CL Logbook

15/10/2021

MIMO $\quad n_t \times n_r$

$\underline{2 \times 2}$

$$\Rightarrow \quad y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\vec{y} = \sqrt{\frac{\rho}{n_T}}\, H \vec{x} + \vec{n}$$

$$H = \begin{bmatrix} \text{randn} + i \cdot \text{randn} & \text{randn} + i \cdot \text{randn} \\ \text{randn} + i \cdot \text{randn} & \text{randn} + i \cdot \text{randn} \end{bmatrix}$$

2×2

Gaussian

Unit variance

$$n = \begin{bmatrix} \text{randn} + i \cdot \text{randn} \\ \text{randn} + i \cdot \text{randn} \end{bmatrix}$$

2×1

Gaussian

ex 1 Capacity

$$C_H = \log_2 \left| I + \frac{\rho}{n_T} H H^H \right|$$

Conjugate transpose

$$\begin{bmatrix} h_1 & h_3 \\ h_2 & h_4 \end{bmatrix} \rightarrow \begin{bmatrix} h_1^* & h_2^* \\ h_3^* & h_4^* \end{bmatrix}$$

in Matlab : H'

$\rho = 0 - 30\,dB$

take 31 samples : $SNR_dB = [0, 1, 2, \dots, 30]$

$$SNR = 10^{\frac{SNR_dB}{10}} = [1, \dots, 1000]$$

16/10/2021

To get Cergodic → take 10000 random H → each H get a C_H (10000 C_H)

Cergodic ← mean

First try to write code in Matlab and plot a graph for $n_r = n_t = 1, 2, 3, 4$ to compare with the provided graph

Let C_H = zeros(10000, p)

each row stores a C_H

→ samples (1, 2, ..., 31)

0 dB 1 dB 30 dB

I = eye(nr(j)) Build the identical matrix

j = [1, 2, 3, 4]

$n_r=1$ $n_r=2$ $n_r=4$

10000 loops

H(m, n) = (rand n + 1i * rand n)

$n_r = n_t = 2$

0~30 dB

C_H(l, k) = $\log_2$(det(I + SNR(k)/n_t(j) * (H * H'))))

10000 loops k = 1 : p

C_ergodic(j, :) = mean(C_H)

Inorder to plot the figure without warning,

take abs (C_ergodic)

its imaginary part is about $1.0e^{-16}$ degree, ignorable

Then plot

the result is

1.0e-16 *

列 1 至 9

0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i
0.0000 + 0.0069i	0.0000 - 0.0040i	0.0000 - 0.0057i	0.0000 + 0.0028i	0.0000 + 0.0096i
0.0000 - 0.0011i	0.0000 + 0.0053i	0.0000 + 0.0079i	0.0000 - 0.0096i	0.0000 + 0.0084i
0.0000 + 0.0022i	0.0000 + 0.0042i	0.0000 + 0.0059i	0.0000 - 0.0045i	0.0000 - 0.0002i

列 10 至 18

0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i
0.0000 - 0.0000i	0.0000 + 0.0117i	0.0000 - 0.0257i	0.0000 - 0.0009i	0.0000 + 0.0115i
0.0000 - 0.0079i	0.0000 - 0.0326i	0.0000 + 0.0045i	0.0000 - 0.0015i	0.0000 - 0.0557i
0.0000 + 0.0182i	0.0000 + 0.0129i	0.0000 + 0.0048i	0.0000 + 0.0547i	0.0000 - 0.0195i

列 19 至 27

These two figures are about the same.

But, there is a problem

It is clear that, my result is a bit larger than the provided one.

Why?

17/10/2021

I forget to normalize H, it does not has unit variance

$$\cancel{H = \begin{bmatrix} randn + i\cdot randn & randn + i\cdot randn \\ randn + i\cdot randn & randn + i\cdot randn \end{bmatrix}}$$

$$\Downarrow$$

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} \sim & \sim \\ \sim & \sim \end{bmatrix}$$

No ise should also has unit variance in ex 2 & 3 !!!

∴ The right figure is:

They are the same.

good, the same

∴ Let $n_r = n_t = 5, 6, 7, 8, 9, 10$

$j = [1, 2, 3, 4, 5, 6]$

Result:

Capacity increase linearly with nr & nt & SNR

18/10/2021

ex 2 Transmission and Reception

$$\vec{y} = \sqrt{\frac{P}{n_T}}\, H \vec{x} + \vec{n}$$

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} \text{randn} + i \cdot \text{randn} & \text{randn} + i \cdot \text{randn} \\ \text{randn} + i \cdot \text{randn} & \text{randn} + i \cdot \text{randn} \end{bmatrix}$$

2x2
Gaussian
unit variance

$$n = \frac{1}{\sqrt{2}} \begin{bmatrix} \text{randn} + i \cdot \text{randn} \\ \text{randn} + i \cdot \text{randn} \end{bmatrix}$$

2x1
Gaussian

$$y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

After getting y, detection

① ML

$$\hat{x} = \arg\min_{x \in C} \left\| y - \sqrt{\frac{P}{n_T}}\, H x \right\|^2$$

QPSK Modulation
with Gray mapping

e.g: a set of bits <u>00011011</u>

00
01
10
11

Gray mapping

QPSK $\Longrightarrow$

$\because$ 2x2 MIMO, each time transmit 2 symbols

$$\therefore \quad x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} \pm 1 \pm i \\ \pm 1 \pm i \end{bmatrix}$$

$2 \times 2 \times 2 \times 2 = 16$ combinations

00	00	00	00	01	01	01	01	10	10	10	10	11	11	11	11
00	01	10	11	00	01	10	11	00	01	10	11	00	01	10	11

$\therefore$ After getting y

Find $y - \sqrt{\frac{\rho}{n_T}} H x \Rightarrow$ 16 values find the minimum one

$\Downarrow$

The corresponding x is regarded as the original two symbols

19/10/2021

Write code in Matlab

$x_00 = \frac{1}{\sqrt{2}}(-1-1i)$ $\quad$ $x_10 = \frac{1}{\sqrt{2}}(1-1i)$

$x_01 = \frac{1}{\sqrt{2}}(-1+1i)$ $\quad$ $x_11 = \frac{1}{\sqrt{2}}(1+1i)$

$$x_gray = \underbrace{\begin{bmatrix} x_00 & x_00 & \cdots & x_11 \\ x_00 & x_01 & \cdots & x_11 \end{bmatrix}}_{16 \text{ columns}}$$

$$C = \frac{1}{\sqrt{2}}\begin{bmatrix} \pm 1 \pm 1i \\ \pm 1 \pm 1i \end{bmatrix} \rightarrow \frac{1{,}000{,}000}{4} \text{ columns}$$

It is suggested to use 10^6 bits

Each column of C contain 2 symbols // 4 bits

$$C_all = \frac{1}{\sqrt{2}}\left(\text{randsrc}\left(n, \frac{b}{4}\right) + 1i * \text{randsrc}\left(n, \frac{b}{4}\right)\right)$$

for L=1: Loop 300 loops → plot figure use average of (300 SER)

$$H = \frac{1}{\sqrt{2}} * \left(\text{randn}(n,n) + 1i * \text{rand}(n,n)\right)$$

Generate a H, carries 10^6 bits

for k=1: p : Because the program is slow, let p=8 to tak 8 samples

for $o = 1 : \frac{b}{4}$ transmission for each row of C_all

$A = \sqrt{\frac{\rho}{n}}$ $\quad$ $y = AHc + n$

Then calculate norm $(y - AH\,x_gray)$ → one of 16 columns

find the min(16 values)

the received two symbol is the corresponding column of x_gray

check: if symbol $1 \neq c(1)$

error = error + 1

if symbol2 ≠ C(2)

error = error + 1

$$\therefore BER_ML(L, k) = \frac{error}{b/2} \times \frac{1}{2}$$

symbol errors

symbols

symbol error rate

Bit Error Rate

BER_ML_mean = mean(BER_ML, 1)

mean of rows

Run

20/10/2021

Result:

300 loops

(The original figure is deleted)

From the figure, only about 1 bit error at 25dB, while more than 10% of bits are wrong when SNR < about 5dB

Then add ZF & MMSE to the program

R_ZF = (AH)\y $\quad (A = \sqrt{\frac{\rho}{n_t}})$

$$R_MMSE = \left(A^2(H'H) + I\right) \backslash (AH'y)$$

Here $R_ZF = \begin{bmatrix} a + bi \\ c + di \end{bmatrix}$ compare with $c\begin{bmatrix} e + fi \\ g + hi \end{bmatrix}$

If $\text{sign}(a) \neq \text{sign}(e)$ or $\text{sign}(b) \neq \text{sign}(f)$

error = error + 1

If $\text{sign}(c) \neq \text{sign}(g)$ or $\text{sign}(d) \neq \text{sign}(h)$

error = error + 1

The received symbol locates at the 1st/2nd/3rd/4th quadrant

⇓

consider it as the corresponding X_gray

⇓

If it does not equals to the original symbol

error = error + 1

It is the same for R_MMSE

Result

10% wrong bits

only 1 bit is wrong

Only runs a few loops so there are some problems at the beginning

But the trend is right

performance of ML > ZF > MMSE

But I find a problem, will the noise change with SNR?

Because signal power is fixed, noise power should change to keep SNR

Hence I tried to change the power of noise

$$n = \frac{1}{\sqrt{2}}\begin{bmatrix} x \\ x \end{bmatrix} \Rightarrow \sqrt{\frac{1}{2SNR}}\begin{bmatrix} x \\ x \end{bmatrix}$$

Result for a few loops (everything is the same except noise power)

decrease too fast

It does not make sense

Wrong!

After checking, I find that

this noise is the noise in wireless channel, it is irrelevant to SNR

this SNR is the SNR at the receive antenna

ex3 Space-Time Coding

Since ex3 can be implemented by changing ex2 a bit, I decided to put them together

21/10/2021

$$X = \begin{bmatrix} x_1 & -x_2^* \\ x_2 & x_1^* \end{bmatrix} \qquad N = \begin{bmatrix} a+bi & e+fi \\ c+di & g+hi \end{bmatrix}$$

$$y = \begin{bmatrix} X & X \\ X & X \end{bmatrix}$$ others are the same

After getting y

Calculate $\left(\text{norm}(y - A\,H\,x, \text{'fro'})\right)^2$

frobenius norm

e.g. $x_gray = \begin{bmatrix} x_{01} \\ x_{11} \end{bmatrix}$, here $X = \begin{bmatrix} x_{01} & -x_{11}^* \\ x_{11} & x_{01}^* \end{bmatrix}$

find the minimum

and find the corresponding x_gray

check: if symbol 1 $\neq$ C(1)

error = error + 1

if symbol 2 $\neq$ C(2)

error = error +1

The calculate the BER

Final Result

Good result.

1. BER give by ZF is constantly the largest one.
2. At the beginning, ML and MMSE show similar performance. But as SNR ↑, ML performs much better.
3. MMSE constantly lower than ZF, but the gap between them is not very large.

4. ML almost do not have error after SNR > 26 dB
MMSE and ZF still show a relatively large
BER at 35 dB

5. If the transmit symbols are coded, the performance
is significantly better than uncoded symbols.

CL 1

```
%% Initialization
clc;
clear;
close all;

%% Set basic parameters
nr = [5 6 7 8 9 10]; % number of receive antennas
nt = nr; % number of transmit antennas

p = 31; % take 31 samples
SNR_dB = (0:30/(p-1):30); % SNR_dB = 0, 0.3, ... , 29.7, 30
SNR = 10.^(SNR_dB/10); % calculate SNR, SNR_dB = 10log10(SNR),
SNR_dB = 1, ... , 1000

C_H = zeros(10000,p); % initialize capacity C_H under certain
nr with 10000 rows, each row is calculated from a random H,
with length p
C_ergodic = zeros(length(nr),p); % initialize mean capacity
C_ergodic, each row is the mean of C_H under certain nr, with
length p

% Avoid the pseudorandomness
rand('twister',mod(floor(now*8640000),2^31-1))

%% Calculation
for j = 1:length(nr) % take different nr = [5 6 7 8 9 10]

    I = eye(nr(j)); % build nr*nr identical matrix
    H = zeros(nr(j)); % initialize channel H (nr*nr)

    for L = 1:10000 % use i.i.d. random H to calculate C_H
        for m = 1:nr(j)
            for n = 1:nr(j)
                % find elements of H, H is a complex Gaussian
random matrix with i.i.d. entries
                H(m,n) = (randn+1i*randn)/sqrt(2); % divided
by sqrt(2) to keep variance = 1
            end
        end
        H_h = H'; % H_h is the conjugate transpose of H

        for k = 1:p % calculate C_H with H, H_h, and p values
of SNR from 1-1000 (0-30dB)
            C_H(L,k) = log2(det(I+SNR(k)/nt(j).*(H*H_h)));
        end
    end
```

```
    C_ergodic(j,:) = mean(C_H); % C_ergodic is the average of
10000 rows of C_H
end

C_ergodic_abs = abs(C_ergodic); % take absolute value to plot
the figure (the imaginary part is about 1.0e-16 degree,
ignorable)

%% Plot the figure
set(gcf,'Position',[10,10,700,500]); % Set the figure size
set(gca,'fontname','Times New Roman'); % Set the font type

a = plot(SNR_dB,C_ergodic_abs(1,:),'LineWidth',1.5); % Plot
the figure
hold on
b = plot(SNR_dB,C_ergodic_abs(2,:),'LineWidth',1.5);
c = plot(SNR_dB,C_ergodic_abs(3,:),'LineWidth',1.5);
d = plot(SNR_dB,C_ergodic_abs(4,:),'LineWidth',1.5);
e = plot(SNR_dB,C_ergodic_abs(5,:),'LineWidth',1.5);
f = plot(SNR_dB,C_ergodic_abs(6,:),'LineWidth',1.5);

a.Color = [1 0 0]; % Set the colour of lines
b.Color = [0 0.9 1];
c.Color = [1 0 1];
d.Color = [1 0.7 0];
e.Color = [0 0.8 0];
f.Color = [0 0 0.8];

grid on
legend('nr = 5','nr = 6','nr = 7','nr = 8','nr = 9','nr =
10','FontSize',11); % Set the legend
xlabel('SNR (dB)','FontSize',13); % Set the axes
ylabel('Capacity (bps/Hz)','FontSize',13);
title('MIMO Capacity','FontSize',15); % Set the title
text(22, 15, sprintf('Yuxiang
Zheng\n17/10/2021'),'FontSize',12);
```

CL 2&3

```
%% Initialization
clc;
clear;
```

```
close all;

%% Set basic parameters
n = 2; % number of receive antennas
p = 8; % take 8 samples
L = 0;
Loop = 1000;
b = 10^6; % transmit 10^6 bits

I = eye(n,n);

SNR_dB = (0:35/(p-1):35); % SNR_dB = 0, 0.35, ... , 34.65, 35
SNR = 10.^(SNR_dB/10); % calculate SNR, SNR_dB = 10log10(SNR),
SNR_dB = 1, ... , 3162.3

N_L = zeros(n,b/2);
ML = zeros(16,0);
ML_coded = zeros(16,0);

BER_ML = zeros(Loop,p);
BER_ZF = zeros(Loop,p);
BER_MMSE = zeros(Loop,p);
BER_ML_coded = zeros(Loop,p);

mean_BER_ML = zeros(1,p);
mean_BER_ZF = zeros(1,p);
mean_BER_MMSE = zeros(1,p);
mean_BER_ML_coded = zeros(1,p);

%% Gray Mapping

x_00 = sqrt(0.5)*(-1-1i); % four types of symbols
x_01 = sqrt(0.5)*(-1+1i);
x_10 = sqrt(0.5)*(+1-1i);
x_11 = sqrt(0.5)*(+1+1i);

x_gray = [[x_00 x_00 x_00 x_00 x_01 x_01 x_01 x_01 x_10 x_10
x_10 x_10 x_11 x_11 x_11 x_11]; % 16 kinds of symbol
combinations
          [x_00 x_01 x_10 x_11 x_00 x_01 x_10 x_11 x_00 x_01
x_10 x_11 x_00 x_01 x_10 x_11]];
x_gray_coded = zeros(n,n,16);
for q = 1:16
    x_gray_coded(:,1,q) = x_gray(:,q);
    x_gray_coded(:,2,q) = [-
conj(x_gray(2,q));conj(x_gray(1,q))]; % 16 kinds of coded
symbol combinations
end
```

```
c_all =  sqrt(0.5)*(randsrc(n,b/4)+1i*randsrc(n,b/4)); %
generate b/2 symbols


%% Plot the figure
set(gcf,'Position',[10,10,700,500]); % Set the figure size
set(gca,'fontname','Times New Roman'); % Set the font type

f_1 = semilogy(SNR_dB,mean_BER_ML,'LineWidth',1.5); % Plot the
figure
axis([min(SNR_dB) max(SNR_dB) 1/b 1])
hold on
f_2 = semilogy(SNR_dB,mean_BER_ZF,'LineWidth',1.5);
f_3 = semilogy(SNR_dB,mean_BER_MMSE,'LineWidth',1.5);
f_4 = semilogy(SNR_dB,mean_BER_ML_coded,'LineWidth',1.5); %
Plot the figure

f_1.Color = [1 0 0]; % Set the colour of lines
f_2.Color = [0 0.9 1];
f_3.Color = [1 0 1];
f_4.Color = [0 0.8 0];

grid on
legend('BER ML','BER ZF','BER MMSE','BER ML
coded','FontSize',11); % Set the legend
xlabel('SNR (dB)','FontSize',13); % Set the axes
ylabel('BER','FontSize',13);
title({'BER vs SNR',['Loops : ',num2str(L)]},'FontSize',15); %
Set the title
text(2, 0.5*10^-5, sprintf('Yuxiang
Zheng\n21/10/2021'),'FontSize',12);
drawnow

f_1.YDataSource = 'mean_BER_ML';
f_2.YDataSource = 'mean_BER_ZF';
f_3.YDataSource = 'mean_BER_MMSE';
f_4.YDataSource = 'mean_BER_ML_coded';

%% Main Loop
for L = 1:Loop

    H = sqrt(0.5)*(randn(n,n)+1i*randn(n,n)); % for each loop,
generate a H
    N_L = sqrt(0.5)*(randn(n,b/2)+1i*randn(n,b/2)); % 16 for
each loop, generate a set of noise

    for k = 1:p
        E_ML = 0;
        E_ZF = 0;
```

```
        E_MMSE = 0;
        E_ML_coded = 0;

        %% Transmission & detection
        for o = 1:b/4
            c = c_all(:,o); % take one column of the symbol
set
            c_coded(:,1) = c;
            c_coded(:,2) = [-conj(c(2));conj(c(1))]; %
generate the coded c

            N = N_L(:,(2*o-1)); % take one column of the noise
set
            N_coded = N_L(:,(2*o-1):2*o); % take two columns
of the noise set for coded c

            A = sqrt(SNR(k)/n);
            y = A*H*c + N; % calculate received y
            y_coded = A*H*c_coded + N_coded; % calculate
received & coded y

            for q = 1:16
                ML(q) = norm(y-A*H*x_gray(:,q));
            end
            [~,Col]=min(ML); % ML detection
            R_ML = x_gray(:,Col);

            R_ZF = (A*H)\y; % Zero forcing
            R_MMSE = (A^2*(H'*H)+I)\(A*H'*y); %  minimum mean-
square error detection

            for q = 1:16
                ML_coded(q) = (norm(y_coded-
A*H*x_gray_coded(:,:,q), 'fro'))^2;
            end
            [~,Col]=min(ML_coded); % ML detection for coded
signal
            R_ML_coded = x_gray(:,Col);

            %% Check the correctness
            if (sign(real(R_ML(1,1))) ~= sign(real(c(1,1))))
|| (sign(imag(R_ML(1,1))) ~= sign(imag(c(1,1))))
                E_ML = E_ML + 1;
            end
            if (sign(real(R_ML(2,1))) ~= sign(real(c(2,1))))
|| (sign(imag(R_ML(2,1))) ~= sign(imag(c(2,1))))
                E_ML = E_ML + 1;
            end
```

```
            if (sign(real(R_ZF(1,1))) ~= sign(real(c(1,1))))
|| (sign(imag(R_ZF(1,1))) ~= sign(imag(c(1,1))))
                E_ZF = E_ZF + 1;
            end
            if (sign(real(R_ZF(2,1))) ~= sign(real(c(2,1))))
|| (sign(imag(R_ZF(2,1))) ~= sign(imag(c(2,1))))
                E_ZF = E_ZF + 1;
            end

            if (sign(real(R_MMSE(1,1))) ~= sign(real(c(1,1))))
|| (sign(imag(R_MMSE(1,1))) ~= sign(imag(c(1,1))))
                E_MMSE = E_MMSE + 1;
            end
            if (sign(real(R_MMSE(2,1))) ~= sign(real(c(2,1))))
|| (sign(imag(R_MMSE(2,1))) ~= sign(imag(c(2,1))))
                E_MMSE = E_MMSE + 1;
            end

            if (sign(real(R_ML_coded(1,1))) ~=
sign(real(c(1,1)))) || (sign(imag(R_ML_coded(1,1))) ~=
sign(imag(c(1,1))))
                E_ML_coded = E_ML_coded + 1;
            end
            if (sign(real(R_ML_coded(2,1))) ~=
sign(real(c(2,1)))) || (sign(imag(R_ML_coded(2,1))) ~=
sign(imag(c(2,1))))
                E_ML_coded = E_ML_coded + 1;
            end
        end

        % Calculate BER
        BER_ML(L,k) = E_ML/b;
        BER_ZF(L,k) = E_ZF/b;
        BER_MMSE(L,k) = E_MMSE/b;
        BER_ML_coded(L,k) = E_ML_coded/b;
    end

    %% Calculate average BER
    mean_BER_ML = mean(BER_ML(1:L,:),1);
    mean_BER_ZF = mean(BER_ZF(1:L,:),1);
    mean_BER_MMSE = mean(BER_MMSE(1:L,:),1);
    mean_BER_ML_coded = mean(BER_ML_coded(1:L,:),1);

    title({'BER vs SNR',['Loops :
',num2str(L)]},'FontSize',15); % Set the title

    refreshdata
    drawnow
end
```