clc;
clear;
close all;

%=====
=====
deta_f          = 20;                                %
BW_channel/num_subcarrier=Subcarrier space;          % Corhence

Bandwidth of channel;15KHz
BW_channel      = 200;                                % bandwidth of
channel = 20MHz
% num_subcarrier = ceil(BW_channel/deta_f);          % Number of
subcarrier or subchannel round
num_subcarrier  = round(BW_channel/deta_f);          % Number of
subcarrier or subchannel
T_ofdm         = 1/deta_f;                            % OFDM time
R_ofdm         = 1/T_ofdm;
Tb             = T_ofdm/num_subcarrier;
Rb             = 1/Tb;                                % ceil
function
A              = 10;
A1            = A^2*Tb;
AA            = A^2*T_ofdm;
f_i           = deta_f:deta_f:BW_channel+deta_f;
f             = -Rb:BW_channel+4*deta_f;
% f_BB        = -Rb:4*Rb;
fc            = 3*max(f);
f2            = -f:1:(fc+BW_channel+4*deta_f);

% PSD of input of OFFDM Modulation Block
PSD_ofdm_in = A1*(sinc((f*Tb)).^2);

PSD_RF_SC   = A1*(sinc(((f2-fc)*Tb)).^2);

% PSD of output of OFFDM Modulation Block
PSD_OFDM    = zeros(num_subcarrier,max(size(f)));
PSD_OFDM_RF = zeros(num_subcarrier,max(size(f2)));
for k = 1:num_subcarrier
    PSD_OFDM(k,:) = AA*(sinc((f-f_i(k))*T_ofdm)).^2;
    % PSD_OFDM(k,:) = rand(1)*AA*(sinc((f-f_i(k))*T_ofdm)).^2;
```

```

        PSD_OFDM_RF(k,:)      = AA*(sinc((f2-f_i(k)-fc)*T_ofdm)).^2;
    % PSD_OFDM_RF(k,:)      = rand(1)*AA*(sinc((f2-f_i(k)-
fc)*T_ofdm)).^2;
end

figure(1)

%-----
subplot(2,2,1);
plot(f,PSD_ofdm_in,'r','LineWidth',3);
xlabel('Tan so
[H_z'],'FontName','.VnTime','color','b','FontSize',12);
ylabel('PSD_I_n_p_u_t_o_f
_O_F_D_M','FontName','.VnTime','color','b','FontSize',14);
title(['Mat do pho cong suat PSD cua tin hieu dau vao khoi OFDM
voi toc do la R_b =',num2str(Rb),'b/s'],...
'FontName','.VnTime','color','b','FontSize',9);
grid on;

%-----
subplot(2,2,2);
plot(f2,PSD_RF_SC,'m','LineWidth',3);
xlabel('Tan so
[H_z'],'FontName','.VnTime','color','b','FontSize',12);

ylabel('PSD_S_C_R_F','FontName','.VnTime','color','b','FontSize',14);
title(['Mat do pho cong suat PSD cua tin hieu SC_R_F voi toc do la
R_b =',num2str(Rb),'b/s'],...
';F_R_F=',num2str(fc),'H_Z'],...
'FontName','.VnTime','color','b','FontSize',9);
grid on;

%-----
subplot(2,2,3);
for k = 1:num_subcarrier
    plot(f,PSD_OFDM(k,:),'b','LineWidth',2);
    hold on
end
xlabel('Tan so
[H_z'],'FontName','.VnTime','color','b','FontSize',12);

ylabel('PSD_O_F_D_M','FontName','.VnTime','color','b','FontSize',14);
title(['PSD cua tin hieu OFDM: BW_C_h_a_n_n_e_l_
=',num2str(BW_channel),...
' H_Z ; Num_S_u_b_c_a_r_r_i_e_r =',num2str(num_subcarrier),...
'; Subcarrier_S_p_a_c_e =',num2str(deta_f),'H_Z'],...
'FontName','.VnTime','color','b','FontSize',9);
grid on;

%-----
subplot(2,2,4);

```

```

for k = 1:num_subcarrier
    plot(f2,PSD_OFDM_RF(k,:), 'b', 'LineWidth', 2);
    hold on
end
    xlabel('Tan so
[H_z]', 'FontName', '.VnTime', 'color', 'b', 'FontSize', 12);

ylabel('PSD_O_F_D_M_R_F', 'FontName', '.VnTime', 'color', 'b', 'FontSize', 14);

    title(['PSD cua tin hieu OFDM_R_F: BW_C_h_a_n_n_e_l_
=' , num2str(BW_channel), ...
        ' H_Z ; Num_S_u_b_c_a_r_r_i_e_r =' , num2str(num_subcarrier), ...
        '; Subcarrier_S_p_a_c_e
=' , num2str(deta_f), 'H_Z', '; f_R_F=' , num2str(fc), 'H_Z'], ...
        'FontName', '.VnTime', 'color', 'b', 'FontSize', 9);
    grid on;

    PSD_OFDM_sum_RF = sum(PSD_OFDM_RF, 'double');

%=====
figure(2)

%-----
subplot(2,1,1);

for k = 1:num_subcarrier
    plot(f2,PSD_OFDM_RF(k,:), 'b', 'LineWidth', 2);
    hold on
end
    h11 = plot(f2, PSD_RF_SC, 'r', 'LineWidth', 3);
    hold on
    h12 = plot(f2, PSD_OFDM_sum_RF, '+r', 'LineWidth', 4);

    xlabel('Tan so
[H_z]', 'FontName', '.VnTime', 'color', 'b', 'FontSize', 12);
    ylabel('PSD_O_F_D_M_R_F &
SC_R_F', 'FontName', '.VnTime', 'color', 'b', 'FontSize', 14);
    title(['So sanh PSD cua tin hieu OFDM_R_F & SC_R_F:
BW_C_h_a_n_n_e_l_ =' , num2str(BW_channel), ...
        ' H_Z ; Num_S_u_b_c_a_r_r_i_e_r =' , num2str(num_subcarrier), ...
        '; Subcarrier_S_p_a_c_e
=' , num2str(deta_f), 'H_Z', '; f_R_F=' , num2str(fc), 'H_Z'], ...
        'FontName', '.VnTime', 'color', 'b', 'FontSize', 12);
    grid on;
    K = legend('PSD cua OFDM_R_F', 'PSD cua SC_R_F', 'PSD cua
OFDM_S_U_M_-_R_F');
    set(K, 'fontname', '.Vntime', 'fontsize', 13);

%-----
subplot(2,1,2)
    plot(f2, PSD_OFDM_sum_RF, 'b', 'LineWidth', 2);

```

```

hold on
plot(f2,PSD_RF_SC,'r','LineWidth',3);

xlabel('Tan so
[H_z'],'FontName','.VnTime','color','b','FontSize',12);
ylabel('PSD_O_F_D_M_R_F &
SC_R_F','FontName','.VnTime','color','b','FontSize',14);
title(['PSD cua tin hieu OFDM_R_F & SC_R_F: BW_C_h_a_n_n_e_l_
=',num2str(BW_channel),...
' H_Z ; Num_S_u_b_c_a_r_r_i_e_r =',num2str(num_subcarrier),...
'; Subcarrier_S_p_a_c_e
=',num2str(deta_f),'H_Z',';F_R_F=',num2str(fc),'H_Z'],...
'FontName','.VnTime','color','r','FontSize',12);
grid on;
L = legend('PSD cua OFDM_S_U_M_-_R_F','PSD cua SC_R_F');
set(L, 'fontname','.Vntime','fontsize',13);
%=====
=====

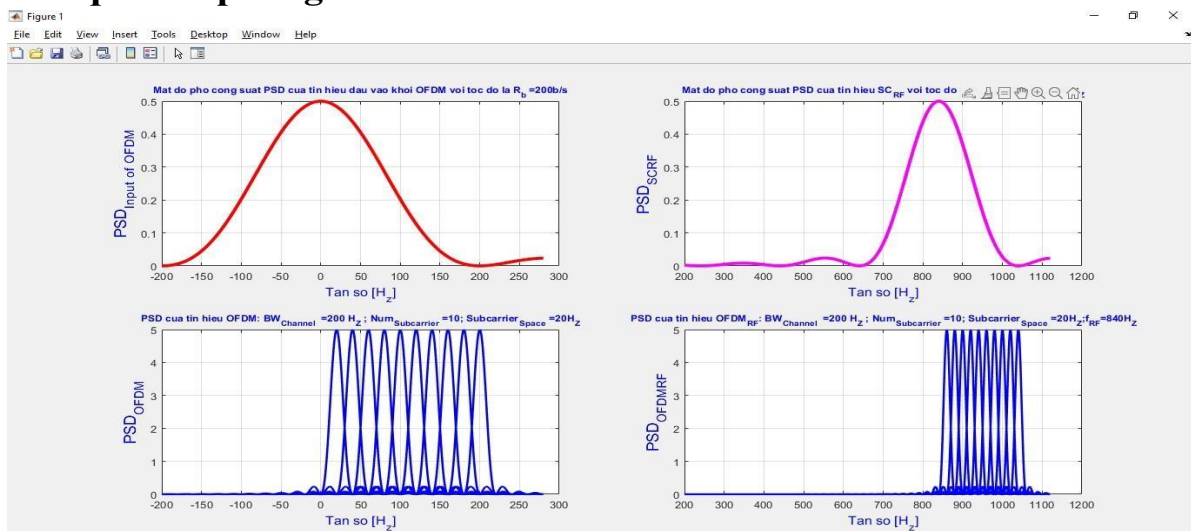
```

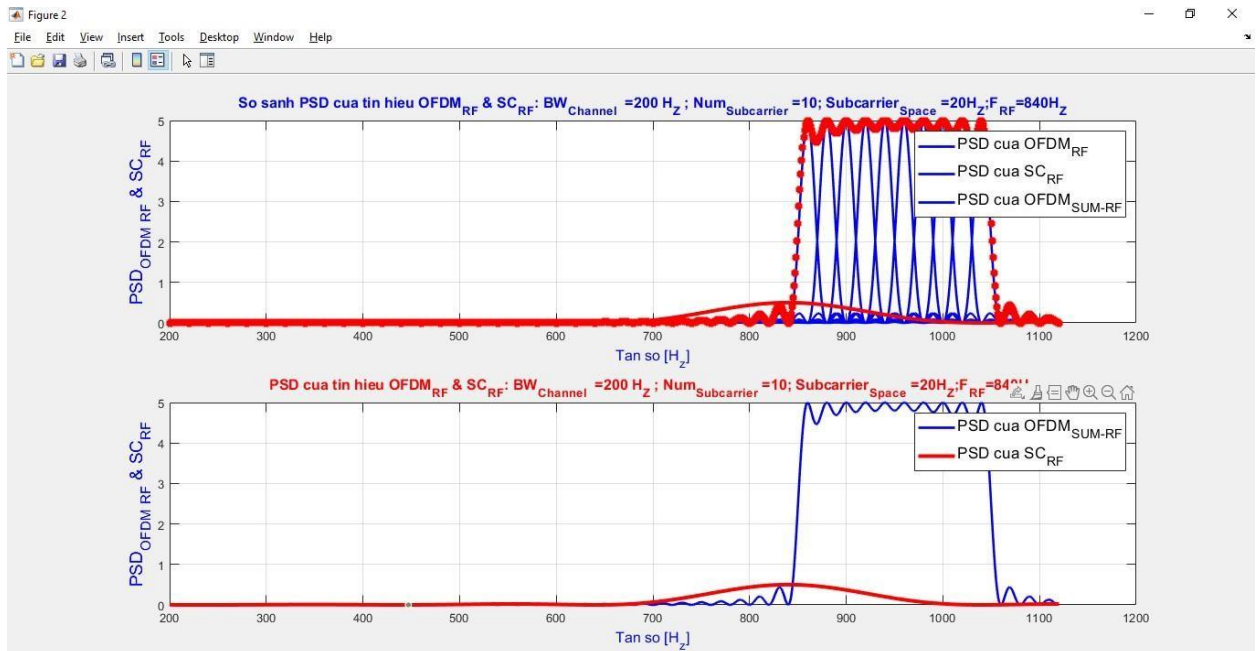
❖ Giải thích code

Tham số	Giải thích
deta_f = 20;	Khoảng cách giữa các tín hiệu subcarrier trong hệ thống OFDM.
BW_channel = 200;	Băng thông của kênh truyền
num_subcarrier = round(BW_channel/deta_f);	Số lượng subcarrier hoặc subchannel được tính toán dựa trên băng thông kênh và khoảng cách giữa các subcarrier.
T_ofdm = 1/deta_f;	Thời gian của một ký hiệu OFDM.
R_ofdm = 1/T_ofdm;	Tốc độ truyền dữ liệu của OFDM.
Tb = T_ofdm/num_subcarrier;	Thời gian của một bit dữ liệu trên mỗi subcarrier.
Rb = 1/Tb;	Tốc độ truyền dữ liệu trên mỗi subcarrier.
A = 10; A1 = A^2*Tb;	A là biên của tín hiệu OFDM và A1 là công thức tính toán năng lượng của tín hiệu OFDM, có liên quan đến biên và thời gian bit
AA = A^2*T_ofdm;	Công thức tính toán năng lượng của tín hiệu OFDM

<code>f_i = deta_f:deta_f:BW_channel+deta_f;</code>	Một mảng chứa các tần số subcarrier trong hệ thống OFDM.
<code>f = - Rb:BW_channel+4*deta_f;</code>	Một mảng tần số.
<code>fc = 3*max(f);</code>	Tần số tạm thời
<code>f2 = - f:1:(fc+BW_channel+4*deta_f);</code>	Mảng tần số f2
Lệnh	Giải thích
<code>PSD_ofdm_in = A1*(sinc((f*Tb)).^2);</code>	Công suất phổ của tín hiệu đầu vào OFDM.
<code>PSD_RF_SC = A1*(sinc(((f2- fc)*Tb)).^2);</code>	Công suất phổ của tín hiệu sau khi được chuyển đổi thành tần số cao hơn
<code>PSD_OFDM = zeros(num_subcarrier,max(size(f))); PSD_OFDM_RF = zeros(num_subcarrier,max(size(f2)));</code>	Các ma trận chứa công suất phổ của tín hiệu OFDM và tín hiệu sau RF cho từng subcarrier.
<code>for k = 1:num_subcarrier PSD_OFDM(k,:) = AA*(sinc((f-f_i(k))*T_ofdm)).^2; PSD_OFDM_RF(k,:) = AA*(sinc((f2-f_i(k)-fc)*T_ofdm)).^2; end</code>	Tính công suất phổ của tín hiệu OFDM và tín hiệu sau RF cho từng subcarrier.

❖ Kết quả mô phỏng





Workspace	
Name	Value
A	10
A1	0.5000
AA	5
BW_channel	200
deta_f	20
f	1x481 1x1 double
f2	1x921 double
f_i	1x11 double
fc	840
h11	1x1 Line
h12	1x1 Line
k	10
K	1x1 Legend
L	1x1 Legend
num_subcarrier	10
PSD_OFDM	10x481 double
PSD_ofdm_in	1x481 double
PSD_OFDM_RF	10x921 double
PSD_OFDM_sum_RF	1x921 double
PSD_RF_SC	1x921 double
R_ofdm	20
Rb	200
T_ofdm	0.0500
Tb	0.0050

b, MA_06_IFFT_FFT_AWGN

```
%-----  
%-----MA_06_IFFT_FFT_AWGN-----  
%-----  
  
clc;  
clear all;  
close all;  
%-----  
FFTsize      = 1000;  
CPsize        = 25;  
snr_in_dB     = 10;  
noisePower    = 10^(-snr_in_dB/10);  
%-----  
% Generate for FFTsize bits: BPSK  
    data      = 0.5*(sign(rand(1,FFTsize))-0.5)+1);  
    data      = 2*data-1;  
%-----  
% IFFT & FFT Principles  
    % step 1: IFFT process  
        data_IFFT      = ifft(data);  
    % step 2: add CP  
        data_IFFT_CP   = [data_IFFT(FFTsize-CPsize+1:FFTsize)  
data_IFFT];  
    % step 3: AWGN channel  
        tmp            = randn(1,FFTsize+CPsize);  
        RV_Gaussian    = tmp*noisePower;  
        RxSymbols      = data_IFFT_CP + RV_Gaussian;  
    % step 4: remove CP  
        data_CPR       = RxSymbols(CPsize+1:FFTsize+CPsize);  
    % step 5: IFFT process  
        data_FFT       = fft(data_CPR);  
  
%%%%% decision and determine error  
% solution 1:  
    % Hard decision  
    data_des1      = zeros(1, length(data));  
    for i = 1:length(data_FFT)  
        if data_FFT(i) >= 0  
            data_des1(i) = 1;  
        else  
            data_des1(i) = -1;  
        end  
    end  
    % to determine error (comparesion)  
    error_vector1   = data~=data_des1;  
    % errCount & number of errors
```

```

num_error1          = sum(error_vector1);
BER1                = num_error1/FFTsize

% solution 2:
% Hard decision
data_des2           = sign(real(data_FFT));
% to determine error (comparesion)
error_vector2       = data~=data_des2;
% errCount & number of errors
num_error2          = sum(error_vector2);
BER2                = num_error2/FFTsize

% optimal solution optimal
BER_op = sum(sign(real(data_FFT))~=data)/FFTsize

```

❖ Giải thích code

Tham số	Giải thích
FFTsize = 1000;	Kích thước của phép biến đổi Fourier (FFT) sử dụng trong hệ thống OFDM.
CPsize = 25;	Kích thước của cyclic prefix (CP), một phần của tín hiệu được thêm vào phía trước của các ký hiệu OFDM để giảm hiện tượng nhiễu nhiễu giữa các ký hiệu.
snr_in_dB = 10;	Tín hiệu đến nhiễu (SNR) đầu vào trong đơn vị đo dB.
noisePower = 10^(-snr_in_dB/10);	Công suất của nhiễu Gaussian thêm vào tín hiệu, được tính bằng cách chuyển đổi snr_in_dB thành tỷ lệ nhiễu và lấy nghịch đảo logarithm cơ số 10.
Lệnh	Giải thích
data = 0.5*(sign(rand(1,FFTsize)-0.5)+1); data = 2*data-1;	Tạo dữ liệu ngẫu nhiên cho việc truyền, biểu diễn dưới dạng tín hiệu BPSK với giá trị +1 và -1.
data_IFFT = ifft(data);	Thực hiện phép biến đổi ngược (IFFT) trên dữ liệu để chuyển từ miền tần số sang miền thời gian.
data_IFFT_CP = [data_IFFT(FFTsize-CPsize+1:FFTsize) data_IFFT];	Thêm cyclic prefix (CP) vào tín hiệu sau khi IFFT.

<pre> tmp = randn(1,FFTsize+CPsize); RV_Gaussian = tmp*noisePower; RxSymbols = data_IFFT_CP + RV_Gaussian; </pre>	<p>Áp dụng kênh nhiễu Gaussian (AWGN) bằng cách thêm nhiễu Gaussian có công suất noisePower vào tín hiệu.</p>
<pre> data_CPR = RxSymbols(CPsize+1:FFTsize+CPsize); </pre>	<p>Loại bỏ cyclic prefix để chuẩn bị cho phép biến đổi Fourier (FFT).</p>
<pre> data_FFT = fft(data_CPR); </pre>	<p>Thực hiện phép biến đổi Fourier trên dữ liệu để chuyển từ miền thời gian sang miền tần số.</p>
<pre> data_des1 = zeros(1, length(data)); for i = 1:length(data_FFT) if data_FFT(i) >= 0 data_des1(i) = 1; else data_des1(i) = -1; end end error_vector1 = data~=data_des1; num_error1 = sum(error_vector1); BER1 = num_error1/FFTsize </pre>	<p>Sử dụng quyết định cứng (hard decision) bằng cách so sánh ký hiệu tín hiệu sau FFT với ngưỡng 0 để xác định các bit. Tính toán số lỗi bit và tỷ lệ lỗi (BER).</p>
<pre> data_des2 = sign(real(data_FFT)); error_vector2 = data~=data_des2; num_error2 = sum(error_vector2); BER2 = num_error2/FFTsize </pre>	<p>Sử dụng quyết định cứng bằng cách lấy phần thực của ký hiệu sau FFT làm kết quả. Tính toán số lỗi bit và tỷ lệ lỗi (BER).</p>
<pre> BER_op = sum(sign(real(data_FFT))~=data)/FFTsize </pre>	<p>Sử dụng một quyết định tối ưu bằng cách so sánh phần thực của ký hiệu sau FFT với dữ liệu gốc. Tính toán tỷ lệ lỗi (BER).</p>

❖ **Kết quả mô phỏng:**

Workspace	
Name ▲	Value
BER1	0.3120
BER2	0.3120
BER_op	0.3120
CPsize	25
data	1x1000 double
data_CPR	1x1000 complex double
data_des1	1x1000 double
data_des2	1x1000 double
data_FFT	1x1000 complex double
data_IFFT	1x1000 complex double
data_IFFT_CP	1x1025 complex double
<input checked="" type="checkbox"/> error_vector1	1x1000 logical
<input checked="" type="checkbox"/> error_vector2	1x1000 logical
FFTsize	1000
i	1000
noisePower	0.1000
num_error1	312
num_error2	312
RV_Gaussian	1x1025 double
RxSymbols	1x1025 complex double
snr_in_dB	10
tmp	1x1025 double

c, MA_06_OFDM_Principle

```

%
%-----
%-----MA_06_OFDM_Principle-----
%-----
%-----

clc;
clear;
N = 4; %input('Enter N =');
V = 2; %input('Enter V =');

X1 = 1:N;
[W_H] = MA_06_IFFT_matrix(N);
[W] = MA_06_FFT_matrix(N); % note W=inv(W_H) W*W_H = I
[CP_insert] = MA_06_CP_insert(N,V);
[CP_Remve] = MA_06_CP_Remove(N,V);

Mode = 1;

if Mode == 1

```

```

else
    X1 = 0.5*(sign(rand(1,N)-0.5)+1);
    % X1 = 2*X1-1;

end
% -----
X2 = X1';
X3 = W_H*X2; % IFFT
X4 = X3';
X5 = X4';
X6 = CP_insert*X5;
X7 = X6';
X8 = X7';
X9 = CP_Remve*X8;
X10 = X9';
X11 = X10';
X12 = W*X11; % FFT
X13 = X12'
% ===== Check for IFFT/FFT; CP_insert_remove
% X13_T = abs(X13)
    Test_IFFT_FFT_matrix = abs(W_H*W);
    Test_CP_inser_remove = CP_Remve*CP_insert;
    % Test_CP_inser_remove2 = CP_insert*CP_Remve
% ===== Check for System Modeling
    X1;
    X13;
    % Test_I_O = xor(X1,X13); % Note khong dung X1~=X13

%=====
% IFFT & FFT Princeples
    % step 1: IFFT process
        data_IFFT = sqrt(N)*ifft(X1,N);
    % step 2: add CP
        data_IFFT=data_IFFT';
        data_IFFT_CP = [data_IFFT(N-V+1:N) data_IFFT];
    % step 3: remove CP
        data_IFFT_CP= data_IFFT_CP';
        data_CPR = data_IFFT_CP(V+1:N+V);
    % step 4: IFFT process
        data_FFT = (1/sqrt(N))*fft(data_CPR,N)

```

❖ Giải thích code

Lệnh	Giải thích
N = 4;	Số lượng subcarrier (tín hiệu con) trong hệ thống OFDM.
V = 2;	Kích thước của cyclic prefix (CP), một phần của tín hiệu OFDM được

	thêm vào phía trước để giảm hiện tượng nhiễu nhiễu giữa các tín hiệu.
<pre>[W_H] = MA_06_IFFT_matrix(N); [W] = MA_06_FFT_matrix(N);</pre>	Tạo ra các ma trận biến đổi IFFT và FFT cho kích thước N.
<pre>[CP_insert] = MA_06_CP_insert(N,V); [CP_Remve] = MA_06_CP_Remove(N,V);</pre>	Tạo ma trận chèn và khử CP
<pre>data_IFFT = sqrt(N)*ifft(X1,N);</pre>	Thực hiện phép biến đổi IFFT bằng cách nhân vector X1 với ma trận biến đổi IFFT W_H.
<pre>data_IFFT_CP = [data_IFFT(N- V+1:N) data_IFFT];</pre>	Thêm cyclic prefix bằng cách lấy phần cuối cùng của data_IFFT (kích thước V) và đặt nó trước data_IFFT.
<pre>data_CPR = data_IFFT_CP(V+1:N+V);</pre>	Loại bỏ cyclic prefix bằng cách lấy các phần tử từ vị trí V+1 đến N+V của data_IFFT_CP.
<pre>data_FFT = (1/sqrt(N))*fft(data_CPR,N)</pre>	Thực hiện phép biến đổi FFT bằng cách nhân vector data_CPR với ma trận biến đổi FFT W.

❖ Kết quả

```
X13 =

    1.0000 - 0.0000i    2.0000 - 0.0000i    3.0000 - 0.0000i    4.0000 + 0.0000i

data_FFT =

    1    2    3    4
```

Workspace	
Name	Value
CP_insert	6x4 double
CP_Remve	4x6 double
data_CPR	[5.0000 + 0.0000i,-1.0...
data_FFT	[1,2,3,4]
data_IFFT	[5.0000 + 0.0000i,-1.0...
data_IFFT_CP	[-1.0000 + 0.0000i,-1....
Mode	1
N	4
Test_CP_inser_rem...	4x4 double
Test_IFFT_FFT_mat...	4x4 double
V	2
W	4x4 complex double
W_H	4x4 complex double
X1	[1,2,3,4]
X10	[5.0000 + 0.0000i,-1.0...
X11	[5.0000 + 0.0000i,-1.0...
X12	[1.0000 + 0.0000i;2.00...
X13	[1.0000 - 0.0000i;2.00...
X2	[1;2;3;4]
X3	[5.0000 + 0.0000i,-1.0...
X4	[5.0000 + 0.0000i,-1.0...
X5	[5.0000 + 0.0000i,-1.0...
X6	[-1.0000 + 0.0000i;-1....
X7	[-1.0000 - 0.0000i,-1.0...
X8	[-1.0000 + 0.0000i;-1....
X9	[5.0000 + 0.0000i,-1.0...

Sim_MA07: Mô hình hóa và mô phỏng hiệu năng BER cho hệ thống truyền dẫn BPSK-OFDM dùng mã kênh trong môi trường kênh AWGN

1, Mục đích và nội dung

a, Mục đích:

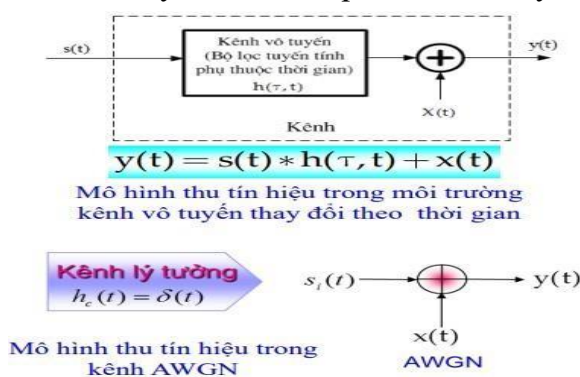
- Mô hình hóa và trực quan hóa nguyên lý hoạt động của hệ thống BPSK-OFDM dùng mã kênh trong môi trường kênh AWGN.
- Matlab hóa và mô phỏng hệ thống BPSK-OFDM dùng mã kênh trong môi trường kênh AWGN để: làm sáng tỏ nguyên lý hoạt động và phân tích đánh giá hiệu năng.

b, Nội dung:

- Xây dựng và trình bày nguyên lý hoạt động quá trình điều chế/giải điều chế OFDM trên cơ sở không gian tín hiệu.
- Xây dựng mô hình và nguyên lý hoạt động hệ thống BPSK-OFDM dùng mã kênh trong môi trường kênh AWGN.
- Tiến trình mô phỏng: Lưu đồ mô phỏng và thực hiện mô phỏng hệ thống BPSK-OFDM dùng mã kênh trong môi trường kênh AWGN.
- Matlab hóa và mô phỏng hiệu năng BER của hệ thống BPSK-OFDM dùng mã kênh trong môi trường kênh AWGN.
 - Thiết lập kịch bản mô phỏng: Định nghĩa tham số và thiết lập tập tham số đầu vào cho chương trình mô phỏng.
 - Matlab hóa mô hình mô phỏng hệ thống BPSK-OFDM dùng mã kênh trong môi trường kênh AWGN.
 - Thiết lập các bước mô phỏng và thực hiện mô phỏng theo kịch bản mô phỏng.
- Thực hiện mô phỏng trên Matlab để: sáng tỏ nguyên lý hoạt động và khảo sát đánh giá hiệu năng BER.

2, Cơ sở lý thuyết

- Mô hình truyền tín hiệu qua kênh vô tuyến



- Mô hình mô phỏng hệ thống truyền dẫn OFDM băng tần cơ sở trong môi trường kênh AWGN.



- Mô hình hệ thống truyền dẫn BPSK-OFDM sử dụng mã kênh trong băng tần gốc trong môi trường kênh AWGN



3, Code matlab

- **Sim_MA_07_BPSK_OFDM_AWGN_ChannelCode**

```
%=====
=====
%===== Sim_MA_07_BPSK_OFDM_AWGN_ChannelCode
=====
%=====
=====

clc;
clear all;
close all;

SNR          = [0:1:9];
FFTsize      = 512;
CPsize       = 20;
numRun       = 10^3;          % Note
NumBits      = FFTsize*numRun;
%=====
=====
mode_Sim      = 2;          % 1 for No channel code (OFDM without channel
coding);
                                % 2 for channel coding (OFDM with channel
coding)
Coddling_Type = 1;          % Code Generation Matrix

if mode_Sim ==2
    if Coddling_Type == 1
        k0      = 1;
        G        = [1 1 1;1 0 1];
    elseif Coddling_Type == 2
        k0      = 1;
        G        = [1 1 1 1 0 0 1;1 0 1 1 0 1 0];
    else
        k0      = 2;
        G        = [0 0 1 0 1 0 0 1;0 0 0 0 0 0 0 1;1 0 0 0 0 0 0 1];
    end
end
%=====
=====
for n = 1:length(SNR),
    errCount = 0;
    for k = 1:numRun
        % Generated BPSK data
        numSymbols      = FFTsize;
        data             = 0.5*(sign(rand(1,numSymbols))-0.5)+1);
        data2            = 2*data-1;
```

```

%-----
if mode_Sim ==2 % Convolution code Encoder
    inputSymbols = FWC_COV_Encoder(G,k0,data);
    inputSymbols = 2*inputSymbols-1;
else
    inputSymbols = 2*data-1;
end
%-----

% IFFT (OFDM Modulation)
TxSamples =
sqrt(length(inputSymbols))*ifft(inputSymbols,length(inputSymbols));
numSymbols = length(inputSymbols);
% Insert CP
Tx_ofdm = [TxSamples(numSymbols-CPsize+1:numSymbols)
TxSamples];
% AWGN channel
numSymbols_2 = length(inputSymbols);
tmp = randn(1,numSymbols_2+CPsize);
noisePower = 10^(-SNR(n)/10);
RxSymbols = Tx_ofdm + sqrt(noisePower)*tmp;
% Remove CP
EstSymbols_1 = RxSymbols(CPsize+1:numSymbols_2+CPsize);
% IFT (OFDM Demodulation)
Y = fft(EstSymbols_1,length(EstSymbols_1));
% Detection and decide
EstSymbols_1 = Y;
EstSymbols_1 = sign(real(EstSymbols_1));
for i = 1:length(EstSymbols_1)
    if EstSymbols_1(i)>0
        Decis(i)= 1;
    else
        Decis(i)= 0;
    end
end
%-----

if mode_Sim==2
    %---- Convolution code Decoder
    EstSymbols = FWC_COV_Dencoder(G,k0,Decis);
else
    EstSymbols = EstSymbols_1;
end
%-----

% Check for Error
if mode_Sim==2
    I = find((data-EstSymbols) == 0);
else
    I = find((data2-EstSymbols) == 0);
end

```



```

        % Countered Errors
        errCount      = errCount + (FFTsize-length(I));
    end
    SER(n,:)          = errCount / (FFTsize*numRun);
end

%=====
=====
if mode_Sim ==2
    save MA_07_BPSK_OFDM_CC_AWGN.mat;
    figure(1);
    G = semilogy(SNR,SER,'-vr');
    title(['Mo phong BER he thong BPSK OFDM trong kenh AWGN voi ma hoa
kenh; So bit mo phong = ',...
        num2str(NumBits),' bits
'], 'FontName', '.VnTime', 'color', 'b', 'FontSize', 16);
    LT=legend('OFDM - kenh AWGN co ma hoa kenh');
    set(LT, 'fontname', '.Vntime', 'fontsize', 16);
else
    save MA_07_BPSK_OFDM_NoCC_AWGN.mat;
    figure(1);
    G = semilogy(SNR,SER,'-ob');
    title(['Mo phong BER he thong BPSK OFDM trong kenh AWGN; So bit mo
phong = ',num2str(NumBits),' bits ',...
        ], 'FontName', '.VnTime', 'color', 'b', 'FontSize', 16);
    LT=legend('OFDM - kenh AWGN khong ma hoa kenh');
    set(LT, 'fontname', '.Vntime', 'fontsize', 16);
end
set(G, 'LineWidth', 1.5);
AX = gca;
set(AX, 'fontsize', 14);
X=xlabel('SNR (dB)');
set(X, 'fontname', '.Vntime', 'fontsize', 14, 'color', 'b');
Y=ylabel('BER');
set(Y, 'fontname', '.Vntime', 'fontsize', 14, 'color', 'b');
grid on;

```

```

%-----
• Presentation_Sim_MA_07
%=====
=====
%===== Presentation_Sim_MA_07
=====
%=====
=====

clc;
clear all;
close all;
%=====
=====
load MA_07_BPSK_OFDM_NoCC_AWGN.mat;

```

```

SER_noChannelCoding = SER;
SNR_1                = SNR;
clear SER;
%
load MA_07_BPSK_OFDM_CC_AWGN.mat;
SER_ChannelCoding = SER;
SNR_2              = SNR;
clear SER;

%
-----
figure(1)
G = semilogy(SNR_1, SER_noChannelCoding, '-ob');
set(G, 'LineWidth', 1.5);
hold on;

%
-----
G = semilogy(SNR_2, SER_ChannelCoding, '-.vr');
set(G, 'LineWidth', 2.5);

%
-----
AX = gca;
set(AX, 'fontsize', 14);
X = xlabel('SNR (dB)');
set(X, 'fontname', 'VnTime', 'fontsize', 14, 'color', 'b');
Y = ylabel('BER');
set(Y, 'fontname', 'VnTime', 'fontsize', 14, 'color', 'b');
title(['Mo phong BER he thong BPSK/OFDM trong kenh AWGN co
va khong ma hoa kenh; So bit mo phong = ', ...
num2str(NumBits), ' bits
'], 'FontName', 'VnTime', 'color', 'b', 'FontSize', 14);
L=legend('OFDM - kenh AWGN khong ma hoa kenh', 'OFDM - kenh
AWGN co ma hoa kenh');
set(L, 'fontname', 'VnTime', 'fontsize', 13);
grid on;

%=====
=====

```

❖ Giải thích code

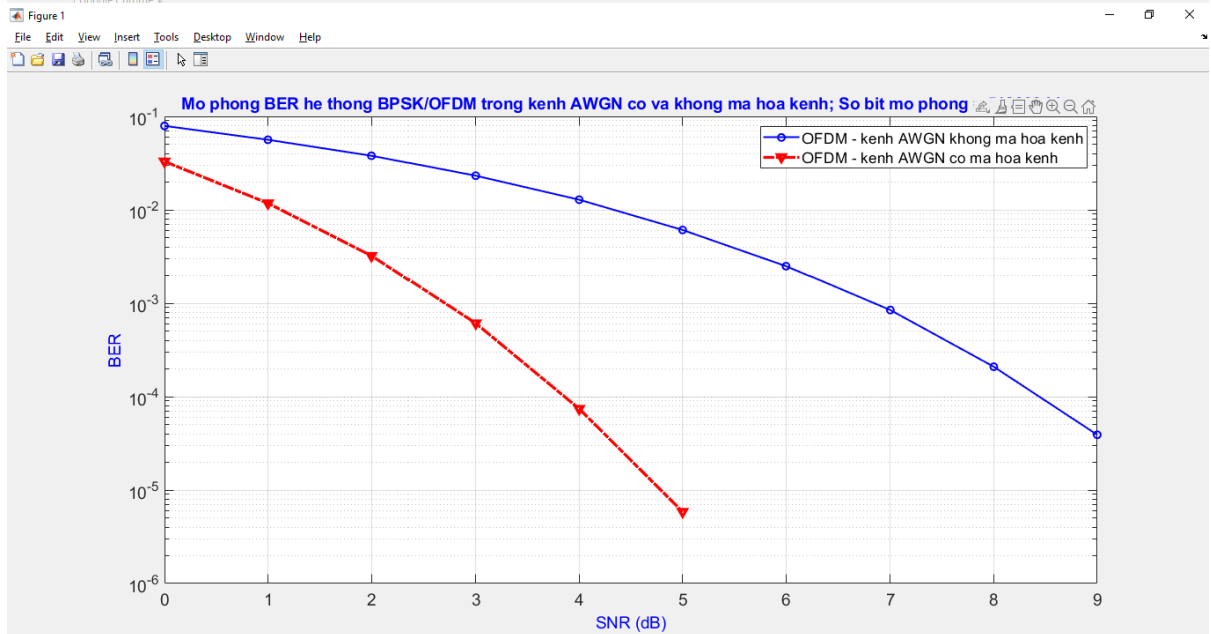
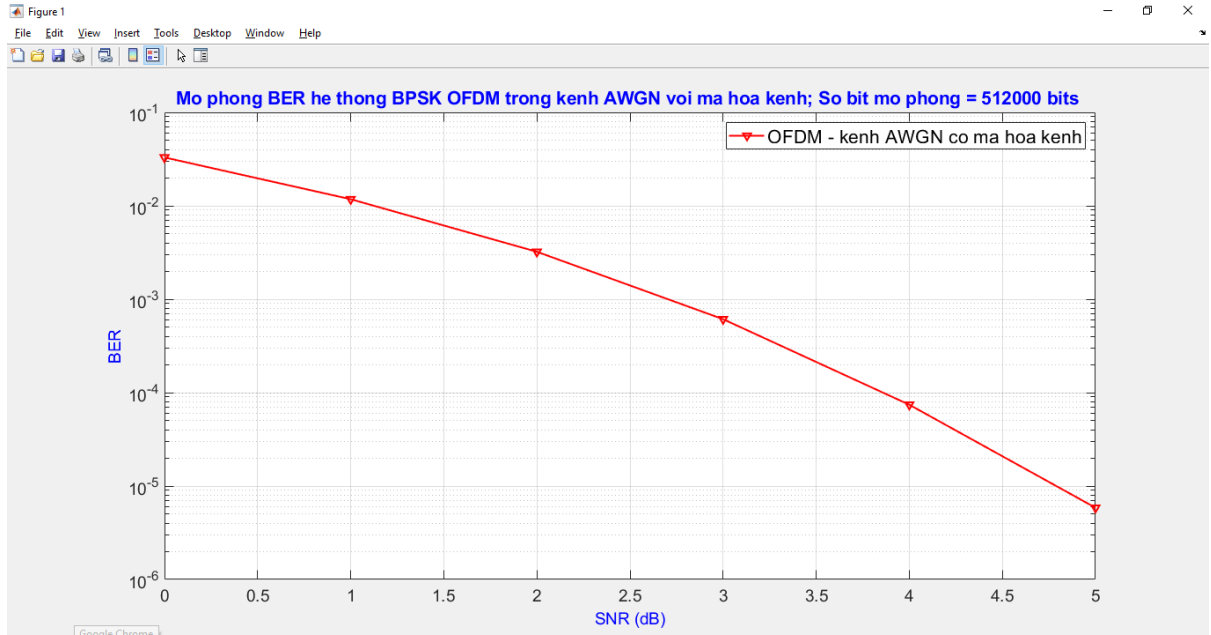
Tham số	Giải thích
SNR = [0:1:9];	Mảng chứa các giá trị Signal-to-Noise Ratio (SNR) đầu vào
FFTsize = 512;	Kích thước phép biến đổi Fourier (FFT) trong hệ thống OFDM.
CPsize = 20;	Kích thước cyclic prefix (CP), một phần của tín hiệu OFDM được thêm vào phía trước để giảm nhiễu

numRun = 10^3;	Số lần chạy mô phỏng, được sử dụng để tính tỷ lệ lỗi trung bình.
NumBits = FFTsize*numRun;	Tổng số bit dữ liệu được truyền
mode_Sim = 2;	Xác định chế độ mô phỏng: 1 cho không có mã hoá kênh (OFDM không có mã hoá kênh); 2 cho mã hoá kênh (OFDM với mã hoá kênh).
Codding_Type = 1;	Xác định loại mã Convolutional Code được sử dụng nếu mode_Sim là 2.
<pre> if mode_Sim ==2 if Codding_Type == 1 k0 = 1; G = [1 1 1;1 0 1]; elseif Codding_Type == 2 k0 = 1; G = [1 1 1 1 0 0 1;1 0 1 1 0 1 0]; else k0 = 2; G = [0 0 1 0 1 0 0 1;0 0 0 0 0 0 0 1;1 0 0 0 0 0 0 1]; end end </pre>	Xác định ma trận mã: Dựa vào Codding_Type, chương trình xác định ma trận mã G và thông số k0.
<pre> for n = 1:length(SNR), errCount = 0; for k = 1:numRun numSymbols = FFTsize; data = 0.5*(sign(rand(1,numSymbols)-0.5)+1); data2 = 2*data-1; if mode_Sim ==2 inputSymbols = FWC_COV_Encoder(G,k0,data); inputSymbols = 2*inputSymbols-1; else inputSymbols = 2*data-1; end end end </pre>	<p>Vòng lặp qua các giá trị SNR: Mỗi lần lặp, biến errCount được sử dụng để đếm số lỗi ký hiệu trong quá trình mô phỏng.</p> <p>Dữ liệu BPSK ngẫu nhiên được tạo và mã hoá (nếu mode_Sim là 2) sử dụng mã Convolutional Code.</p>

<pre> TxSamples = sqrt(length(inputSymbols))* ifft(inputSymbols,length(inputSymbols)); numSymbols = length(inputSymbols); Tx_ofdm = [TxSamples(numSymbols- CPsize+1:numSymbols) TxSamples]; </pre>	<p>Dữ liệu sau mã hoá (hoặc dữ liệu BPSK ban đầu) được biến đổi thành tín hiệu OFDM và thêm cyclic prefix (CP).</p>
<pre> numSymbols_2 = length(inputSymbols); tmp = randn(1,numSymbols_2+CPsize); noisePower = 10^(- SNR(n)/10); RxSymbols = Tx_ofdm + sqrt(noisePower)*tmp; </pre>	<p>Nhiều Gaussian ngẫu nhiên được tạo với công suất nhiễu được tính dựa trên giá trị SNR. Nhiều này được thêm vào tín hiệu OFDM.</p>
<pre> EstSymbols_1 = RxSymbols(CPsize+1:numSymbols_2+CPsize); </pre>	<p>Cyclic prefix được loại bỏ để chuẩn bị cho việc giải mã tín hiệu.</p>
<pre> Y = fft(EstSymbols_1,length(EstSymbols_1)); EstSymbols_1 = Y; EstSymbols_1 = sign(real(EstSymbols_1)); for i = 1:length(EstSymbols_1) if EstSymbols_1(i)>0 Decis(i)= 1; else Decis(i)= 0; end end </pre>	<p>Thực hiện phép biến đổi FFT trên tín hiệu để giải mã.</p>
<pre> if mode_Sim==2 EstSymbols = FWC_COV_Dencoder(G,k0,Decis); else EstSymbols = EstSymbols_1; end </pre>	<p>Quyết định các ký hiệu sau FFT và giải mã bằng mã Convolutional Code (nếu mode_Sim là 2).</p>
<pre> if mode_Sim==2 I = find((data-EstSymbols) == 0); else I = find((data2-EstSymbols) == 0); end % Countered Errors errCount = errCount + (FFTsize-length(I)); end </pre>	<p>So sánh dữ liệu gốc (hoặc dữ liệu giữa) và dữ liệu sau giải mã để xác định số lỗi ký hiệu trong mỗi lần chạy mô phỏng.</p>
<pre> SER(n,:) = errCount / (FFTsize*numRun); </pre>	<p>SER là tỷ lệ lỗi ký hiệu tính được cho mỗi giá trị SNR.</p>

End

❖ Kết quả mô phỏng



Workspace	
Name ▲	Value
Codding_Type	1
CPsize	20
data	1x512 double
data2	1x512 double
errCount	0
FFTsize	512
G	[1,1,1;1,0,1]
k	1
k0	1
mode_Sim	2
n	1
NumBits	512000
numRun	1000
numSymbols	512
SNR	[0,1,2,3,4,5,6,7,8,9]

Sim_MA08: Mô hình hóa và mô phỏng hiệu năng SER cho hệ thống truyền dẫn OFDM trong môi trường kênh AWGN và kênh pha đỉnh đa đường

1, Mục đích và nội dung

a, Mục đích:

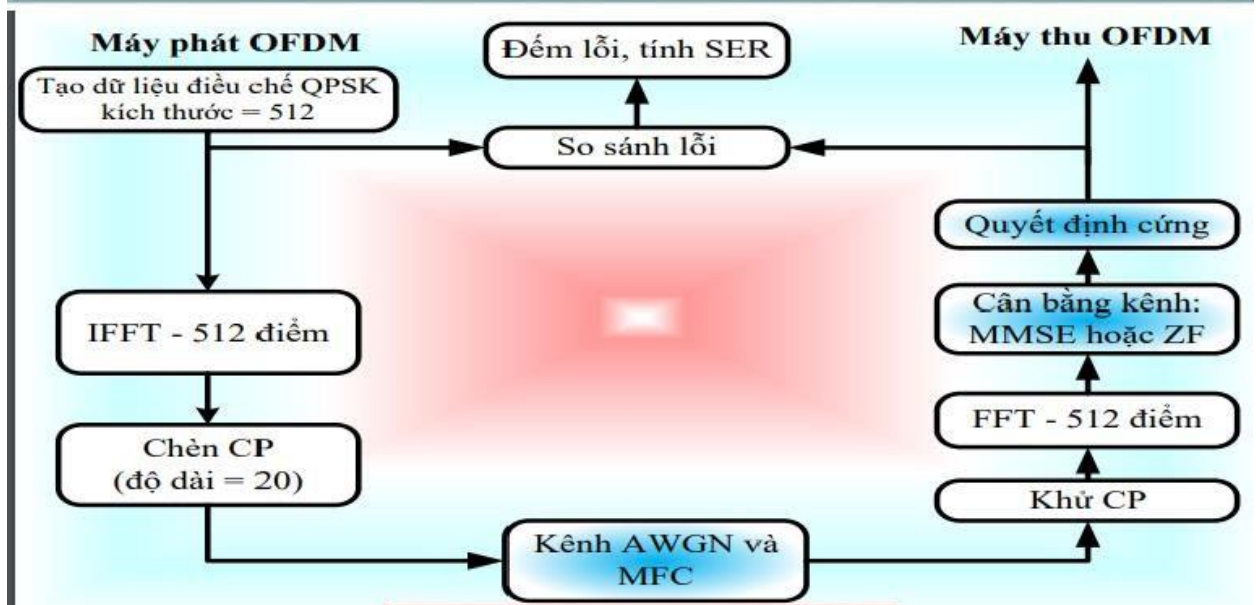
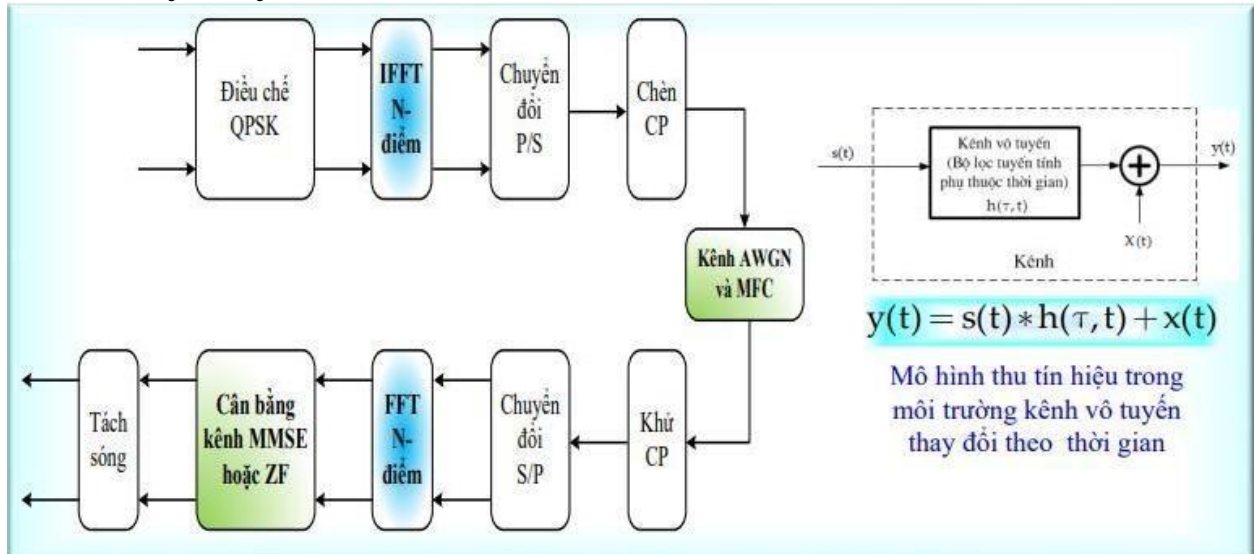
- Mô hình hóa và trực quan hóa nguyên lý hoạt động của hệ thống truyền dẫn OFDM trong môi trường kênh AWGN và kênh pha đỉnh đa đường.
- Matlab hóa và mô phỏng hệ thống truyền dẫn OFDM trong môi trường kênh AWGN và kênh pha đỉnh đa đường để: Làm sáng tỏ nguyên lý hoạt động và phân tích đánh giá hiệu năng.

b, Nội dung:

- Xây dựng và trình bày nguyên lý hoạt động quá trình điều chế/giải điều chế OFDM trên cơ sở không gian tín hiệu.
- Mô hình kênh AWGN và kênh pha đỉnh đa đường: Mô hình kênh đa đường/tham số (lý lịch trễ công suất).
- Kênh pha đỉnh đa đường và cân bằng kênh MMSE/ZF.
- Mô hình hóa và nguyên lý hoạt động hệ thống truyền dẫn OFDM trong môi trường kênh AWGN và kênh pha đỉnh đa đường.
- Tiến trình mô phỏng: Lưu đồ mô phỏng và thực hiện mô phỏng hệ thống truyền dẫn OFDM trong môi trường kênh AWGN và kênh pha đỉnh đa đường.
- Matlab hóa và mô phỏng hiệu năng SER của hệ thống OFDM trong môi trường kênh AWGN và kênh pha đỉnh đa đường.
 - Thiết lập kịch bản mô phỏng: Định nghĩa tham số và thiết lập tập tham số đầu vào cho chương trình mô phỏng.

- Matlab hóa mô hình mô phỏng hệ thống OFDM trong môi trường kênh AWGN và kênh pha đình đa đường.
- Thiết lập các bước mô phỏng và thực hiện mô phỏng theo kịch bản mô phỏng.
- Thực hiện mô phỏng trên Matlab để: làm sáng tỏ nguyên lý hoạt động và khảo sát so sánh đánh giá hiệu năng hệ thống truyền dẫn OFDM trong môi trường kênh AWGN và kênh pha đình đa đường, hiệu năng của bộ cân bằng kênh MMSE/ZF.

2, Cơ sở lý thuyết



3, Code matlab

```

=====
=====
%===== Sim_MA_08_QPSK_OFDM_MFC
=====

```

```

%===== NVD_D12VT_SER_OFDM_AWGN_MFC
=====
% ----- Comparision: SER of OFDM in AWGN & MFC -----
-----
%=====
=====
clc;
clear all;
close all;
%
%-----
SNR          = [1:2:18];
FFTsize      = 512;
CPsize       = 20;
numRun       = 10^4;          % 10^3
dataType     = 'Q-PSK';      % 'Q-PSK'
SER_ofdm_AWGN = [];
SER_ofdm_pedA = [];
SER_ofdm_vehA = [];
%
%-----
for n = 1:length(SNR),
    %-----
    channelType = 'AWGN';
    SER_ofdm_AWGN(n) = MA_08_SER_ofdm(SNR(n), numRun, ...
        FFTsize, dataType, CPsize, channelType, []);
    %-----
    channelType = 'pedA';
    equalizerType = 'ZERO'; % notechon 'ZERO' hoac 'MMSE'
    SER_ofdm_pedA_ZF(n) =
MA_08_SER_ofdm(SNR(n), numRun, FFTsize, dataType, ...
        CPsize, channelType, equalizerType);
    %-----
    channelType = 'vehA';
    equalizerType = 'ZERO'; % notechon 'ZERO' hoac 'MMSE'
    SER_ofdm_vehA_ZF(n) =
MA_08_SER_ofdm(SNR(n), numRun, FFTsize, dataType, ...
        CPsize, channelType, equalizerType);
    %-----
    channelType = 'pedA';
    equalizerType = 'MMSE'; % notechon 'ZERO' hoac 'MMSE'
    SER_ofdm_pedA_MMSE(n) =
MA_08_SER_ofdm(SNR(n), numRun, FFTsize, dataType, ...
        CPsize, channelType, equalizerType);
    %-----
    channelType = 'vehA';
    equalizerType = 'MMSE'; % notechon 'ZERO' hoac 'MMSE'
    SER_ofdm_vehA_MMSE(n) =
MA_08_SER_ofdm(SNR(n), numRun, FFTsize, dataType, ...
        CPsize, channelType, equalizerType);
end
%=====

```

```

save Sim_MA_08_QPSK_OFDM_MFC.mat;
%=====

figure(1)
G = semilogy(SNR,SER_ofdm_AWGN','-vk',SNR,SER_ofdm_pedA_ZF','-
*r',SNR,SER_ofdm_vehA_ZF','-sb');
set(G,'LineWidth',[1.5]);
X =xlabel('SNR (dB)');
set(X,'fontname','.Vntime','fontsize',14,'color','b');
Y =ylabel('SER');
set(Y,'fontname','.Vntime','fontsize',14,'color','b');
L =legend('OFDM - kênh AWGN','OFDM - kênh pha dinh moi truong di
bo','OFDM - kênh pha dinh moi truong xe co');
set(L,'fontname','.Vntime','fontsize',13);
grid on
T=title(strvcat(strcat('SER cua he thong OFDM trong cac mo',...
' hinh kênh khác nhau'),' va phuong phap can bang kênh
ZF'));
set(T,'fontname','.Vntime','fontsize',18,'color','b');

figure(2)
G = semilogy(SNR,SER_ofdm_AWGN','-vk',SNR,SER_ofdm_pedA_MMSE','-
*r',SNR,SER_ofdm_vehA_MMSE','-sb');
set(G,'LineWidth',[1.5]);
X =xlabel('SNR (dB)');
set(X,'fontname','.Vntime','fontsize',14,'color','b');
Y =ylabel('SER');
set(Y,'fontname','.Vntime','fontsize',14,'color','b');
L =legend('OFDM - kênh AWGN','OFDM - kênh pha dinh moi truong di
bo','OFDM - kênh pha dinh moi truong xe co');
set(L,'fontname','.Vntime','fontsize',13);
grid on
T=title(strvcat(strcat('SER cua he thong OFDM trong cac mo',...
' hinh kênh khác nhau'),' va phuong phap can bang kênh
MMSE'));
set(T,'fontname','.Vntime','fontsize',18,'color','b');

figure(3)
G = semilogy(SNR,SER_ofdm_AWGN','-vk',SNR,SER_ofdm_pedA_MMSE','-
*r',SNR,SER_ofdm_vehA_MMSE','-sb',...
SNR,SER_ofdm_pedA_ZF','-vb',SNR,SER_ofdm_vehA_ZF','-vr');
set(G,'LineWidth',[1.5]);
X =xlabel('SNR (dB)');
set(X,'fontname','.Vntime','fontsize',14,'color','b');
Y =ylabel('SER');
set(Y,'fontname','.Vntime','fontsize',14,'color','b');
L =legend('OFDM - kênh AWGN','OFDM - Kênh MFC/pedA & Bo loc
MMSE','OFDM - Kênh MFC/vehA & Bo loc MMSE',...
'OFDM - Kênh MFC/pedA & Bo loc ZF','OFDM - Kênh MFC/vehA & Bo
loc ZF');
set(L,'fontname','.Vntime','fontsize',13);
grid on

```

```

T=title(strvcat(strcat('SER của hệ thống OFDM trong các mô',...
    ' hình kênh khác nhau'),'          và phương pháp cân bằng kênh
MMSE & ZF')));
set(T,'fontname','.Vntime','fontsize',18,'color','b');
%=====
=====

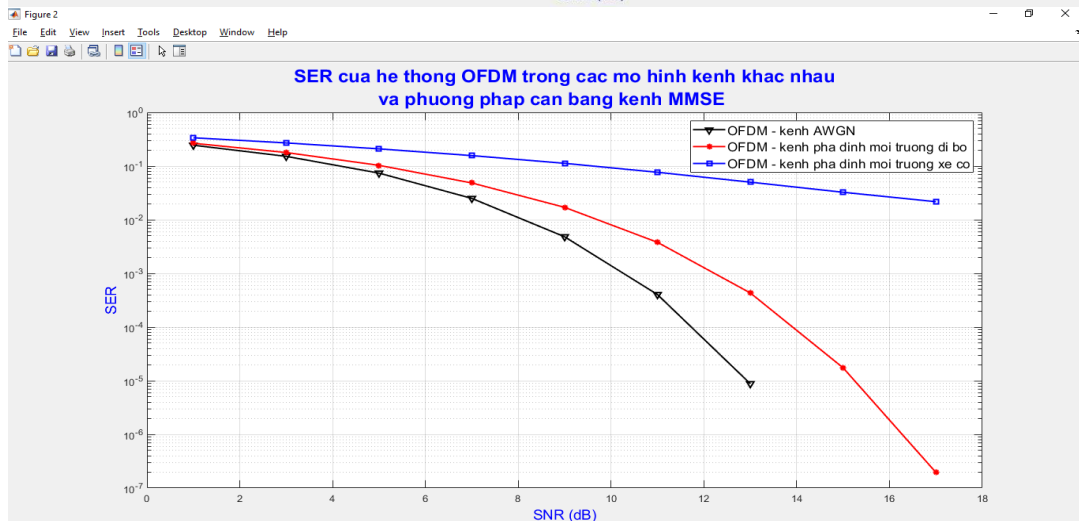
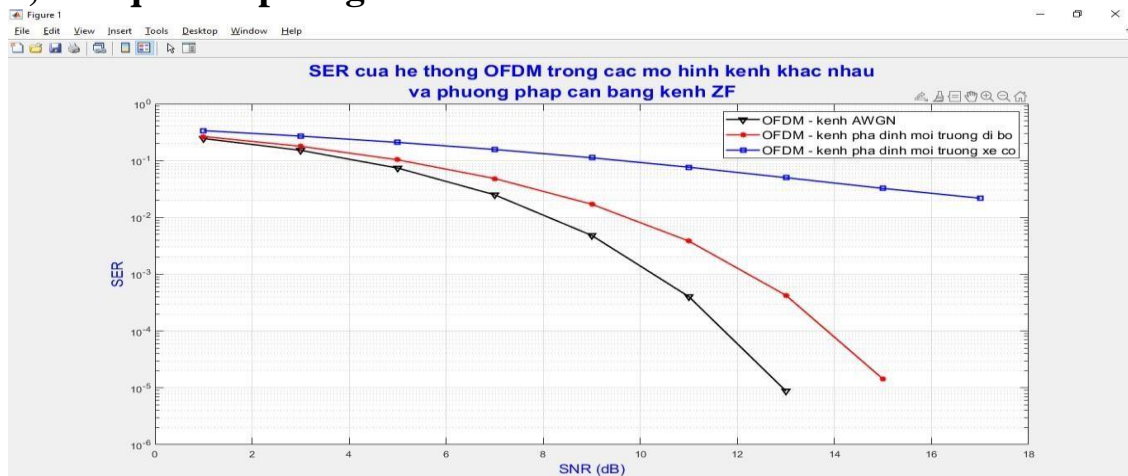
```

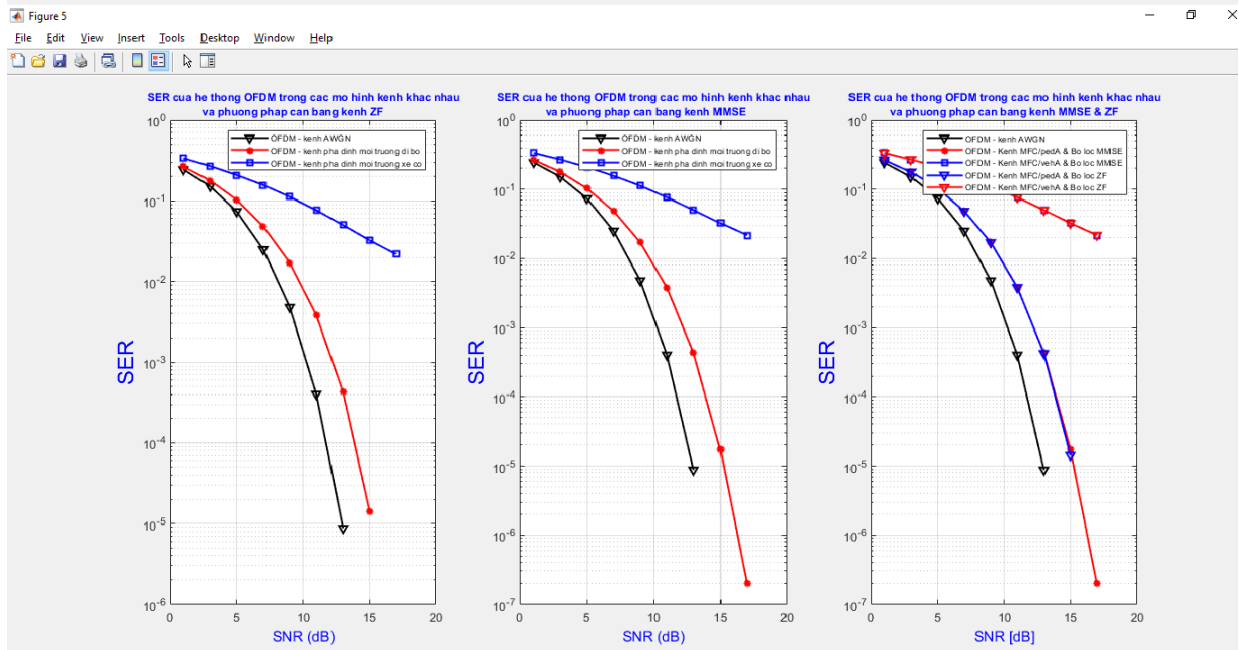
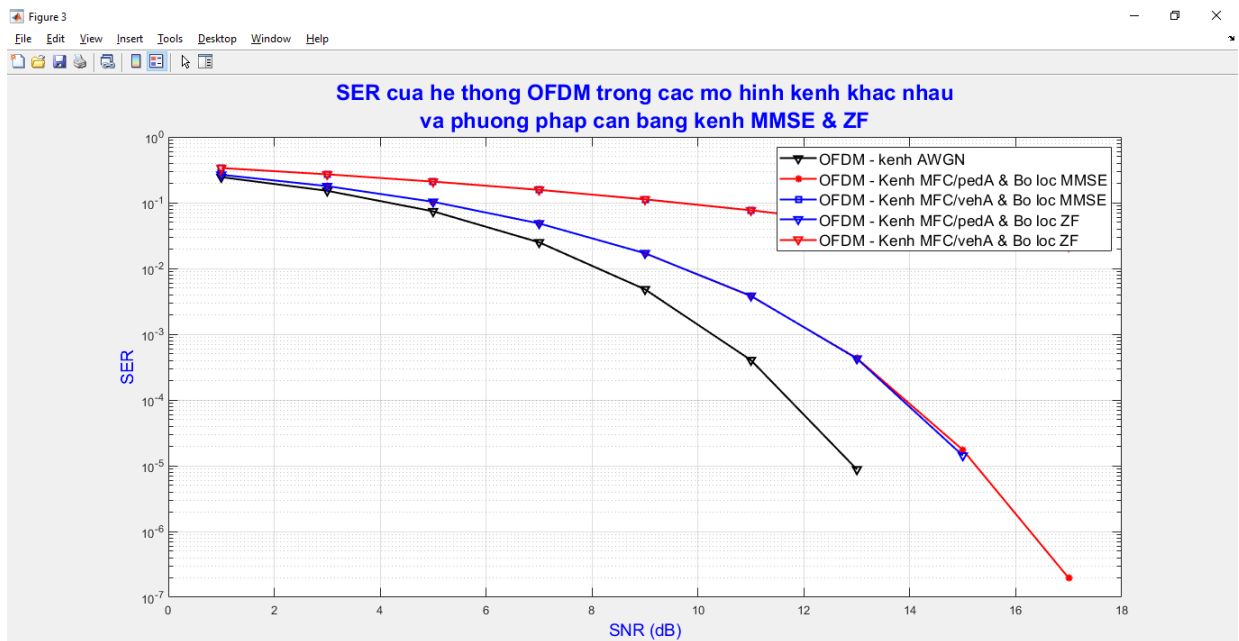
❖ Giải thích code

Tham số	Giải thích
SNR = [1:2:18];	Một mảng chứa các giá trị SNR từ 1 đến 18 với khoảng cách 2 đơn vị.
FFTsize = 512;	Kích thước của FFT trong hệ thống OFDM
CPsize = 20;	Kích thước của cyclic prefix (CP) trong OFDM
numRun = 10 ⁴ ;	Số lần chạy mô phỏng
dataType = 'Q-PSK';	Loại tín hiệu đang được sử dụng
SER_ofdm_AWGN =[]; SER_ofdm_pedA =[]; SER_ofdm_vehA =[];	Các biến dùng để lưu tỷ lệ lỗi ký phân (SER) tương ứng cho các trường hợp.
Lệnh	Giải thích
for n = 1:length(SNR), channelType = 'AWGN'; SER_ofdm_AWGN(n) = MA_08_SER_ofdm(SNR(n),numRun,... FFTsize,dataType,CPsize,channelType,[]);	Dòng đầu tiên tính SER cho hệ thống OFDM trong môi trường AWGN (Nhiều trắng Gaussian) bằng cách gọi hàm MA_08_SER_ofdm với các tham số tương ứng.
channelType = 'pedA'; equalizerType = 'ZERO'; SER_ofdm_pedA_ZF(n) = MA_08_SER_ofdm(SNR(n),numRun,FFTsize,dataType, pe,... CPsize,channelType,equalizerType); channelType = 'vehA'; equalizerType = 'ZERO'; SER_ofdm_vehA_ZF(n) = MA_08_SER_ofdm(SNR(n),numRun,FFTsize,dataType, pe,...	SER_ofdm_pedA_ZF và SER_ofdm_vehA_ZF tính SER cho bù trừ Zero-Forcing.

<pre> CPsize,channelType,equalizerType); channelType = 'pedA'; equalizerType = 'MMSE'; SER_ofdm_pedA_MMSE(n) = MA_08_SER_ofdm(SNR(n),numRun,FFTsize,dataTy pe,... CPsize,channelType,equalizerType); channelType = 'vehA'; equalizerType = 'MMSE'; SER_ofdm_vehA_MMSE(n)= MA_08_SER_ofdm(SNR(n),numRun,FFTsize,dataTy pe,... CPsize,channelType,equalizerType); End </pre>	SER_ofdm_pedA_MM SE và SER_ofdm_vehA_MM SE tính SER cho bù trừ Minimum Mean Square Error.
--	--

4, Kết quả mô phỏng





Workspace		
Name ▲	Value	
channelType	'vehA'	
CPsize	20	
dataType	'Q-PSK'	
equalizerType	'MMSE'	
FFTsize	512	
G	5x1 Line	
L	1x1 Legend	
n	9	
numRun	10000	
SER_ofdm_AWGN	[0.2445,0.1517,0.0741,...	
SER_ofdm_pedA	[]	
SER_ofdm_pedA_...	[0.2653,0.1786,0.1032,...	
SER_ofdm_pedA_ZF	[0.2660,0.1787,0.1030,...	
SER_ofdm_vehA	[]	
SER_ofdm_vehA_...	[0.3371,0.2699,0.2093,...	
SER_ofdm_vehA_ZF	[0.3368,0.2699,0.2090,...	
SNR	[1,3,5,7,9,11,13,15,17]	
T	1x1 Text	
X	1x1 Text	
Y	1x1 Text	

Sim FWC05: Mô hình hóa và mô phỏng hệ thống đa Anten SVD MIMO

1, Mục đích và nội dung

a, Mục đích:

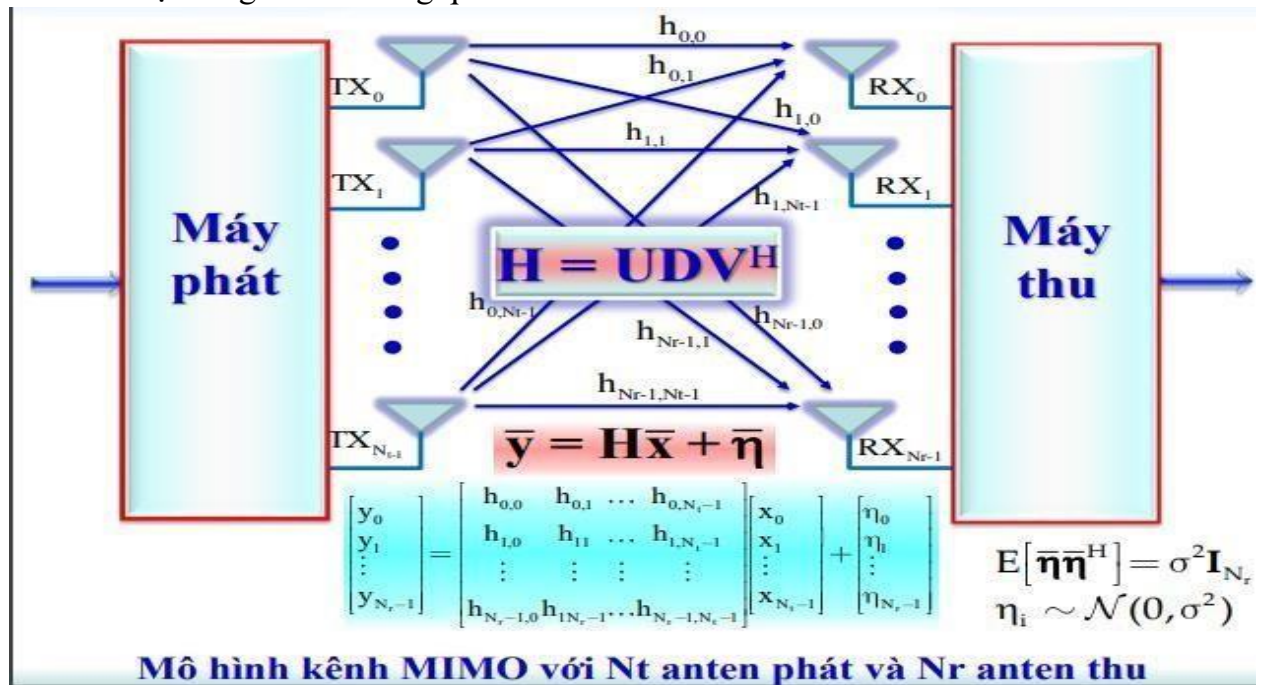
Mô hình hóa và mô phỏng/hệ thống đa Anten MIMO trong môi trường truyền sóng pha đình phân bố Rayleigh trên cơ sở SVD.

b, Nội dung:

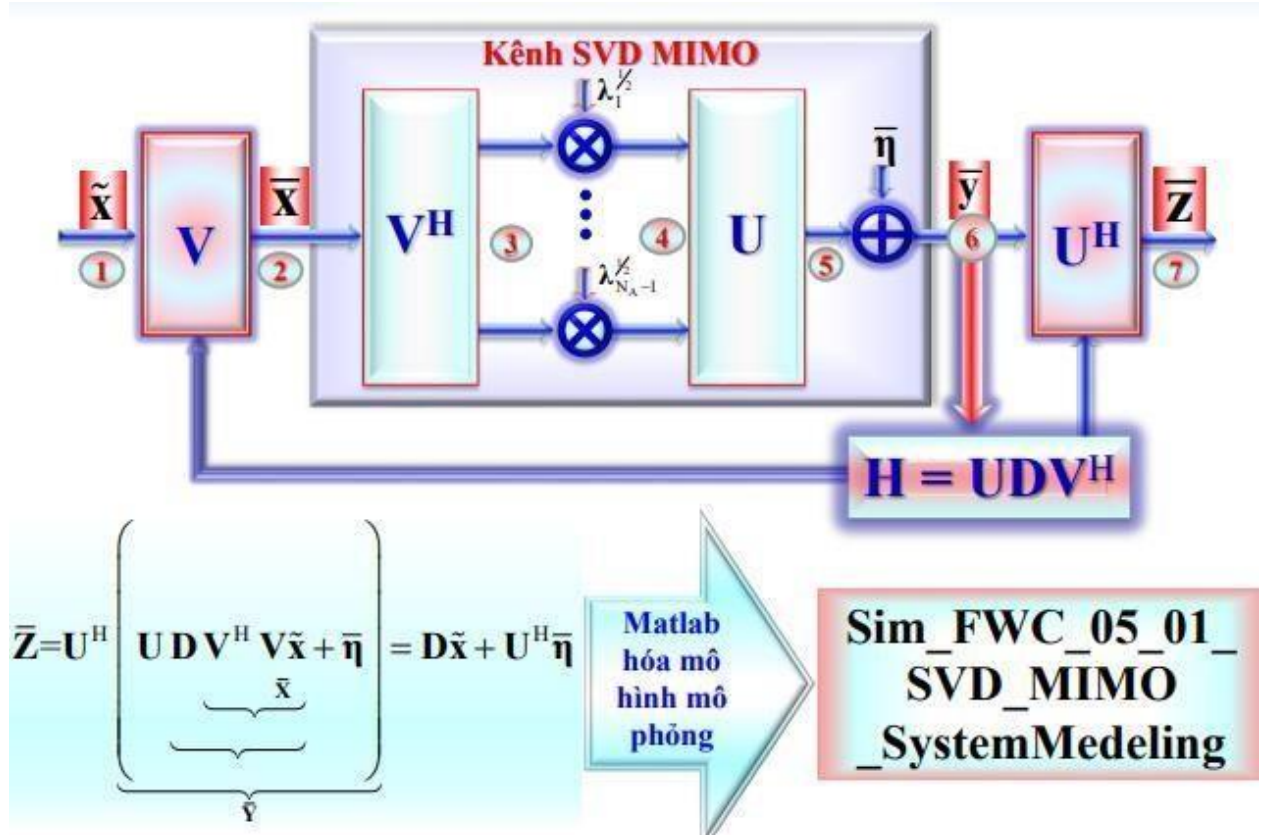
- Tóm tắt lý thuyết: Mô hình kênh và hệ thống MIMO tổng quát; mô hình kênh và hệ thống SVD MIMO, kênh SISO trong môi trường truyền sóng pha đình phân bố Rayleigh.
- Mô hình hóa và mô phỏng kênh SVD MIMO trong môi trường truyền sóng pha đình Rayleigh:
 - Mô hình hóa kênh MIMO trên cơ sở SVD.
 - Matlab hóa mô hình kênh MIMO trên cơ sở SVD.
 - Mô phỏng kênh SVD MIMO trong môi trường truyền sóng pha đình Rayleigh.
- Mô hình hóa và mô phỏng hệ thống SVD MIMO tối ưu trong môi trường truyền sóng pha đình phân bố Rayleigh:
 - Mô hình hóa hệ thống MIMO trên cơ sở SVD.
 - Matlab hóa mô hình hệ thống MIMO trên cơ sở SVD.
 - Mô phỏng hệ thống SVD MIMO trong môi trường truyền sóng pha đình Rayleigh.

2, Cơ sở lý thuyết

- Mô hình hệ thống MIMO tổng quát



- Mô hình hóa hệ thống SVD MIMO tối ưu



3, Code matlab

```

%=====
=====
%===== Sim_FWC_03_02_BPSK_AWGN_ChannelCode
=====
%=====

% function NVD_D12VT_SVDMIMO_SystemMedeling

clc;
clear all;
close all;

N_Tx      = 8;      % Num of Tx anten
N_Rx      = 4;      % Num of Rx anten
fD        = 10;     % Doppler Frequency
N_symbol  = 100;    % No of symbol 2048
T_sim     = 0.1;    % Simulation Time

%=====
=====
% Gaussian Distribution
H_Gaussian = zeros(N_Rx,N_Tx,N_symbol);
for k = 1:N_symbol
    H_Gaussian(:, :, k) = (randn(N_Rx,N_Tx) +
i*randn(N_Rx,N_Tx))/sqrt(2);
end
H_Gau      = zeros(N_Rx,N_Tx);
H_Gau      = H_Gaussian(:, :, 1);

%=====
=====
% Rayleigh Distribution
H_Rayleigh = zeros(N_Rx,N_Tx,N_symbol);
for m = 1:N_Rx
    for n = 1:N_Tx
        H_Rayleigh(m,n,:) = (FWC_Rayleigh_Channel(T_sim,N_symbol,fD)
...
+
i*FWC_Rayleigh_Channel(T_sim,N_symbol,fD))/sqrt(2);
    end
end
H_Ray      = H_Rayleigh(:, :, 1);

%=====
=====
% SVD for MIMO channel and Dispaly

disp('      Ma tran kenh H');
H_Ray
disp('      Kich thuoc H');
disp(size(H_Ray))

```

```

disp('          Phan tich SVD cua H: [U, D, V] = svd(H) ');

[U_Gau, D_Gau, V_Gau] = svd(H_Gau);
[U_Ray, D_Ray, V_Ray] = svd(H_Ray);

disp('          Ma tran tien ma hoa V ');
% disp(V);
V_Ray
disp('          Kich thuoc size(V) ');
disp(size(V_Ray));

disp('          Ma tran hau ma hoa U ');
U_Ray
disp('          Kich thuoc size(U) ');
disp(size(U_Ray));

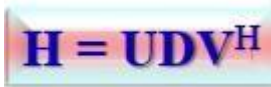
disp('          Ma tran duong cheo D ');
D_Ray
disp('          Kich thuoc size(D) ');

disp(size(D_Ray));

```

❖ Giải thích code

Tham số	Giải thích
N_Tx = 8;	Số lượng anten phát là 8
N_Rx = 4;	Số lượng anten thu là 4
fD = 10;	Tần số Doppler là 10
N_symbol = 100;	Thiết lập giá trị số kí hiệu
T_sim = 0.1;	Thời gian mô phỏng
Lệnh	Giải thích
H_Gaussian = zeros(N_Rx,N_Tx,N_symbol);	Tạo ma trận H_Gaussian có 4 hàng, 8 cột, các phần tử đều là 0
for k = 1:N_symbol H_Gaussian(:, :, k) = (randn(N_Rx,N_Tx) + i*randn(N_Rx,N_Tx))/sqrt(2); end	Thiết lập ma trận H_Gaussian thứ k
H_Gau = zeros(N_Rx,N_Tx);	Sinh ma trận 0 H_gauss kích thước N_Rx,N_Tx
H_Gau = H_Gaussian(:, :, 1);	Chọn ma trận gauss = giá trị ma trận gauss thứ 1 trong mảng H_Gaussian
H_Rayleigh = zeros(N_Rx,N_Tx,N_symbol);	Tạo ma trận H_Rayleigh có 4 hàng,

	8 cột, các phần tử đều là 0
<pre> for m = 1:N_Rx for n = 1:N_Tx H_Rayleigh(m,n,:) = (FWC_Rayleigh_Channel(T_sim,N_symbol,fD) ... + i*FWC_Rayleigh_Channel(T_sim,N_symbol,fD))/sqrt(2); end end </pre>	Thiết lập ma trận H_Rayleigh thứ k
<pre> H_Ray = H_Rayleigh(:, :, 1); </pre>	Chọn ma trận rayleigh = giá trị ma trận rayleigh thứ 1 trong mảng H_Rayleigh
<pre> [U_Gau, D_Gau, V_Gau] = svd(H_Gau); [U_Ray, D_Ray, V_Ray] = svd(H_Ray); </pre>	

4, Kết quả mô phỏng

```

Ma tran kênh H

H_Ray =

Columns 1 through 6

-0.0537 + 0.4004i    0.1606 - 0.2477i    0.7740 + 0.8569i    0.3010 + 0.3950i    -0.2809 - 0.2363i    -0.5640 + 0.1307i
-0.3120 + 0.4027i    0.1017 + 0.8871i    -0.6359 + 0.5442i    0.5199 - 1.2061i    0.8227 + 0.6054i    0.2336 + 0.3984i
1.1334 + 1.0910i    -0.1297 - 0.7210i    0.2065 - 0.3661i    1.3100 - 0.5496i    0.7454 + 0.2516i    0.0681 - 0.2699i
1.3423 + 0.7040i    -0.4225 + 0.4034i    0.2134 - 0.2362i    -0.7645 - 1.3983i    -0.4951 + 1.8873i    -0.9298 + 0.2508i

Columns 7 through 8

-0.3634 + 0.1862i    -0.1931 + 0.3374i
-0.8028 + 0.2209i    -0.8174 - 0.0407i
0.5087 - 0.4210i    1.1560 - 0.2452i
-0.0909 - 0.6847i    0.2917 + 0.3325i

Kích thước H
4      8

Phân tích SVD của H: [U, D, V] = svd(H)
Ma trận tiền ma hóa V

V_Ray =

Columns 1 through 6

0.4312 + 0.3084i    -0.0396 - 0.3746i    0.2063 + 0.1586i    0.2137 + 0.0450i    -0.4407 - 0.1146i    0.1109 - 0.1685i
-0.0745 - 0.2128i    0.3305 - 0.1610i    -0.0773 + 0.1312i    -0.0829 + 0.1293i    -0.5798 - 0.0728i    -0.4025 - 0.1242i
-0.1419 + 0.1039i    0.2114 - 0.3193i    0.2407 - 0.0531i    0.3790 - 0.4482i    0.2991 + 0.0669i    0.1502 - 0.3693i
-0.4311 + 0.3231i    -0.3313 + 0.0979i    -0.5202 + 0.2079i    0.2526 - 0.2099i    -0.1840 + 0.0581i    -0.1440 - 0.1329i
0.3736 - 0.2587i    0.0629 - 0.3044i    -0.5008 + 0.4043i    0.0717 + 0.1165i    0.4535 + 0.0547i    -0.1697 - 0.0375i
-0.0604 - 0.2005i    -0.0285 - 0.1070i    -0.1985 + 0.2228i    -0.4453 - 0.2193i    -0.2240 + 0.1449i    0.6931 - 0.0843i
-0.2173 + 0.1270i    -0.0659 - 0.2581i    0.1483 + 0.0080i    -0.4283 + 0.0692i    0.1485 + 0.0969i    -0.0865 + 0.1715i
0.0176 + 0.1813i    -0.1107 - 0.5147i    -0.0761 - 0.0580i    -0.0711 + 0.1340i    0.0279 - 0.0895i    0.1081 + 0.1351i

Columns 7 through 8

0.0979 + 0.1791i    -0.3219 + 0.2614i
0.1058 - 0.3665i    0.1795 - 0.2659i
0.1299 - 0.3121i    0.0803 - 0.2088i
0.1405 - 0.0276i    0.0585 + 0.2629i
0.0494 - 0.0608i    -0.1345 + 0.0274i
-0.0944 - 0.1901i    -0.0626 + 0.0276i
0.7532 + 0.0466i    -0.1325 + 0.0090i
-0.1415 + 0.1933i    0.7489 + 0.0247i

Kích thước size(V)
8      8

```

Ma tran hau ma hoa U

U_Ray =

```
-0.2127 + 0.0000i    0.2581 + 0.0000i    0.1045 + 0.0000i    0.9366 + 0.0000i
 0.2149 + 0.0483i    0.2982 + 0.5822i   -0.5521 + 0.4233i    0.0283 - 0.1967i
-0.0209 + 0.5164i   -0.2122 - 0.5380i   -0.3639 + 0.4597i    0.0943 + 0.2142i
 0.4663 + 0.6494i    0.4075 + 0.0709i    0.2099 - 0.3425i   -0.0298 + 0.1662i
```

Kich thước size(U)

4 4

Ma tran duong cheo D

D_Ray =

```
3.6796    0    0    0    0    0    0    0
 0    2.6028    0    0    0    0    0    0
 0    0    2.2627    0    0    0    0    0
 0    0    0    1.3069    0    0    0    0
```

Kich thước size(D)

4 8

Workspace		
Name ▲	Value	
D_Gau	4x8 double	
D_Ray	4x8 double	
fD	10	
H_Gau	4x8 complex double	
H_Gaussian	4x8x100 complex dou...	
H_Ray	4x8 complex double	
H_Rayleigh	4x8x100 complex dou...	
k	100	
m	4	
n	8	
N_Rx	4	
N_symbol	100	
N_Tx	8	
T_sim	0.1000	
U_Gau	4x4 complex double	
U_Ray	4x4 complex double	
V_Gau	8x8 complex double	
V_Ray	8x8 complex double	

Nhận xét:

- Với số anten phát $T_x = 4$, Anten thu $R_x = 8 \Rightarrow$ Ta thu được ma trận H kích thước 4×8 đúng theo như lý thuyết.
- Phân tích thành phần ma trận H theo SVD với $H = U \cdot D \cdot V(H)$ ta thu được các ma trận con U, V, D.
- Ma trận U kích thước 4×4 là ma trận nhất phân.

- Ma trận V kích thước 8×8 là ma trận nhất phân.
- Ma trận D kích thước 4×8 có 4 giá trị đơn không âm trên đường chéo chính của D .
- Với mỗi ký hiệu đi qua kênh \Rightarrow kênh lấy mẫu một lần và H thay đổi theo mỗi ký hiệu.

Sim FWC06: Mô hình hóa và mô phỏng dung lượng của hệ thống SVD MIMO

1, Mục đích và nội dung

a, Mục đích:

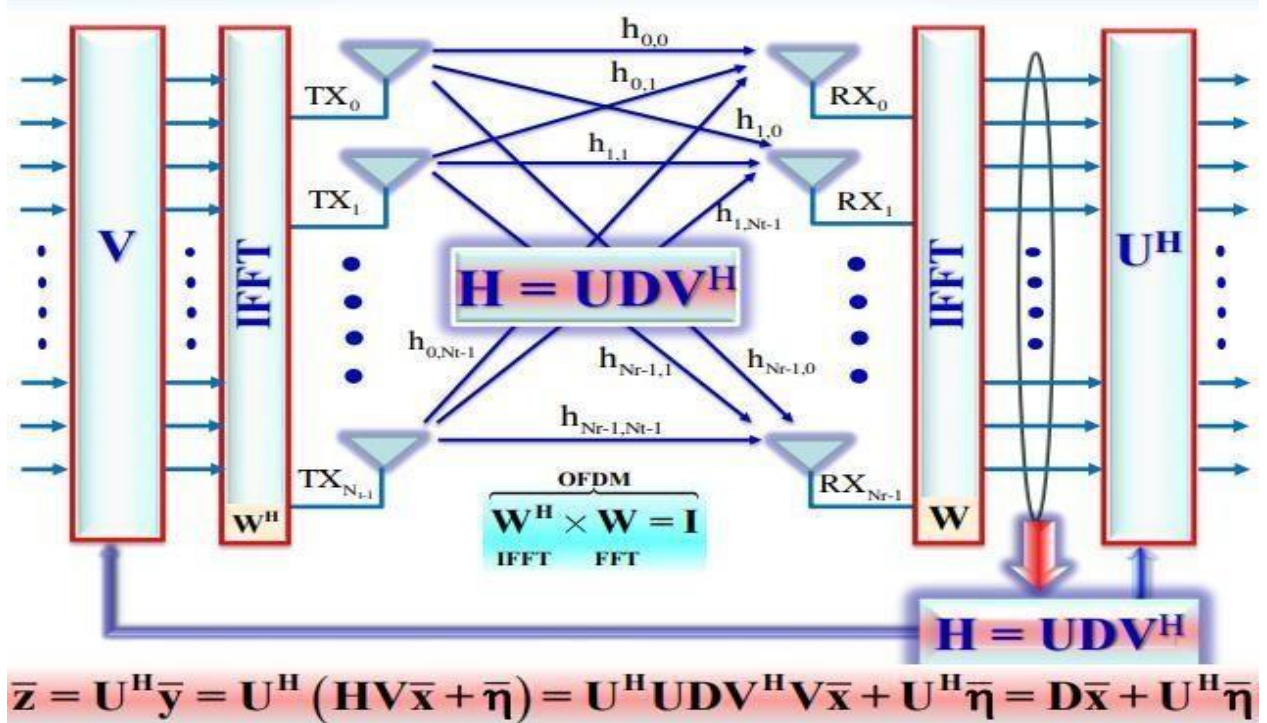
Mô hình hóa và mô phỏng dung lượng của hệ thống đa anten MIMO trong môi trường truyền sóng pha đỉnh phân bố Rayleigh.

b, Nội dung:

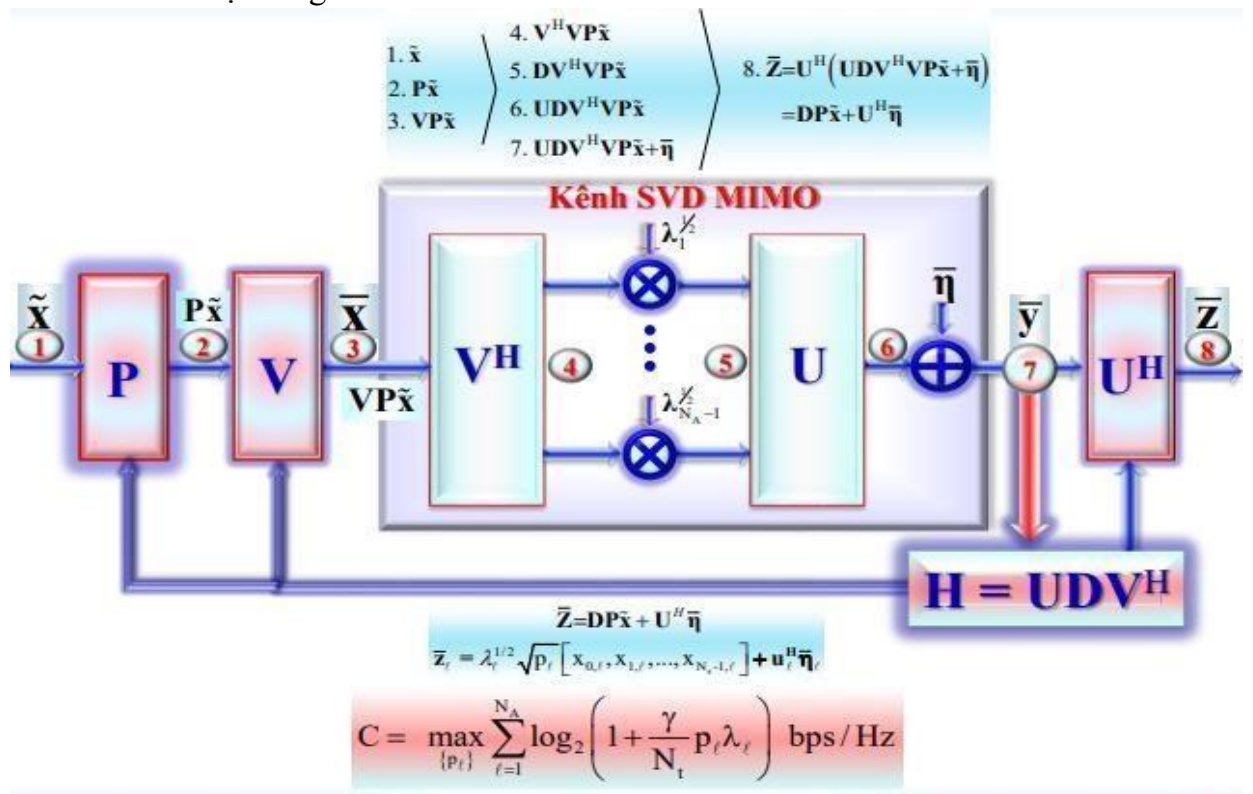
- Tóm tắt lý thuyết: Mô hình kênh và mô hình hệ thống MIMO; mô hình kênh và hệ thống SVD MIMO, kênh SISO trong môi trường truyền sóng pha đỉnh phân bố Rayleigh; lý thuyết dung lượng kênh (thiết lập công thức dung lượng kênh MIMO).
- Mô hình hóa và mô phỏng dung lượng kênh MIMO ngẫu nhiên phân bố Rayleigh khi không có thông tin trạng thái kênh CSI ở phía phát (hệ thống MIMO vòng hở OL):
 - Mô hình hóa hệ thống MIMO trên cơ sở SVD
 - Thiết lập dung lượng kênh của hệ thống MIMO vòng hở.
 - Matlab hóa mô hình hệ thống và dung lượng hệ thống MIMO ngẫu nhiên vòng hở.
 - Mô phỏng dung lượng của hệ thống MIMO vòng hở trong môi trường truyền sóng pha đỉnh Rayleigh.
- Mô hình hóa và mô phỏng dung lượng kênh MIMO ngẫu nhiên phân bố Rayleigh khi có thông tin trạng thái kênh CSI ở phía phát (hệ thống MIMO vòng kín CL):
 - Mô hình hóa hệ thống MIMO trên cơ sở SVD.
 - Thiết lập dung lượng kênh của hệ thống MIMO vòng kín (thuật toán đổ đầy nước Waterfilling).
 - Matlab hóa mô hình hệ thống và dung lượng hệ thống MIMO ngẫu nhiên vòng kín.
 - Mô phỏng dung lượng của hệ thống MIMO vòng kín trong môi trường truyền sóng pha đỉnh Rayleigh.
- Tổng hợp, phân tích, so sánh đánh giá nhận kết quả mô phỏng dung lượng kênh giữa hai hệ thống MIMO vòng hở và vòng kín.

2, Cơ sở lý thuyết

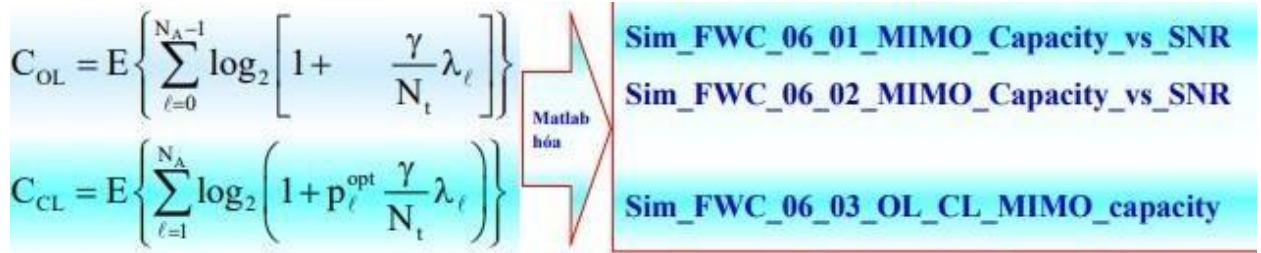
- Mô hình hệ thống SVD MIMO kết hợp OFDM



- Mô hình hóa hệ thống SVD MIMO tối ưu



3, Code matlab và kết quả mô phỏng



a, *Sim_FWC_06_01_MIMO_Capacity_vs_SNR*

```
%=====
=====
%=====   Sim_FWC_03_02_BPSK_AWGN_ChannelCode
=====
%=====
=====

% function NVD_D12VT_MIMO_Capacity_vs_SNR1

clc;
clear all,
close all;

SNR_dB      = [0:5:30];
SNR_linear  = 10.^(SNR_dB/10.);
N_iter      = 10000;
nT          = 8;
nR          = 4; % 4x4
n           = min(nT,nR);
I           = eye(n);
C           = zeros(1,length(SNR_dB));

for iter=1:N_iter
    H = sqrt(0.5)*(randn(nR,nT)+j*randn(nR,nT));
    if nR>=nT,
        HH = H'*H;
    else
        HH = H*H';
    end
    for i=1:length(SNR_dB) %random channel generation
        C(i) = C(i)+log2(real(det(I+SNR_linear(i)/nT*HH)));
    end
end
C = C/N_iter;
%=====
=====

h1 = figure(1);
set(h1,'color','c');
```

```

set(h1,'Name','Simulation for MIMO_Ergodic_Capacity_vs_SNR');
plot(SNR_dB,C,'r-s','LineWidth',[3.5]) ;hold on;
X   = xlabel('SNR [dB]');
set(X,'fontname','.Vntime','fontsize',18,'color','b');
Y   = ylabel('Ergodic Capacity vs SNR [bps/Hz]');
set(Y,'fontname','.Vntime','fontsize',18,'color','b');
T=title(strvcat(strcat('Ergodic MIMO channel capacity',...
    ' when CSI is not available at the transmitter')));
set(T,'fontname','.Vntime','fontsize',18,'color','b');
grid on;
set(gca,'fontname','.Vntime','fontsize',9);
s1='MIMO: {\it N_T_x}=4,{\it N_R_x}=4';
legend(s1);

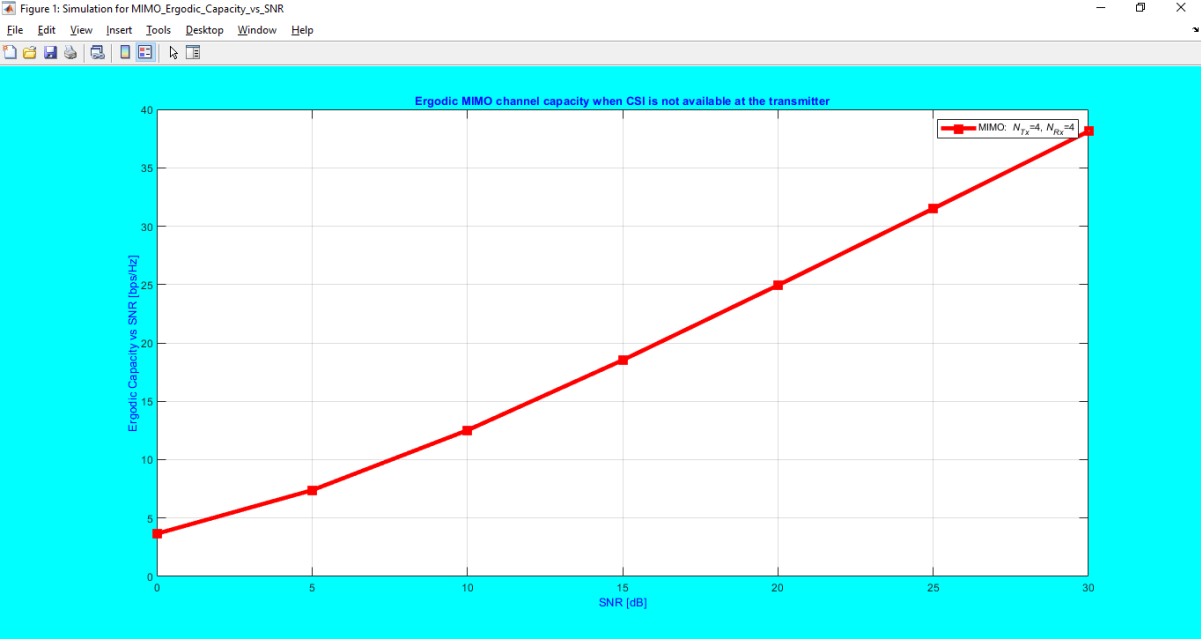
```

❖ Giải thích code

Tham số	Giải thích
SNR_dB = [0:5:30];	Một ma trận chứa các giá trị SNR theo đơn vị đo dB từ 0 đến 30 với bước nhảy là 5
SNR_linear = 10.^(SNR_dB/10.);	Một ma trận chứa các giá trị SNR biểu diễn dưới dạng đơn vị linear
N_iter = 10000;	Khởi tạo biến N_iter
nT = 8;	Số lượng anten phát là 8
nR = 4;	Số lượng anten thu là 4
n = min(nT,nR);	Lấy giá trị nhỏ nhất trong 2 giá trị nT và nR
I = eye(n);	Khởi tạo biến I với giá trị là ma trận đơn vị n x n
C = zeros(1,length(SNR_dB));	Khởi tạo biến C với giá trị là một matrix với 1 hàng và số cột bằng chiều dài mảng giá trị SNR_dB
Lệnh	Giải thích

<pre> for iter=1:N_iter H=sqrt(0.5)*(randn(nR,nT) +j*randn(nR,nT)); if nR>=nT, HH = H'*H; else HH = H*H'; end </pre>	<p>Tạo ma trận H kênh phadinh phân bố gaussian.</p> <p>Nếu $n_R \geq n_T$, tạo giá trị $HH = H$ chuyển vị nhân.</p> <p>Nếu ngược lại thì $HH=H^*H'$</p>
<pre> for i=1:length(SNR_dB) C(i) C(i)+log2(real(det(I+SNR_linear(i)/nT*HH))); end end </pre>	<p>Tính giá trị dung lượng kênh C với công thức</p>
<pre> C = C/N_iter; </pre>	<p>Tính trung bình cộng của mảng C</p>

❖ Kết quả mô phỏng:



Workspace	
Name ▲	Value
C	[3.6567,7.3820,12.513...
H	4x8 complex double
h1	1x1 Figure
HH	4x4 complex double
i	7
I	4x4 double
iter	10000
n	4
N_iter	10000
nR	4
nT	8
s1	'MIMO: {\it N_T_x}=4...
SNR_dB	[0,5,10,15,20,25,30]
SNR_linear	[1,3.1623,10,31.6228,1...
T	1x1 Text
X	1x1 Text
Y	1x1 Text

- Nhận xét:

+ Mô phỏng kênh MIMO khi không có CSI với số lượng anten phát và thu bằng nhau $T_x = R_x = 4$.

+ Với $T_x = R_x = 4$ sử dụng hệ thống MIMO SVD ngẫu nhiên giá trị dung lượng kênh C tăng mạnh khi tỉ lệ SNR tăng.

b, *Sim_FWC_06_02_MIMO_Capacity_vs_SNR*

```
%=====
=====
%===== Sim_FWC_03_02_BPSK_AWGN_ChannelCode
=====
%=====
=====

% function NVD_D12VT_MIMO_Capacity_vs_SNR2

clc;
clear all;
close all;

SNR_dB      = [0:5:30];
SNR_linear  = 10.^(SNR_dB/10.);
N_iter      = 10000;

for Icase=1:5
    if Icase==1,
        nT=4; nR=4; % 4x4
    elseif Icase==2,
        nT=2; nR=2; % 2x2
    elseif Icase==3,
        nT=1; nR=2; % 1x2
```

```

elseif Icase==4,
    nT=2; nR=1; % 2x1
else
    nT=1; nR=1; % 1x1
end

n = min(nT,nR);
I = eye(n);
C(Icase,:) = zeros(1,length(SNR_dB));

for iter=1:N_iter
    H = sqrt(0.5)*(randn(nR,nT)+j*randn(nR,nT));
    if nR>=nT,
        HH = H'*H;
    else
        HH = H*H';
    end
    for i=1:length(SNR_dB) %random channel generation
        C(Icase,i) =
C(Icase,i)+log2(real(det(I+SNR_linear(i)/nT*HH)));
    end
end
end
C = C/N_iter;
%=====
=====

h1 = figure(1);
set(h1,'color','c');
set(h1,'Name','Simulation for MIMO Ergodic Capacity vs SNR');
plot(SNR_dB,C(1,:), 'b-o', 'LineWidth',3.5); hold on;
plot(SNR_dB,C(2,:), 'r-<', 'LineWidth',3.0); hold on;
plot(SNR_dB,C(3,:), 'r-s', 'LineWidth',2.5); hold on;
plot(SNR_dB,C(4,:), 'k->', 'LineWidth',2.0); hold on;
plot(SNR_dB,C(5,:), 'g-^', 'LineWidth',1.5); hold on;

X = xlabel('SNR [dB]');
set(X,'fontname','.Vntime','fontsize',18,'color','b');
Y = ylabel('Ergodic Capacity vs SNR [bps/Hz]');
set(Y,'fontname','.Vntime','fontsize',18,'color','b');

T=title(strvcat(strcat('Ergodic MIMO channel capacity',...
' when CSI is not available at the transmitter')));
set(T,'fontname','.Vntime','fontsize',18,'color','b');
grid on;

% set(gca,'fontsize',9);
set(gca,'fontname','.Vntime','fontsize',8,'color','c');
set(gca,'fontname','.Vntime','fontsize',14);
s1='MIMO: {\it N_T_x}=4,{\it N_R_x}=4';
s2='MIMO: {\it N_T_x}=2,{\it N_R_x}=2';
s3='MIMO: {\it N_T_x}=1,{\it N_R_x}=2';
s4='MIMO: {\it N_T_x}=2,{\it N_R_x}=1';

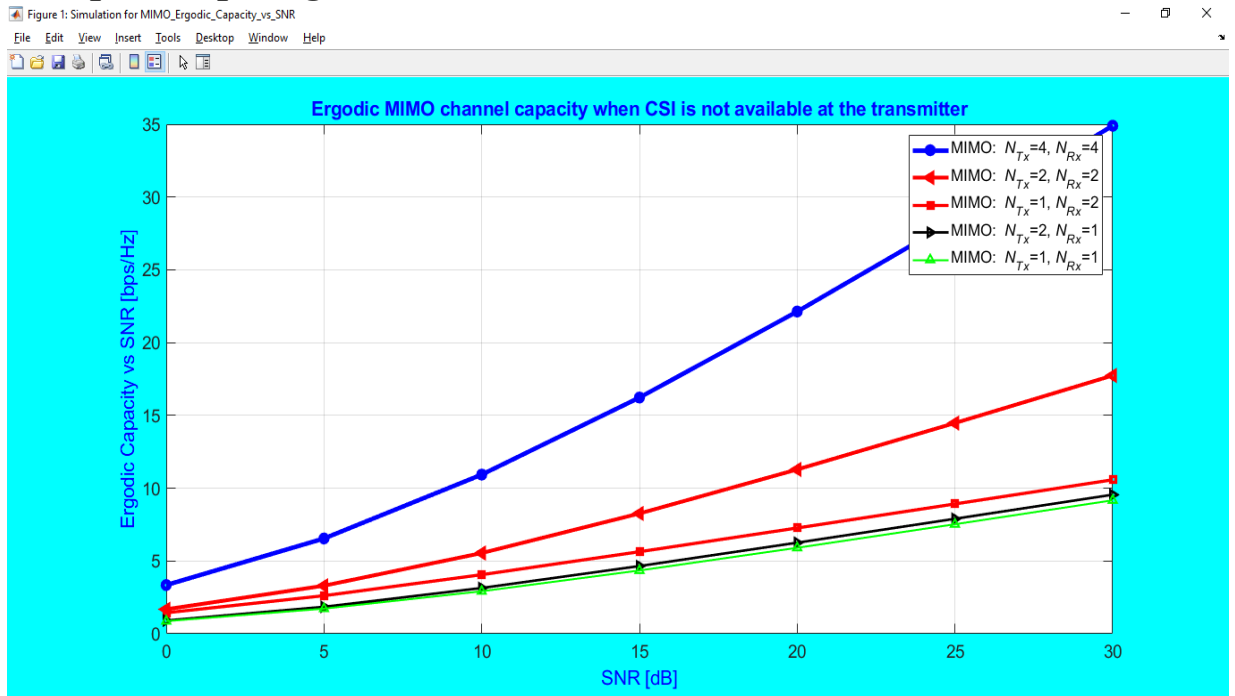
```

```
s5='MIMO: {\it N_T_x}=1,{\it N_R_x}=1';
legend(s1,s2,s3,s4,s5);
```

❖ Giải thích code

Tham số	Giải thích
SNR_dB = [0:5:30];	Tạo một mảng giá trị từ 0 tới 30 mỗi giá trị cách nhau 5 đơn vị
SNR_linear = 10.^(SNR_dB/10.);	Tạo biến SNR_linear và gán giá trị
N_iter = 10000;	Khởi tạo biến N_iter
Lệnh	Giải thích
<pre>for Icase=1:5 if Icase==1, nT=4; nR=4; % 4x4 elseif Icase==2, nT=2; nR=2; % 2x2 elseif Icase==3, nT=1; nR=2; % 1x2 elseif Icase==4, nT=2; nR=1; % 2x1 else nT=1; nR=1; % 1x1 end</pre>	<p>Icase ==1: nT và nR được đặt bằng 4 hay ma trận 4x4</p> <p>Icase ==2: nT và nR được đặt bằng 2 hay ma trận 2x2</p> <p>Icase ==3: nT=1, nR=2 hay ma trận 1x2</p> <p>Icase ==4: nT=2, nR=1 hay ma trận 2x1</p> <p>Icase bất kì có nT=nR=1 hay ma trận 1x1</p>
<pre>C(Icase,:) = zeros(1,length(SNR_dB));</pre>	Khởi tạo biến C với giá trị là một matrix với 1 hàng và số cột bằng chiều dài mảng giá trị SNR_dB

❖ Kết quả mô phỏng



Workspace		
Name ▲	Value	
C	5x7 double	
H	0.9526 - 0.1614i	
h1	1x1 Figure	
HH	0.9335	
i	7	
l	1	
lcase	5	
iter	10000	
n	1	
N_iter	10000	
nR	1	
nT	1	
s1	'MIMO: {\it N_T_x}=4...	
s2	'MIMO: {\it N_T_x}=2...	
s3	'MIMO: {\it N_T_x}=1...	
s4	'MIMO: {\it N_T_x}=2...	
s5	'MIMO: {\it N_T_x}=1...	
SNR_dB	[0,5,10,15,20,25,30]	
SNR_linear	[1,3.1623,10,31.6228,1...	
T	1x1 Text	
X	1x1 Text	
Y	1x1 Text	

Nhận xét: Mô phỏng kênh MIMO khi không có CSI với các trường hợp Tx và Rx khác nhau

- Với Tx và Rx khác nhau khi Tx > Rx (Tx = 2, Rx = 1) có dung lượng kênh nhỏ và tăng gần như tuyến tính với SNR - Với Tx và Rx khác nhau nhưng Tx < Rx (Tx = 1, Rx = 2) có dung lượng lớn hơn khi Tx > Rx.
- Với số lượng Tx và Rx bằng nhau trường hợp này khá lý tưởng khi truyền có dung lượng kênh lớn khi SNR tăng cao, càng nhiều anten phát và anten thu thì C càng tăng mạnh.

c, Sim_FWC_06_03_OL_CL_MIMO_capacity

$$\begin{aligned}
 C_{CL} &= E \left\{ \max_{\sum_{\ell=1}^{N_A} p_{\ell} = N_t} \sum_{\ell=1}^{N_A} \log_2 \left(1 + \frac{\gamma}{N_t} p_{\ell} \lambda_{\ell} \right) \right\} \\
 &= E \left\{ \sum_{\ell=1}^{N_A} \log_2 \left(1 + \frac{\gamma}{N_t} p_{\ell}^{\text{opt}} \lambda_{\ell} \right) \right\}
 \end{aligned}$$

$$\sum_{\ell=1}^{N_A} p_{\ell} = \sum_{\ell=1}^{N_A} \left(\mu - \frac{N_t}{\gamma \lambda_{\ell}} \right)^+ = N_t$$

$$p_{\ell}^{\text{opt}} = \left[\mu - \frac{N_t}{\gamma \lambda_{\ell}} \right]_+ = \begin{cases} \mu - \frac{N_t}{\gamma \lambda_{\ell}}, & \text{if } \mu > \frac{N_t}{\gamma \lambda_{\ell}} \\ 0, & \text{if } \mu < \frac{N_t}{\gamma \lambda_{\ell}} \end{cases}$$

```

%=====
=====
%===== Sim_FWC_03_02_BPSK_AWGN_ChannelCode
=====
%=====
=====
% function NVD_D12VT_CL_OL_MIMO_capacity
% The ergodic channel capacity for the open-loop system without using
CSI at the
% transmitter side, from Equation: (9.31), (9.41), (9.42)
% The ergodic channel capacity for the closed-loop (CL) system using
CSI at the
% transmitter side, from Equation: (9.44), (9.44)
clc;
clear all;
close all;
SNR_dB = [0:2:22];
SNR_linear = 10.^(SNR_dB/10.);
N_iter = 1000;
%% ..... 4x4 .....
nT = 4;
nR = 4;
n = min(nT,nR);
I = eye(n);
rho = 0.2;
sq2 = sqrt(0.5);
Rtx = [1 rho rho^2 rho^3; rho 1 rho rho^2; rho^2 rho 1 rho; rho^3
rho^2 rho 1];
rho = 0.2;
Rrx = [1 rho rho^2 rho^3; rho 1 rho rho^2; rho^2 rho 1 rho; rho^3
rho^2 rho 1];
C_44_OL = zeros(1,length(SNR_dB));
C_44_CL = zeros(1,length(SNR_dB));
for iter=1:N_iter
    Hw = sq2*(randn(4,4) + j*randn(4,4));
    H = Rrx^(1/2)*Hw*Rtx^(1/2);
    tmp = H'*H/nT;
    Lamda= svd(H'*H);
    for i=1:length(SNR_dB)
        % random channel generation
        C_44_OL(i) = C_44_OL(i) + log2(det(I+SNR_linear(i)*tmp));
    end
end
% Eq 9.41

```

```

P_opt          = FWC_Water_Filling(Lamda, SNR_linear(i), nT);
C_44_CL(i)     =
C_44_CL(i)+log2(det(I+SNR_linear(i)/nT*diag(P_opt)*diag(Lamda))); %
Eq 9.44
end
end
C_44_OL        = real(C_44_OL)/N_iter;
C_44_CL        = real(C_44_CL)/N_iter;
h1 = figure(1);
set(h1,'color','c');
set(h1,'Name','MIMO_Ergodic_Capacity_vs_SNR for closed-loop and open-
loop systems Programmed by Nguyen Viêt Dam PTIT');
plot(SNR_dB, C_44_OL, 'b-o', 'LineWidth', [3.5]);
hold on;
plot(SNR_dB, C_44_CL, 'r-<', 'LineWidth', [3.0]);
hold on;
X = xlabel('SNR [dB]');
set(X,'fontname','.Vntime','fontsize',18,'color','b');
Y = ylabel('Ergodic Capacity vs SNR [bps/Hz]');
set(Y,'fontname','.Vntime','fontsize',18,'color','b');
T=title(strvcat(strcat('Ergodic capacities for the closed-loop and
open-loop systems')));
set(T,'fontname','.Vntime','fontsize',18,'color','b');
grid on;
% set(gca,'fontsize',9);
set(gca,'fontname','.Vntime','fontsize',8,'color','c');
set(gca,'fontname','.Vntime','fontsize',12);
s1='Channel Unknown';
s2='Channel Known';
legend(s1,s2);

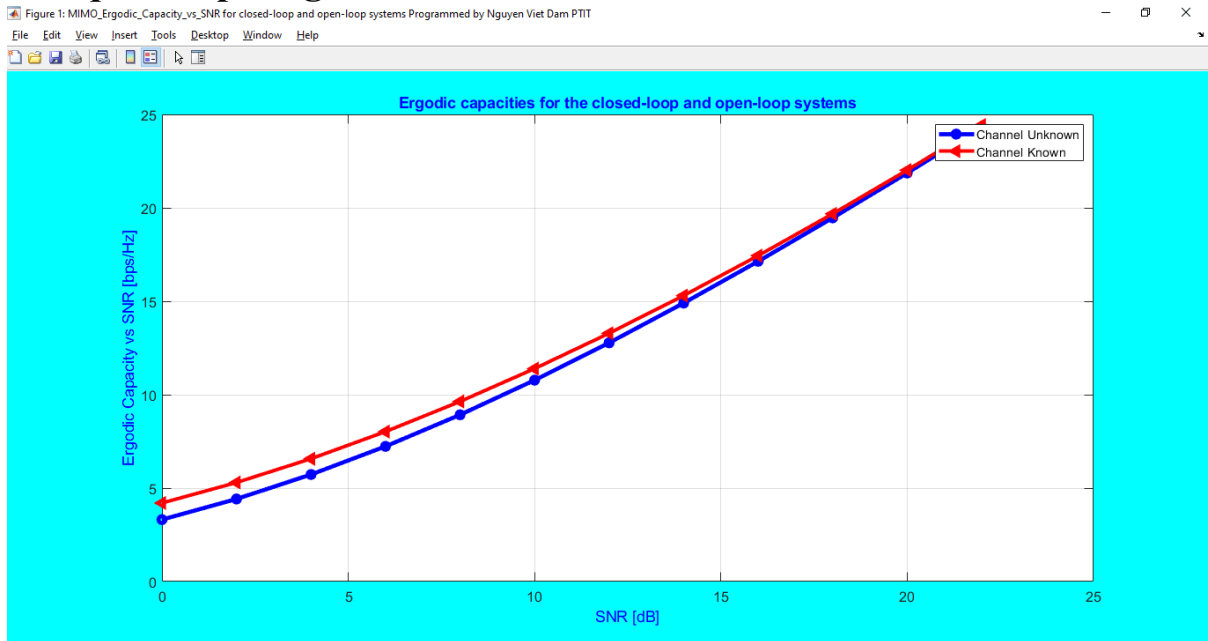
```

❖ Giải thích code

Lệnh	Giải thích
<pre> Rrx = [1 rho rho^2 rho^3; rho 1 rho rho^2; rho^2 rho 1 rho; rho^3 rho^2 rho 1]; </pre>	Tạo ma trận chéo hóa Rrx 4x4
<pre> Rtx = [1 rho rho^2 rho^3; rho 1 rho rho^2; rho^2 rho 1 rho; rho^3 rho^2 rho 1]; </pre>	Tạo ma trận chéo hóa Rtx 4x4
<pre> C_44_OL = zeros(1,length(SNR_dB)); C_44_CL = zeros(1,length(SNR_dB)); </pre>	Khởi tạo biến C_44_OL và C_44_CL có giá trị bằng ma trận 0 kích thước 1x length(SNR_d B)

<pre>P_opt = FWC_Water_Filling(Lamda,SNR_linear(i),nT);</pre>	<p>Tính giá trị P_opt dựa vào hàm FWC_Water_filling</p>
<pre>C_44_CL(i) = C_44_CL(i)+log2(det(I+SNR_linear(i)/nT*diag(P_opt)*diag(Lamda)));</pre>	<p>Tính giá trị C_44_CL thứ i dựa vào giá trị SNR thứ i và gán vào mảng C_44_CL</p>
<pre>C_44_OL = real(C_44_OL)/N_iter; C_44_CL = real(C_44_CL)/N_iter;</pre>	<p>Lấy các giá trị phần thực của mảng giá trị C_44_OL và C_44_CL</p>

❖ Kết quả mô phỏng



Workspace	
Name ▲	Value
C_44_CL	1x12 double
C_44_OL	1x12 double
H	4x4 complex double
h1	1x1 Figure
Hw	4x4 complex double
i	12
I	4x4 double
iter	1000
Lamda	[5.7022;3.9542;1.6152;...
n	4
N_iter	1000
nR	4
nT	4
P_opt	[1.0402,1.0382,1.0290,...
rho	0.2000
Rrx	4x4 double
Rtx	4x4 double
s1	'Channel Unknown'
s2	'Channel Known'
SNR_dB	1x12 double
SNR_linear	1x12 double
sq2	0.7071
T	1x1 Text
tmp	4x4 complex double
X	1x1 Text
Y	1x1 Text

Nhận xét:

- Xét kênh vô tuyến MIMO sử dụng $T_x = R_x = 4$.
- Với kênh sử dụng CSI ở phía transmitter có dung lượng kênh vòng đóng lớn hơn so với kênh không sử dụng CSI ở giá trị SNR thấp.
- Khi giá trị SNR tăng giá trị dung lượng kênh của cả 2 phương thức đều tăng mạnh và tới một giá trị nào đó sẽ có dung lượng kênh tiệm cận nhau.
- Khi sử dụng CSI cùng giải pháp Waterfilling solution ta sẽ sử dụng được băng tần hiệu quả, cấp phát công suất tối ưu cho kênh.

Sim FWC07: Mô hình hóa và mô phỏng dung lượng của hệ thống MIMO tương quan

1, Mục đích và nội dung

a, Mục đích:

Mô phỏng, phân tích ảnh hưởng của sự tương quan giữa các kênh SISO lên dung lượng hệ thống MIMO trong môi trường truyền sóng phân bố Rayleigh.

b, Nội dung:

- Tóm tắt lý thuyết: Mô hình kênh và mô hình hệ thống SVD MIMO, mô hình kênh MIMO tương quan trong môi trường truyền sóng pha đỉnh phân bố Rayleigh; lý thuyết dung lượng kênh (thiết lập công thức dung lượng kênh).
- Mô phỏng, phân tích ảnh hưởng của sự tương quan giữa các kênh SISO lên dung lượng hệ thống MIMO trong môi trường truyền sóng pha đỉnh phân bố Rayleigh:
 - Lập mô hình kênh MIMO tương quan.
 - Thiết lập và phân tích công thức dung lượng kênh MIMO không tương quan.
 - Matlab hóa mô hình hệ thống MIMO tương quan.
 - Mô phỏng dung lượng hệ thống MIMO tương quan.
- Tổng hợp, phân tích, so sánh đánh giá nhận kết quả mô phỏng dung lượng kênh giữa hai hệ thống MIMO tương quan và không tương quan.

2, Cơ sở lý thuyết

- Khái quát và mô hình kênh MIMO tương quan

❖ **Khái quát:** Các độ lợi kênh MIMO không có phân bố đồng nhất và độc lập thống kê nhau (không *i.i.d*). Tính tương quan của kênh có quan hệ mật thiết, ảnh hưởng trực tiếp lên dung lượng của kênh MIMO => xét dung lượng kênh MIMO khi các độ lợi kênh giữa các cặp anten phát/thu tương quan nhau.

❖ Mô hình kênh MIMO tương quan

$$\mathbf{H} = \mathbf{R}_r^{1/2} \mathbf{H}_w \mathbf{R}_t^{1/2}$$

- ✓ \mathbf{R}_t là ma trận tương quan, phản ánh sự tương quan giữa các anten phát (sự tương quan giữa các vector cột của \mathbf{H});
 - ✓ \mathbf{R}_r là ma trận tương quan, phản ánh sự tương quan giữa các anten thu (sự tương quan giữa các vector hàng của \mathbf{H});
 - ✓ \mathbf{H}_w là ma trận độ lợi kênh pha đỉnh Rayleigh có phân bố đồng nhất và độc lập thống kê nhau (*i.i.d*).
 - ✓ Các thực thể đường chéo \mathbf{R}_r và \mathbf{R}_t được ràng buộc bằng đơn vị.
- Dung lượng kênh tương quan

❖ Dung lượng kênh tương quan (1/2)

$$C = \log_2 \det \left(\mathbf{I}_{N_R} + \frac{E_X}{N_T N_0} \mathbf{H} \mathbf{H}^H \right); \mathbf{H} = \mathbf{R}_r^{1/2} \mathbf{H}_w \mathbf{R}_t^{1/2}$$



$$C = \log_2 \det \left(\mathbf{I}_{N_R} + \frac{E_X}{N_T N_0} \mathbf{R}_r^{1/2} \mathbf{H}_w \mathbf{R}_t \mathbf{H}_w^H \mathbf{R}_r^{H/2} \right)$$

SNR lớn, $N_T = N_R$; \Rightarrow
 \mathbf{R}_t và \mathbf{R}_r là các ma trận
 có hạng đầy đủ

$$C \approx \log_2 \det \left(\frac{E_X}{N_T N_0} \mathbf{H}_w \mathbf{H}_w^H \right) + \underbrace{\log_2 \det(\mathbf{R}_r) + \log_2 \det(\mathbf{R}_t)}_{\substack{\text{is always negative} \\ \text{capacity reduction due to the correlation} \\ \text{between the transmit and receive antennas}}}$$

❖ Dung lượng kênh MIMO tương quan (2/2)

$$C = \max_{\text{Tr}(\mathbf{R}_{xx})=N_{Tx}} \left[\log_2 \det \left(\mathbf{I}_{N_{Rx}} + \frac{E_x}{N_{Tx} N_0} \mathbf{H} \mathbf{R}_{xx} \mathbf{H}^H \right) \right]$$

$\mathbf{R}_{xx} = E\{\mathbf{X}\mathbf{X}^H\}$ is the autocorrelation of transmitted signal vector

$\text{Trace}(\mathbf{R}_{xx}) = N_{Tx}$ when the transmission power for each transmit antenna is assumed to be 1

When the SNR is *high* (e.i. $\log_2(1+x) \approx \log_2 x$), the deterministic channel capacity can be approximated as

$$C \approx \underbrace{\max_{\text{Tr}(\mathbf{R}_{xx})=N} \log_2 \det(\mathbf{R}_{xx})}_{\text{note: } \det(\mathbf{R}_{xx}) \text{ is maximized when } \mathbf{R}_{xx} = \mathbf{I}_N} + \underbrace{\log_2 \det \left(\frac{E_X}{N N_0} \mathbf{H}_w \mathbf{H}_w^H \right)}_{\text{constant}}$$

$$\begin{aligned} C &= \log_2 \det \left(\mathbf{I}_{N_R} + \frac{E_X}{N_T N_0} \mathbf{H} \mathbf{H}^H \right); \mathbf{H} = \mathbf{R}_r^{1/2} \mathbf{H}_w \mathbf{R}_t^{1/2} \\ &\downarrow \\ C &= \log_2 \det \left(\mathbf{I}_{N_R} + \frac{E_X}{N_T N_0} \mathbf{R}_r^{1/2} \mathbf{H}_w \mathbf{R}_t \mathbf{H}_w^H \mathbf{R}_r^{H/2} \right) \end{aligned} \quad \xrightarrow{\text{SNR lớn}} \quad \begin{aligned} C &\approx \log_2 \det \left(\frac{E_X}{N_T N_0} \mathbf{H}_w \mathbf{H}_w^H \right) \\ &\quad + \underbrace{\log_2 \det(\mathbf{R}_r) + \log_2 \det(\mathbf{R}_t)}_{\substack{\text{is always negative} \\ \text{capacity reduction due to the correlation} \\ \text{between the transmit and receive antennas}}} \end{aligned}$$

3, Code matlab:

```

%=====
=====
%===== Sim_FWC_03_02_BPSK_AWGN_ChannelCode
=====
%=====
=====
% function NVD_D12VT_Correlation_MIMO_Capacity
% Capacity reduction due to correlation of the MIMO channels
% In general, the MIMO channel gains are not independent and
% identically distributed (i.i.d.).
% The channel correlation is closely related to the capacity of the
% MIMO channel.
% In the sequel, we consider the capacity of the MIMO channel
% when the channel gains between transmit and received antennas are
% correlated.
% we consider the capacity for Case: the SNR is high and low.
clc;
clear all,
close all;
SNR_dB      = [0:2:20];
SNR_linear  = 10.^(SNR_dB/10);

N_iter      = 1000;
N_SNR       = length(SNR_dB);

%%-----4x4-----
nT = 4;
nR = 4;
n   = min(nT,nR);
I   = eye(n);
sq2 = sqrt(0.5);

R=[1                                0.76*exp(0.17j*pi)    0.43*exp(0.35j*pi)
0.25*exp(0.53j*pi);
    0.76*exp(-0.17j*pi)    1                                0.76*exp(0.17j*pi)
0.43*exp(0.35j*pi);
    0.43*exp(-0.35j*pi)    0.76*exp(-0.17j*pi)    1
0.76*exp(0.17j*pi);
    0.25*exp(-0.53j*pi)    0.43*exp(-0.35j*pi)    0.76*exp(-0.17j*pi)    1
];

C_44_iid      = zeros(1,N_SNR);
C_44_corr     = zeros(1,N_SNR);

for iter=1:N_iter

    H_iid      = sq2*(randn(nR,nT)+i*randn(nR,nT));
    H_corr     = H_iid*R^(1/2);
    tmp1       = H_iid'*H_iid/nT;
    tmp2       = H_corr'*H_corr/nT;

    for i=1:N_SNR                                % Eq 9.48

```



```

        C_44_iid(i)    = C_44_iid(i) + log2(det(I+SNR_linear(i)*tmp1));
        C_44_corr(i)   = C_44_corr(i) + log2(det(I+SNR_linear(i)*tmp2));
    end
end
C_44_iid    = real(C_44_iid)/N_iter;
C_44_corr   = real(C_44_corr)/N_iter;

h1 = figure(1);
set(h1,'color','c');
set(h1,'Name','Capacity reduction due to correlation of the MIMO channels');
plot(SNR_dB,C_44_iid,'b-o','LineWidth',3.5);    hold on;
plot(SNR_dB,C_44_corr,'r-v','LineWidth',5.0);  hold on;
X    = xlabel('SNR [dB]');
set(X,'fontname','.Vntime','fontsize',18,'color','b');
Y    = ylabel('Ergodic Capacity vs SNR [bps/Hz]');
set(Y,'fontname','.Vntime','fontsize',18,'color','b');
T = title('Capacity reduction due to correlation of the MIMO channels');
set(T,'fontname','.Vntime','fontsize',18,'color','b');
grid on;
set(gca,'fontname','.Vntime','fontsize',14);
    s1='iid 4x4 channels';
    s2='correlated 4x4 channels';
legend(s1,s2);

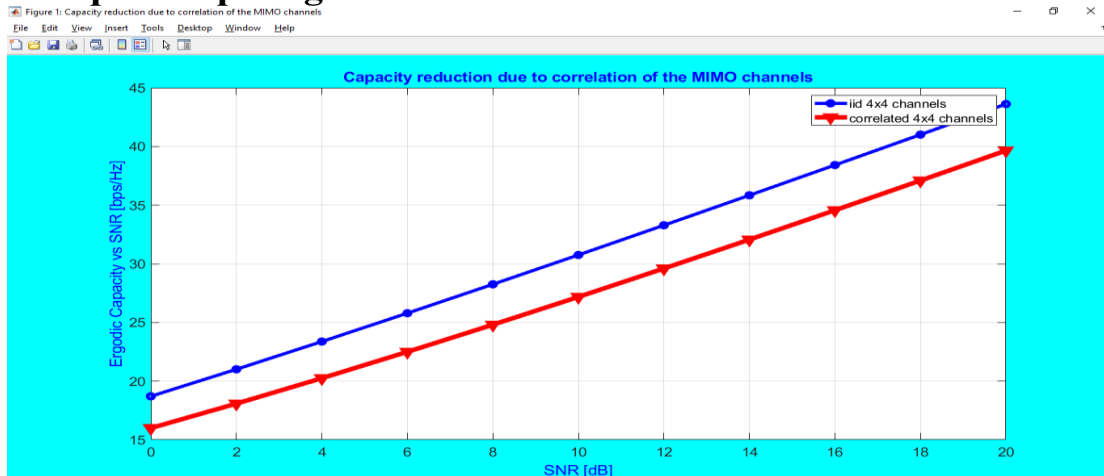
```

























❖ Giải thích code

Tham số	Giải thích
SNR_dB = [0:2:20];	Tạo một mảng giá trị từ 0 tới 20 mỗi giá trị cách nhau 2 đơn vị
SNR_linear = 10.^(SNR_dB/10);	Tạo biến SNR_linear và gán giá trị
N_iter = 1000;	Khởi tạo biến N_iter
N_SNR = length(SNR_dB);	Số lượng giá trị SNR
nT = 4;	Số lượng anten phát là 4
nR = 4;	Số lượng anten thu là 4
n = min(nT,nR);	Lấy giá trị nhỏ nhất trong 2 giá trị nT và nR
I = eye(n);	Khởi tạo biến I với giá trị là ma trận đơn vị n x n
sq2 = sqrt(0.5);	Giá trị căn bậc hai của 0.5, thường được sử dụng trong tính toán để đại diện cho căn bậc hai của năng lượng
R=[1 0.76*exp(0.17j*pi) 0.43*exp(0.35j*pi) 0.25*exp(0.53j*pi);	Khởi tạo ma trận R là ma trận tương quan cho kênh tương quan.

<pre> 0.76*exp(-0.17j*pi) 1 0.76*exp(0.17j*pi) 0.43*exp(0.35j*pi); 0.43*exp(-0.35j*pi) 0.76*exp(-0.17j*pi) 1 0.76*exp(0.17j*pi); 0.25*exp(-0.53j*pi) 0.43*exp(-0.35j*pi) 0.76*exp(- 0.17j*pi) 1]; </pre>	<p>Các giá trị trong ma trận này có thể là các tham số kênh thực tế.</p>
<pre> C_44_iid = zeros(1,N_SNR); C_44_corr = zeros(1,N_SNR); </pre>	<p>Hai mảng để lưu trữ thông lượng ước tính cho trường hợp độc lập và trường hợp tương quan.</p>
<pre> for iter=1:N_iter H_iid = sqrt(2)*(randn(nR,nT)+i*randn(nR,nT)); H_corr = H_iid*R^(1/2); tmp1 = H_iid'*H_iid/nT; tmp2 = H_corr'*H_corr/nT; </pre>	<p>Vòng lặp chạy qua mỗi lần mô phỏng để tính thông lượng ước tính cho từng giá trị SNR. Trong mỗi lần lặp, tạo ma trận kênh tương ứng bằng cách sử dụng mô phỏng nhiễu Gauss. tmp1 và tmp2 tính ma trận tương quan của các tín hiệu đầu ra ứng với hai trường hợp.</p>
<pre> for i=1:N_SNR C_44_iid(i) = C_44_iid(i) + log2(det(I+SNR_linear(i)*tmp1)); C_44_corr(i) = C_44_corr(i) + log2(det(I+SNR_linear(i)*tmp2)); end end </pre>	<p>Vòng lặp chạy qua các giá trị SNR và tính thông lượng ước tính dựa trên các ma trận tương quan đã tính.</p>
<pre> C_44_iid = real(C_44_iid)/N_iter; C_44_corr = real(C_44_corr)/N_iter; </pre>	<p>Thông lượng ước tính được tính trung bình qua các lần mô phỏng.</p>

❖ Kết quả mô phỏng:



Workspace		
Name ▲	Value	
 C_44_corr	1x11 double	
 C_44_iid	1x11 double	
 h1	1x1 Figure	
 H_corr	4x4 complex double	
 H_iid	4x4 double	
 i	11	
 I	4x4 double	
 iter	1000	
 n	4	
 N_iter	1000	
 N_SNR	11	
 nR	4	
 nT	4	
 R	4x4 complex double	
 s1	'iid 4x4 channels'	
 s2	'correlated 4x4 chann...	
 SNR_dB	1x11 double	
 SNR_linear	1x11 double	
 sq2	0.7071	
 T	1x1 Text	1x11 double
 tmp1	4x4 double	
 tmp2	4x4 complex double	
 X	1x1 Text	
 Y	1x1 Text	