通訊實驗

實驗七 第五組

電機112 林珮玉 E24084096
航太112 楊秉融 F44086181

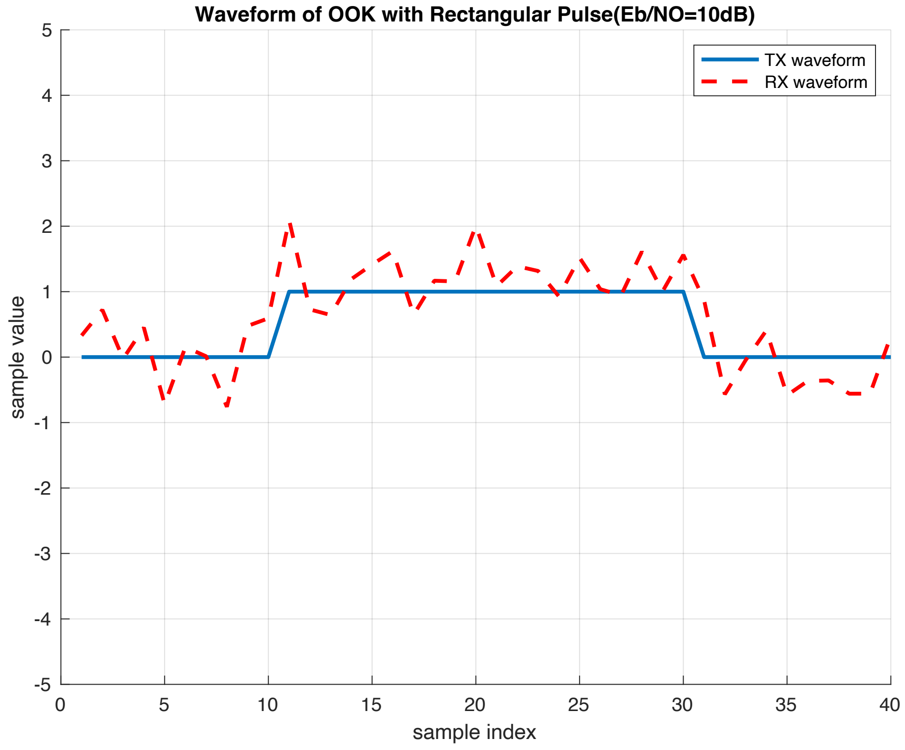
Q1

1. 修改範例程式，畫出 transmitted waveform 和 received waveform

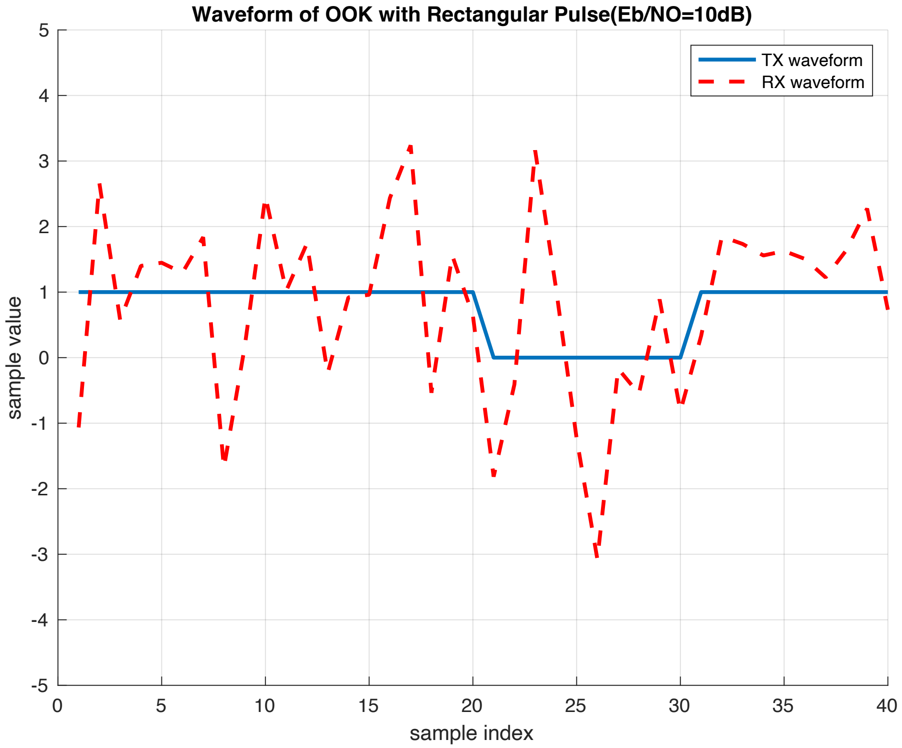
(duration = 4 symbol periods，$\frac{E_b}{N_0} = 10$ dB)，並比較之。

再畫 $\frac{E_b}{N_0} = 3$ dB 的圖，說明與 $\frac{E_b}{N_0} = 10$ dB 時之差異。

$$\frac{E_b}{N_0} = 10 \text{ dB}$$



$$\frac{E_b}{N_0} = 3 \text{ dB}$$



Q1 explanation

定義信噪比 $SNR_0 = \frac{P_{signal}}{P_{noise}} = \frac{E_b}{N_0}$，其中$N_0$代表高斯雜訊的一個定值 $(S_w(f) = \frac{N_0}{2}$ 、E_b代表on-off keying每個bit的平均能量

$$E_b = \frac{1}{2}A^2T + \frac{1}{2} * 0 = \frac{A^2T}{2}$$

因此降低SNR，N_0保持一致，等同降低降低每個bit傳送時的平均能量，由於$P_e = Q(\sqrt{SNR})$，因此BER會變差。

- 而範例程式這邊與我上述直覺**理解不同點**是在維持E_b下，調動noise的power σ^2，降低SNR，會讓 σ^2變大，因此接收端(RX)訊號右圖會比左圖變化幅度還大。

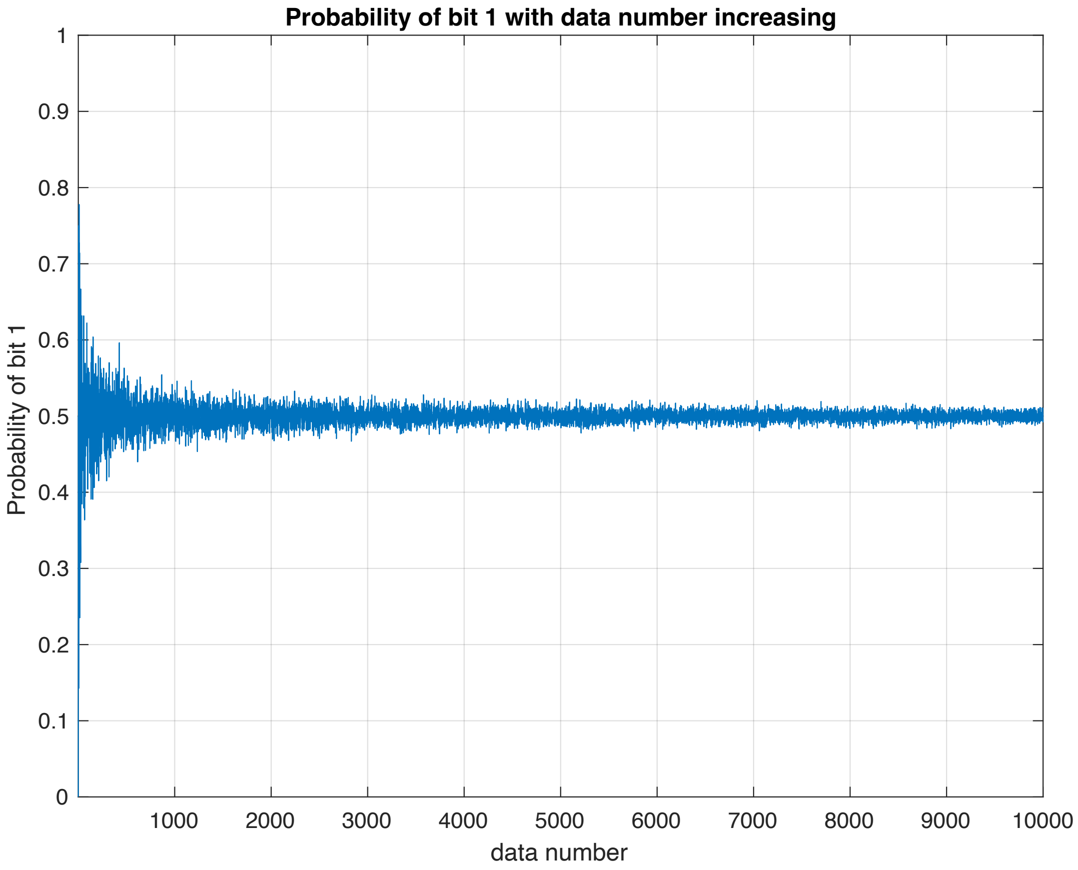
Q2a

2. 理解範例程式，並回答下列問題。

a. 解釋為何（p. 16）

```
Data_bit=(rand(1,data_number) > 0.5 );
```

可產生 random bits?
此行所產生之 bit 0 和 bit 1，其發生機率理論上應為何？（說明之。）
實際於模擬中 bit 0 和 bit 1 發生的機率為何？是否符合理論預測？



指令rand(M,N)代表返回一個 $M\times N$ 的矩陣，每個element的值是取區間0到1的均勻分布並使用logic判斷式判斷是否大於0.5，若符合回傳1，反之回傳0，回傳的data type是logic。由於是均勻分布，0到0.5的區間機率是$\frac{1}{2}$，判定為logic 0；0.5到1的區間機率是$\frac{1}{2}$，判定為logic 1，實際模擬如左圖。

藉由做很多次的隨機試驗，可以得到relative frequency $f_n(A)=\frac{N_n(A)}{n}$，當試驗次數越多，會趨於機率，記為 $\lim_{n\to\infty} f_n(A)=P(A)$，由上圖可知data number越多，機率會趨於$\frac{1}{2}$。

b. 說明（p. 16）

Q2b

Data_pulse=reshape(Data_pulse_array,1,length(p1)*data_number);

這行程式碼的作用為何？

```
data_number = 4; % # of bits
Fs=10; % sampling frequency (used to generate received samples)
Data_bit = (rand(1,data_number) > 0.5 ); % random bits
p1 = ones(1,Fs); % discrete-time rectangular pulse that represents one symbol
Data_pulse_array = (p1')*Data_bit;
Data_pulse = reshape(Data_pulse_array,1,length(p1)*data_number);
```

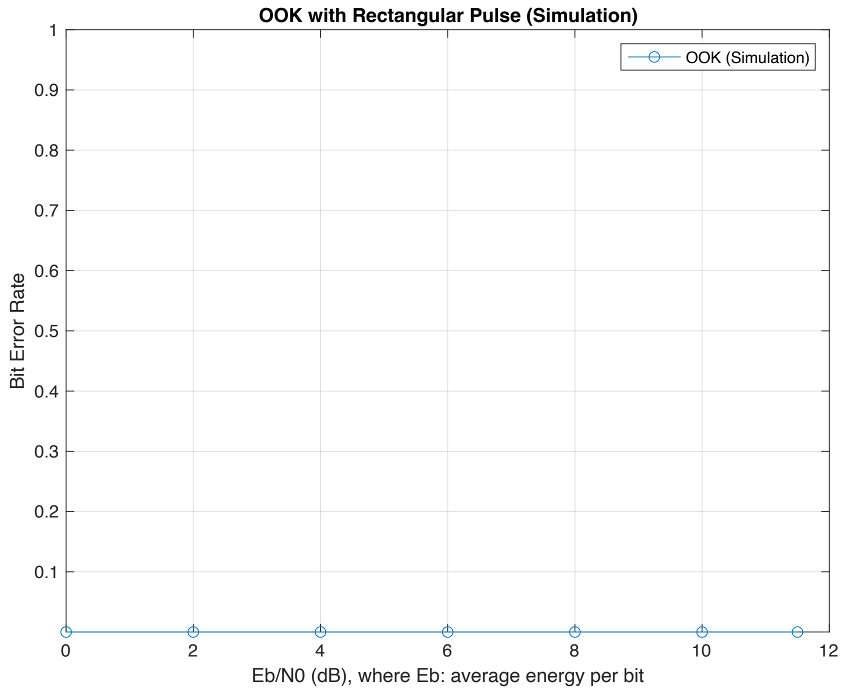
Data_bit為由機率各半$\frac{1}{2}$的logic 0和1形成1×1051×105的矩陣，Data_pulse_array為將Data_bit的值拷貝成10×10^5的矩陣，此時data type是double，最後reshape將10×10^5的矩陣，以column為順序，轉換為(1,length(p1)*data_number)的矩陣，代表意義是每10個單位傳送一個symbol。

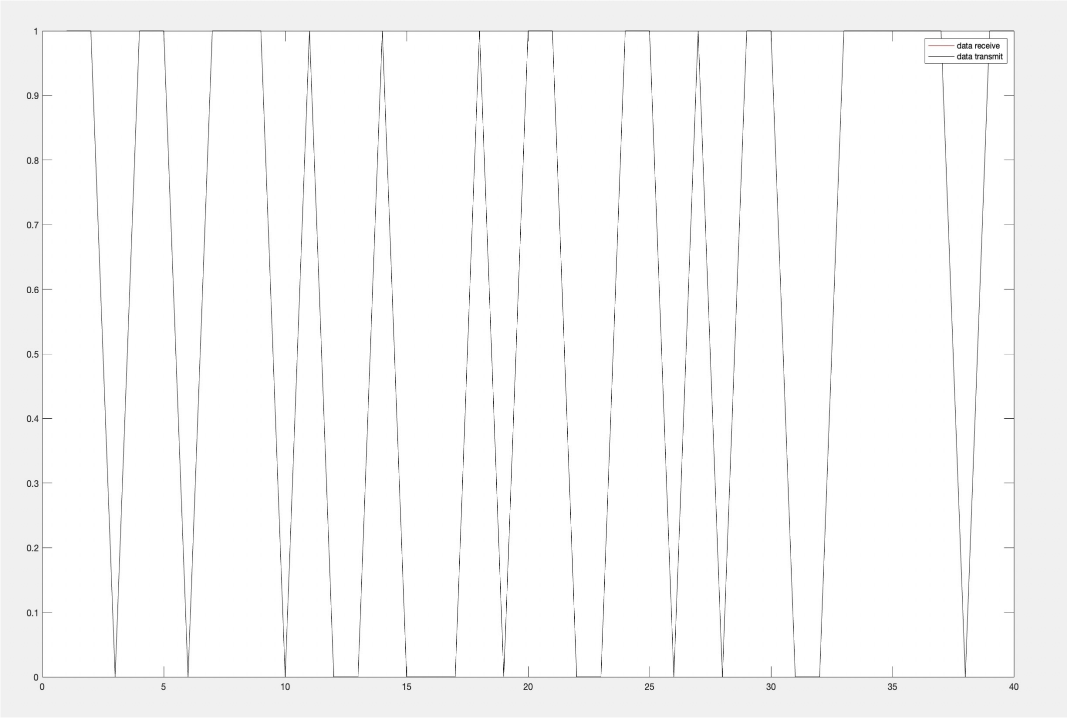
Q2c

c. 沒有雜訊時，matched filter output （即 p. 11 圖中之 $v(T)$ ）之理論值應為何？（以數學推理之）。而用此程式模擬時（請先不加雜訊），變數 D_demapping（於 p. 17 程式中代表 matched filter output）的值為何？理論值和模擬值相同嗎？解釋之。

由於本題假設沒有雜訊，因此傳送端訊號等於接收端訊號。假設傳送訊號為一方波，透過convolution計算結果可知在接收端需要除以$A * 2W = 10$的scaling，由matlab模擬驗證matched filter output $y(t)$ 與比較錯誤率下如下

$$\left[A * \Pi\left(\frac{t}{2W}\right)\right] * \left[A * \Pi\left(\frac{t}{2W}\right)\right] = A^2 * 2W * \Lambda\left(\frac{t}{2W}\right)$$





Q2d

d. 解釋程式碼（p. 17）

D_demapping=D_filtered(10:10:end)/10;

為何需取 10:10:end？

```
%% receiver
D_filtered=conv(Data_receive,p1); % MF output
D_demapping=D_filtered(10:10:end)/10; % sampling at symbol rate
```

$$\left[A * \Pi\left(\frac{t}{2W}\right)\right] * \left[A * \Pi\left(\frac{t}{2W}\right)\right] = A^2 * 2W * \Lambda(\frac{t}{2W})$$

參考c小題解釋，取 matched filter 在 $t = T$ 下的輸出，並除上一個 $A \cdot 2W = 1 \cdot 2 \cdot 5 = 10$ 的 Scaling。

Q2e

e. p. 17 程式碼中，sgma^2 代表 noise sample 之（平均）功率。

解釋為何

```
sgma = sqrt( 0.5/EbN0/2*10 );
```

如此設定可達到所定義之 $\frac{E_b}{N_0}$ 值。

如Q1觀念，定義信噪比$SNR_0 \triangleq \frac{P_{signal}}{P_{noise}} \triangleq \frac{E_v}{N_0}$，

其中N_0代表高斯雜訊的一個定值($S_w(f) \triangleq \frac{N_0}{2}$)、$E_b$代表on-off keying每個bit的平均能量$E_b = \frac{1}{2}A^2T + \frac{1}{2} * 0 = \frac{A^2T}{2}$

由於本題條件$A = 1, T = 1$，因此$E_b = 12$，另外假設取樣頻率$Ts = 10$，每個sample point的noise power為σ_s^2，計算σ_s^2值如下：

$$Let\ N = \int_0^T g(T-\tau)\omega(\tau)d\tau$$

Mean of N : $E[N] = \int_0^T g(T-\tau)E[\omega(\tau)]d\tau = 0$

$$\sigma_s^2 = \frac{1}{T_s} * \sigma^2 = \frac{1}{T_s} * (E[N * N] - E[N]^2)$$

$$= \frac{1}{T_s} * \int_0^T \int_0^T E[\omega(t) * \omega(t)]dtd\tau$$

$$= \frac{1}{T_s} * \int_0^T \int_0^T R_\omega(t-\tau)dtd\tau = \frac{1}{T_s} * \int_0^T \int_0^T \frac{N_0}{2}(t-\tau)dtd\tau$$

$$= \frac{1}{T_s} * \frac{N_0}{2}$$

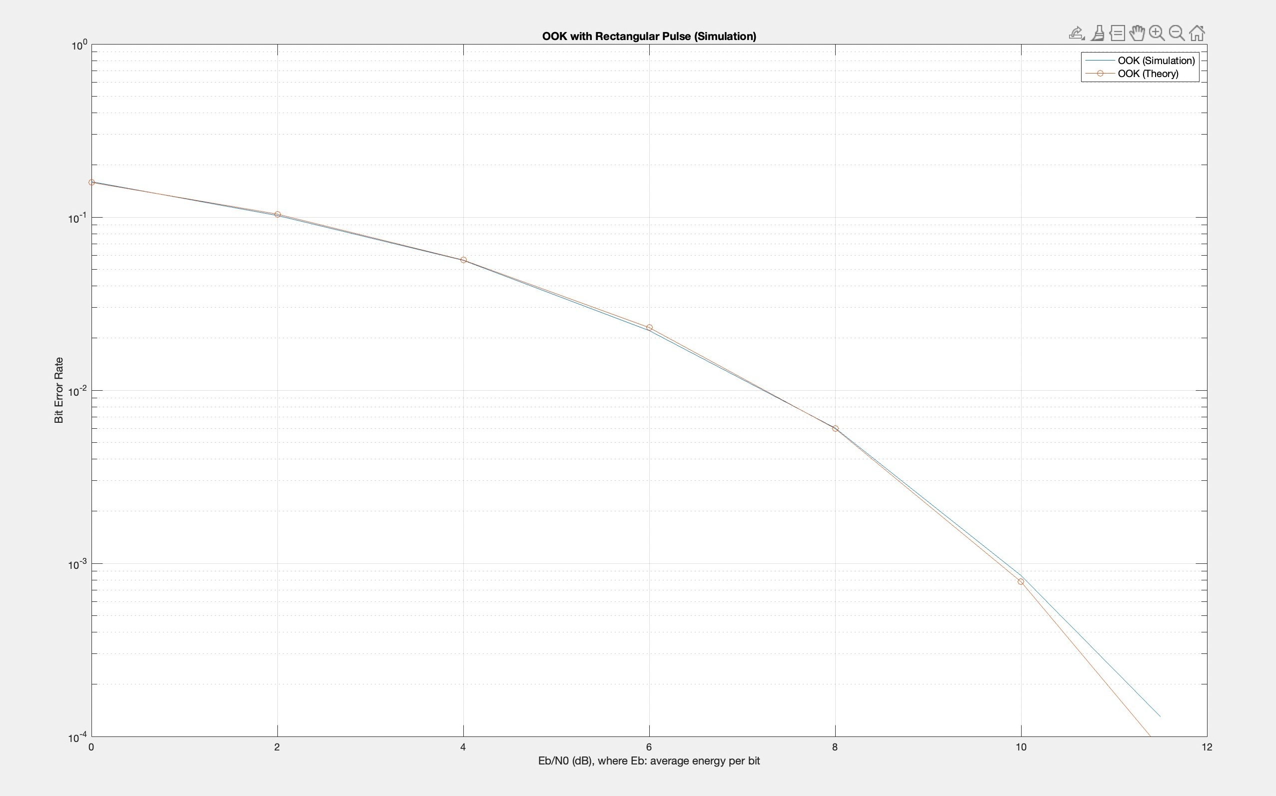
$$\Rightarrow N_0 = 2 * T_s * \sigma_s^2$$

SNR可改寫為 $SNR_0 = \frac{P_{signal}}{P_{noise}} = \frac{\frac{1}{2}\mathrm{A}^2\mathrm{T}}{2 * T_s * \sigma_s^2} = \frac{\frac{1}{2}}{2 * 10 * \sigma_s^2}$

Q2f

f. 畫出 OOK 之模擬錯誤率與 $\frac{E_b}{N_0}$ 之關係圖，並修改範例程式，另外畫出OOK 之理論錯誤率，比較與說明模擬結果與理論結果之差異。

```
D_demap_N = (D_demapping > 0.5); % >0.5: 1; <=0.5: 0
% BER computation
Error_num = sum(xor(D_demap_N,Data_bit)); % same -> 0; diff -> 1
BER = Error_num/data_number;
```



$$P_e = \int_{\frac{AT}{2\sigma}}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{y^2}{2}} dy$$

$$= Q\left(\frac{AT}{2\sigma}\right)$$

$$= Q\left(\frac{AT}{2\sqrt{\frac{N_0}{T}}}\right)$$

$$= Q\left(\sqrt{\frac{A^2T}{2} * \frac{1}{N_0}}\right)$$

$$= Q\left(\sqrt{\frac{E_b}{N_0}}\right)$$

由左圖可知，

理論值計算與Matlab模擬值之間誤差非常小。

Q2g

g. Conditional pdf at MF output：

① 請修改範例程式，畫出

$$f(v(T)\,|\,\text{bit 0 sent}) \text{ 與 } f(v(T)\,|\,\text{bit 1 sent})$$

於 $\frac{E_b}{N_0} = 3$ dB 之理論值。利用 MF output 之模擬值（如變數 D_demapping，但你可能要適當 scale 此變數），畫出 $f(v(T)\,|\,\text{bit 1 sent})$ 之實驗值。

② 於 $\frac{E_b}{N_0} = 10$ dB 時，重複①步驟，並說明 conditional pdf 之變化。

Hint: 使用 MATLAB 指令：histogram。

```
index_0 = 1;
for i = 1:data_number
    if Data_bit(i) == 0 % TX == 0
        D_demapping_0(index_0) = D_demapping(i);
        index_0 = index_0 + 1;
    end
end
```

如果一開始傳送訊號是0，提取輸出訊號$y(T)$並存入另一個自定義的array，代表**選取指定事前機率下的輸出數據**。

```
histogram(D_demapping_0, 'FaceColor', '#0072BD','Normalization','pdf');
```

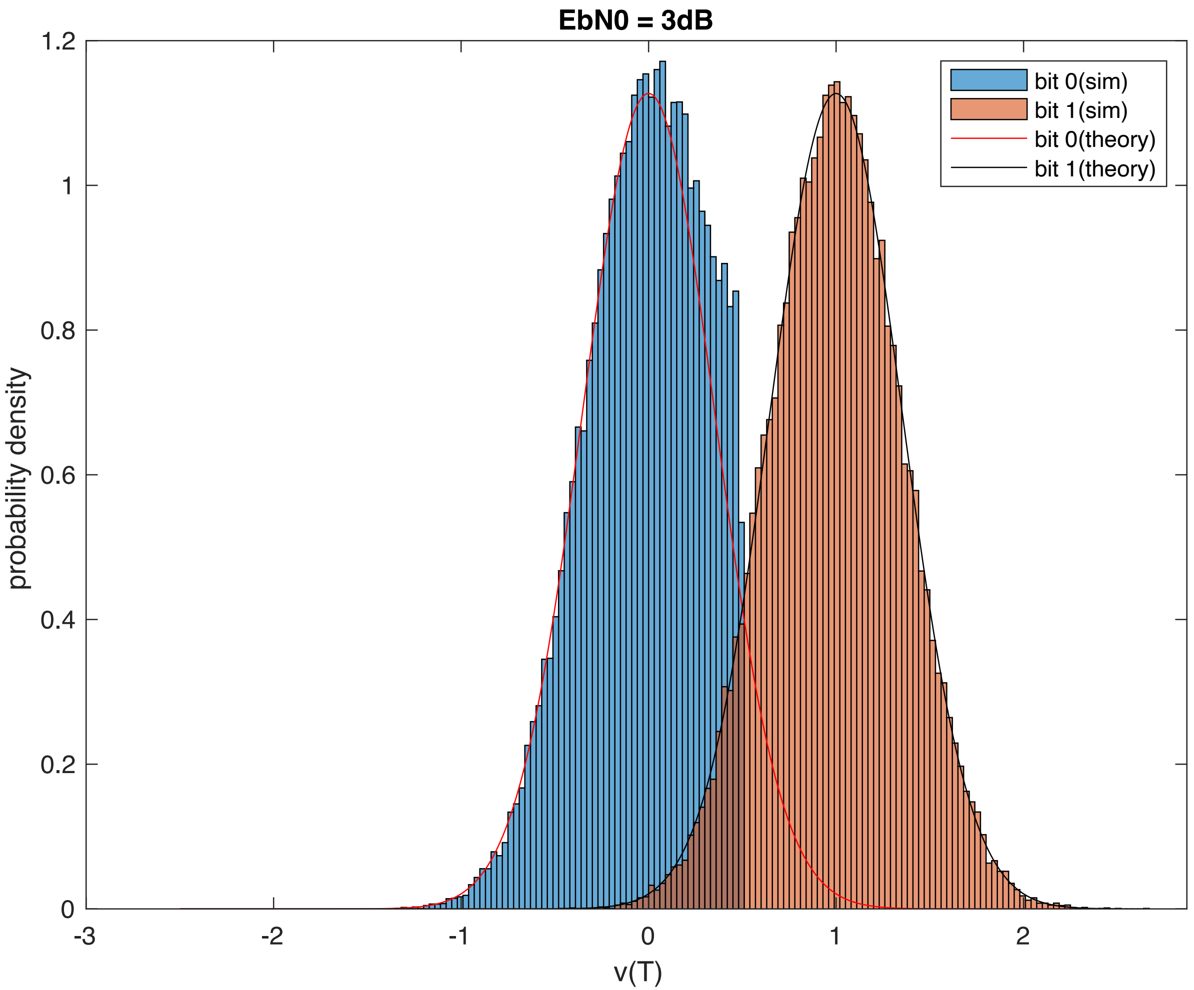
下指令 histogram(X,'Normalization','pdf') 畫各個數值區間內的統計數量，後面那兩個參數是取數據 X 的probability density function。

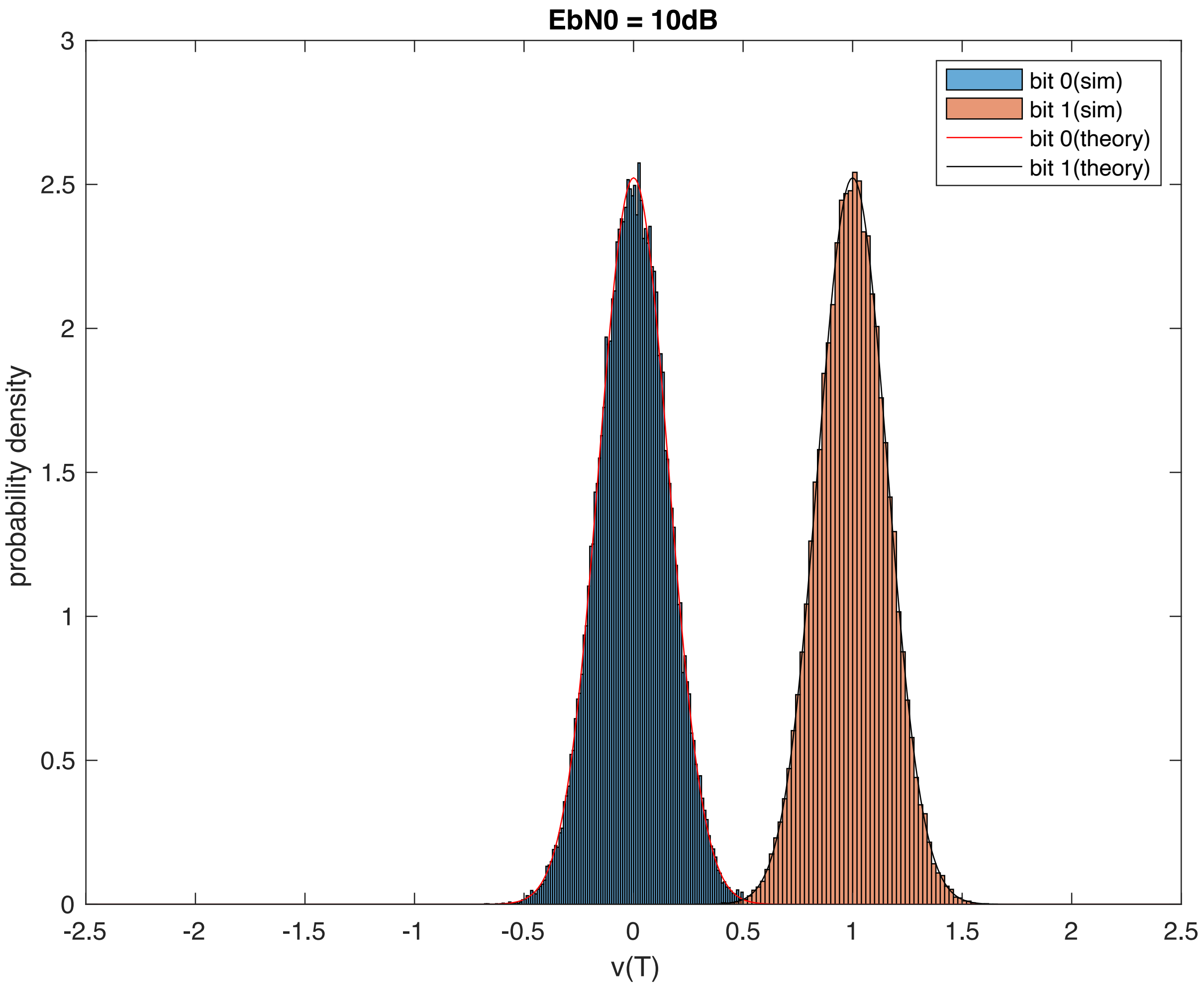
```
plot(xaxis,normpdf(xaxis,0,sgma / sqrt(10)), 'LineWidth',3);
```

理論值計算下指令normpdf，代表使用常態(高斯)分布$N\sim(\mu,\sigma^2)$，其中mean μ 是0，而standard deviation(標準差) σ^2 是需要乘以$\frac{1}{\sqrt{10}}$的scaling，原因是根據Q2c觀念，經過convolution後的訊號要除以1010的scaling才是最終的$y(T)$， $y(T)$其中也包括雜訊 $N\sim(\mu,\sigma^2)$ ，因此$\sigma^2 \rightarrow \frac{\sigma^2}{\sqrt{10}} \Rightarrow \sigma \rightarrow \frac{\sigma}{\sqrt{10}}$。分別就不同$SNR\frac{E_b}{N_0}$下畫出conditional pdf $f(y(T)|bit\ 0\ sent)$與$f(y(T)|bit\ 1\ sent)$ 的理論值與模擬值。

Q2g

降低SNR，在維持E_b的條件下，調動noise的power σ^2 ，會讓σ^2變大，因此左圖的PDF會比右圖的PDF還矮胖。





附件：Q1

```
clear all; clc; close all;
%% parameters
data_number = 4; % # of bits
Fs=10; % sampling frequency (used to generate received samples)


%% transmitter
Data_bit = (rand(1,data_number) > 0.5 ); % random bits
p1 = ones(1,Fs); % discrete-time rectangular pulse that represents one
symbol
Data_pulse_array = (p1')*Data_bit;
Data_pulse = reshape(Data_pulse_array,1,length(p1)*data_number);


%% AWGN channel and receiver
%% AWGN channel
EbN0dB = 10; % 3dB, 10dB
[a, b] = size(Data_pulse);
EbN0 = 10^(EbN0dB/10); % EbN0 is now in linear scale
sgma = sqrt( 0.5/EbN0/2*10 );
noise = normrnd(0, sgma, a, b ) ;
Data_receive=Data_pulse+noise;


%% generate plots
figure;
xais = 1:40;
hold on;
plot(xais, Data_pulse, 'LineWidth',2);
plot(xais, Data_receive, '--r', 'LineWidth',2);
xlim([0 40]);
ylim([-5 5]);
xlabel('sample index');
ylabel('sample value');
legend('TX waveform', 'RX waveform');
title('Waveform of OOK with Rectangular Pulse(Eb/NO=10dB)');
grid on;
```

附件：Q2a

```
clear all; clc; close all;


data_number = 10000;
for j=1:data_number
sum = 0;
Data_bit=(rand(1,j) > 0.5 );
for i=1:length(Data_bit)
sum = sum+Data_bit(i);
end
average(j) = sum/j;
end
x = 1:length(Data_bit);
plot(x, average);
grid on;
axis([1 10000 0 1]);
xlabel('data number');
ylabel('Probability of bit 1');
title('Probability of bit 1 with data number
increasing');
xtick([0:10:10000]);
ytick([0:0.1:1]);
```

附件：Q2c

```
% matched filter
% - on-off keying (OOK) using rectangular pulse (T=1)
% That is, when OFF, assuming A=0, thus E1 = 0
% when ON: assuming A=1, thus E2 = 1
% The “average bit energy” (Eb) = (E1+E2)/2 = 1/2 %
clear all; close all;
%% parameters
data_number = 10^5; % # of bits
EbN0dB_vec = [0:2:10 11.5]; % Eb/N0 in dB
Fs=10; % sampling frequency (used to generate received
samples)


%% transmitter
Data_bit=(rand(1,data_number) > 0.5 ); % random bits
p1=ones(1,Fs);% discrete-time rectangular pulse that
represents one symbol
Data_pulse_array=(p1.')*Data_bit;
Data_pulse=reshape(Data_pulse_array,1,length(p1)*data_number);
```

```
for kk=1:length(EbN0dB_vec)
%% AWGN channel
noise = 0 ;
Data_receive=Data_pulse+noise; % received samples
%% receiver
D_filtered=conv(Data_receive,p1); % MF output
D_demapping=D_filtered(10:10:end)/10; % sampling at symbol
rate
% decsion based on D_demapping
D_demap_N = (D_demapping > 0.5); % >0.5: 1; <=0.5: 0
% BER computation
Error_num=sum(xor(D_demap_N,Data_bit));
BER(kk) = Error_num/data_number;
%fprintf('EbN0 in dB is %g\n',EbN0dB);
%fprintf('Bit error rate is %g\n',BER);
end


%% generate plots
figure;
plot(EbN0dB_vec, BER, 'o-');
xlabel('Eb/N0 (dB), where Eb: average energy per bit');
ylabel('Bit Error Rate')
legend('OOK (Simulation)');
grid
axis([0 12 10^-4 1])
title('OOK with Rectangular Pulse (Simulation)')
```

附件：Q2f

```
% matched filter
% - on-off keying (OOK) using rectangular pulse (T=1)
% That is, when OFF, assuming A=0, thus E1 = 0
% when ON: assuming A=1, thus E2 = 1
% The “average bit energy” (Eb) = (E1+E2)/2 = 1/2 %
clear all; close all;
%% parameters
data_number = 10^5; % # of bits
EbN0dB_vec = [0:2:10 11.5]; % Eb/N0 in dB
Fs=10; % sampling frequency (used to generate received samples)


%% transmitter
Data_bit=(rand(1,data_number) > 0.5 ); % random bits
p1=ones(1,Fs);% discrete-time rectangular pulse that represents one symbol
Data_pulse_array=(p1.')*Data_bit;
Data_pulse=reshape(Data_pulse_array,1,length(p1)*data_number);
```

```
for kk=1:length(EbN0dB_vec)
%% AWGN channel
[a b] = size(Data_pulse);
EbN0dB = EbN0dB_vec(kk);
EbN0 = 10^(EbN0dB/10); % EbN0 is now in linear scale
sgma = sqrt( 0.5/EbN0/2*10 );
noise = normrnd(0, sgma, a, b ) ;
Data_receive=Data_pulse+noise; % received samples
%% receiver
D_filtered=conv(Data_receive,p1); % MF output
D_demapping=D_filtered(10:10:end)/10; % sampling at symbol rate
% decsion based on D_demapping
D_demap_N = (D_demapping > 0.5); % >0.5: 1; <=0.5: 0
% BER computation
Error_num=sum(xor(D_demap_N,Data_bit));
BER(kk) = Error_num/data_number;
%fprintf('EbN0 in dB is %g\n',EbN0dB);
%fprintf('Bit error rate is %g\n',BER);


%% theory BER for ook
ber_theory(kk)=qfunc(sqrt(EbN0));
end


%% generate plots
figure;
semilogy(EbN0dB_vec, BER, EbN0dB_vec,ber_theory,'o-');
hold on;
xlabel('Eb/N0 (dB), where Eb: average energy per bit');
ylabel('Bit Error Rate')
legend('OOK (Simulation)', 'OOK (Theory)','FontSize',10);
grid
axis([0 12 10^-4 1])
title('OOK with Rectangular Pulse (Simulation)')
```

附件：Q2g

```
% matched filter
% - on-off keying (OOK) using rectangular pulse (T=1)
% That is, when OFF, assuming A=0, thus E1 = 0
% when ON: assuming A=1, thus E2 = 1
% The “average bit energy” (Eb) = (E1+E2)/2 = 1/2 %
clear all; close all;
%% parameters
data_number = 10^5; % # of bits
EbN0dB_vec = [0:2:10 11.5]; % Eb/N0 in dB
Fs=10; % sampling frequency (used to generate received samples)


%% transmitter
Data_bit=(rand(1,data_number) > 0.5 ); % random bits
p1=ones(1,Fs);% discrete-time rectangular pulse that represents one
symbol
Data_pulse_array=(p1.')*Data_bit;
Data_pulse=reshape(Data_pulse_array,1,length(p1)*data_number);
```

```
%% AWGN channel
[a b] = size(Data_pulse);
EbN0dB = 3; % 3dB, 10dB;
EbN0 = 10^(EbN0dB/10); % EbN0 is now in linear scale
sgma = sqrt( 0.5/EbN0/2*10 );
noise = normrnd(0, sgma, a, b ) ;
Data_receive=Data_pulse+noise; % received samples
%% receiver
D_filtered=conv(Data_receive,p1); % MF output
D_demapping=D_filtered(10:10:end)/10; % sampling at symbol rate
% decsion based on D_demapping
D_demap_N = (D_demapping > 0.5); % >0.5: 1; <=0.5: 0


k=1;
for i = 1:data_number
if D_demap_N(i) == 0
D_demap_0(k)=D_demapping(i);
k = k+1; end
end
index_1 = 1;
for i = 1:data_number
if Data_bit(i) == 1
D_demap_1(index_1) = D_demapping(i);
index_1 = index_1 + 1; end
end

%% generate plots
histogram(D_demap_0,'Normalization','pdf');
hold on
histogram(D_demap_1,'Normalization','pdf')
xaxis = -2.5:0.01:2.5;
plot(xaxis,normpdf(xaxis,0,sgma/(sqrt(10))),'r')
hold on
plot(xaxis,normpdf(xaxis,1,sgma/(sqrt(10))),'k')

legend('bit 0(sim)','bit 1(sim)','bit 0(theory)','bit 1(theory)');
xlabel('v(T)');ylabel('probability density')
title('EbN0 = 3dB')
```