40

Mahdis Mosaheb

Homework #3, 695C

This code can get any generating matrix and create the all possible codeword vectors. The output of a binary counter multiply to a generating matrix and create the code vector corresponding to the binary information vactor.

I ran the code for G1 and G2 and the output matrices are below matrices such that every column is a codeword.

G1=

```
0     1     1     1
1     0     1     1
1     1     0     1
1     0     0     0
0     1     0     0
0     0     1     0
0     0     0     1
```

G2=

```
1     0     0     0
1     0     0     1
1     0     1     0
1     0     1     1
1     1     0     0
1     1     0     1
1     1     1     0
1     1     1     1
```

X1 =

```
0     1     1     0     1     0     0     1     0     1     1     0     1     0     0     1
0     1     1     0     0     1     1     0     1     0     0     1     1     0     0     1
0     1     0     1     1     0     1     0     1     0     1     0     0     1     0     1
0     0     0     0     0     0     0     0     1     1     1     1     1     1     1     1
0     0     0     0     1     1     1     1     0     0     0     0     1     1     1     1
0     0     1     1     0     0     1     1     0     0     1     1     0     0     1     1
0     1     0     1     0     1     0     1     0     1     0     1     0     1     0     1
```

X2 =

```
0     0     0     0     0     0     0     0     1     1     1     1     1     1     1     1
0     1     0     1     0     1     0     1     1     0     1     0     1     0     1     0
0     0     1     1     0     0     1     1     1     1     0     0     1     1     0     0
0     1     1     0     0     1     1     0     1     0     0     1     1     0     0     1
0     0     0     0     1     1     1     1     1     1     1     1     0     0     0     0
0     1     0     1     1     0     1     0     1     0     1     0     0     1     0     1
0     0     1     1     1     1     0     0     1     1     0     0     0     0     1     1
0     1     1     0     1     0     0     1     1     0     0     1     0     1     1     0
```

Question1

1-2) This implementation is a codeword-wise ML decoder, which gets the four probabilities (p00, p01, p10, p11) and also the Yobs vector and calculates the joint probability of Y obs with the all codewords and the codeword which has the maximum probability is going to be the decoded Yobs.

1-3) This part approximates the FER and BER of the codewise-decoder .

FER: In order to get the FER, at first we should calculate the error probability of any possible Y . That means to find the probability of decoded Y is not equal to x (all x vector).

FER= $\sum_{y=000000}^{111111} \sum_{0}^{15} P_{\vec{x},\vec{y}}(\vec{x},\vec{y})\{f(y) \neq x\}$

We need to check the decoded y with all X's, in order to find which one is not equal to f(y), then get the summation of those probabilities. Also, we shouldn't forget to get the error probability for all possible y's .

BER: In order to calculate BER, we should at first find the Hamming distance between any decoded y and all possible codewords(X).Hamming distance gives us the number of bit differences of two vectors. Then we multiply the Hamming distance of two vectors(X, f(y)) by the joint probability of the same X and y . We repeat this algorithm for all possible y and then divide by the number of bits in one codeword to get the BER.

BER= $1/n \sum_{y=000000}^{111111} \sum_{0}^{15} P_{\vec{x},\vec{y}}(\vec{x},\vec{y})|f(y) - x|$

Where $|f(y) - x|$ *is hamming distance.*

Question2

2-2) To implement the Bitwise-decoder, at the first step we calculate the probability of first bit of X. for example, we calculate the probability of X(1)=1 given Yobs , if it was less than ½ the bitwise decoder, decides the first bit of decoded y is going to be 0 otherwise it is 1 . We repeat that for all bits of X.

The problem of this decoder is sometimes the decoded vector is not even a codeword. But this decoder minimizes the BER.

2-3) The functions of FER and BER in this question are the same as question 1 but the decoder that we use here is bit-wise decoder,.

According to the graphs, we can see the FER has the better performance by codewise decoder after the jump in the FERbitwise , before the jump, they show the equal performance. On the other hand, as we said before, BER has better performance with bitwise decoder. The BERbitwise and BERcodewise are the same at first but eventually the BERbitwise decreased from BERcodewise. So we can conclude, Bitwise decoder is more efficient for BER than Codewise decoder.

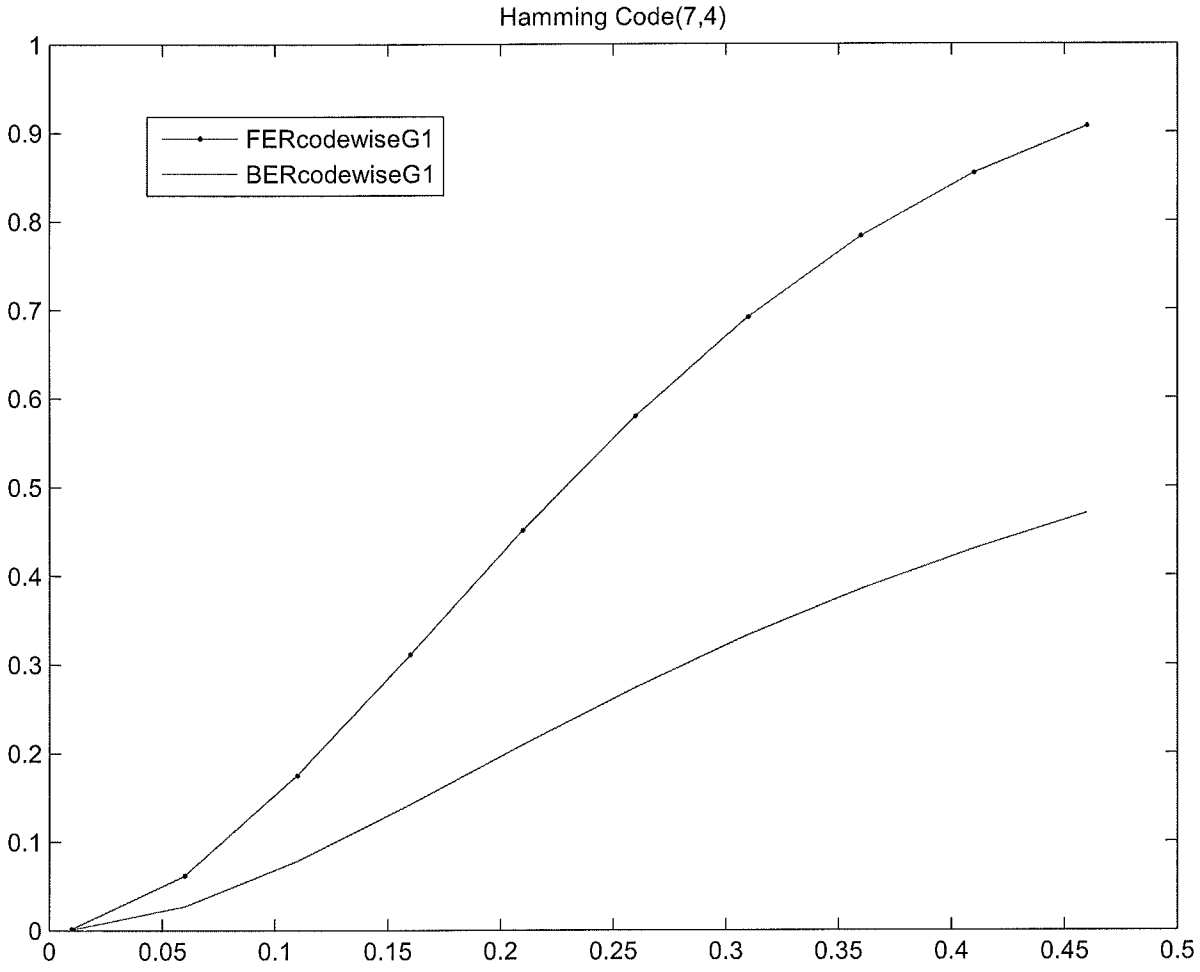
```
G = [0 1 1 1;1 0 1 1;1 1 0 1;1 0 0 0; 0 1 0 0; 0 0 1 0;0 0 0 1];
%G = [1 0 0 0;1 0 0 1;1 0 1 0;1 0 1 1;1 1 0 0; 1 1 0 1; 1 1 1 0; 1 1 1 1 ]
column = length(G(1,:));
row = length(G(:,1));
X = zeros(row,2^column);

temp=dec2bin((0:2^column-1));
for i=1:2^column
    for j=1:column
        I(j,i)=str2num(temp(i,j));
    end
end
tempy=dec2bin((0:2^row-1));
for i=1:2^row
    for j=1:row
       Y(j,i)=str2num(tempy(i,j));
    end
end
for x = 1:row;
    for z = 1:2^column;
        for y = 1:column;
           X(x, z) = xor(X(x, z) , and(G(x, y) , I(y, z)));
    end
    end
end
X1 =X
%X2 =X
```

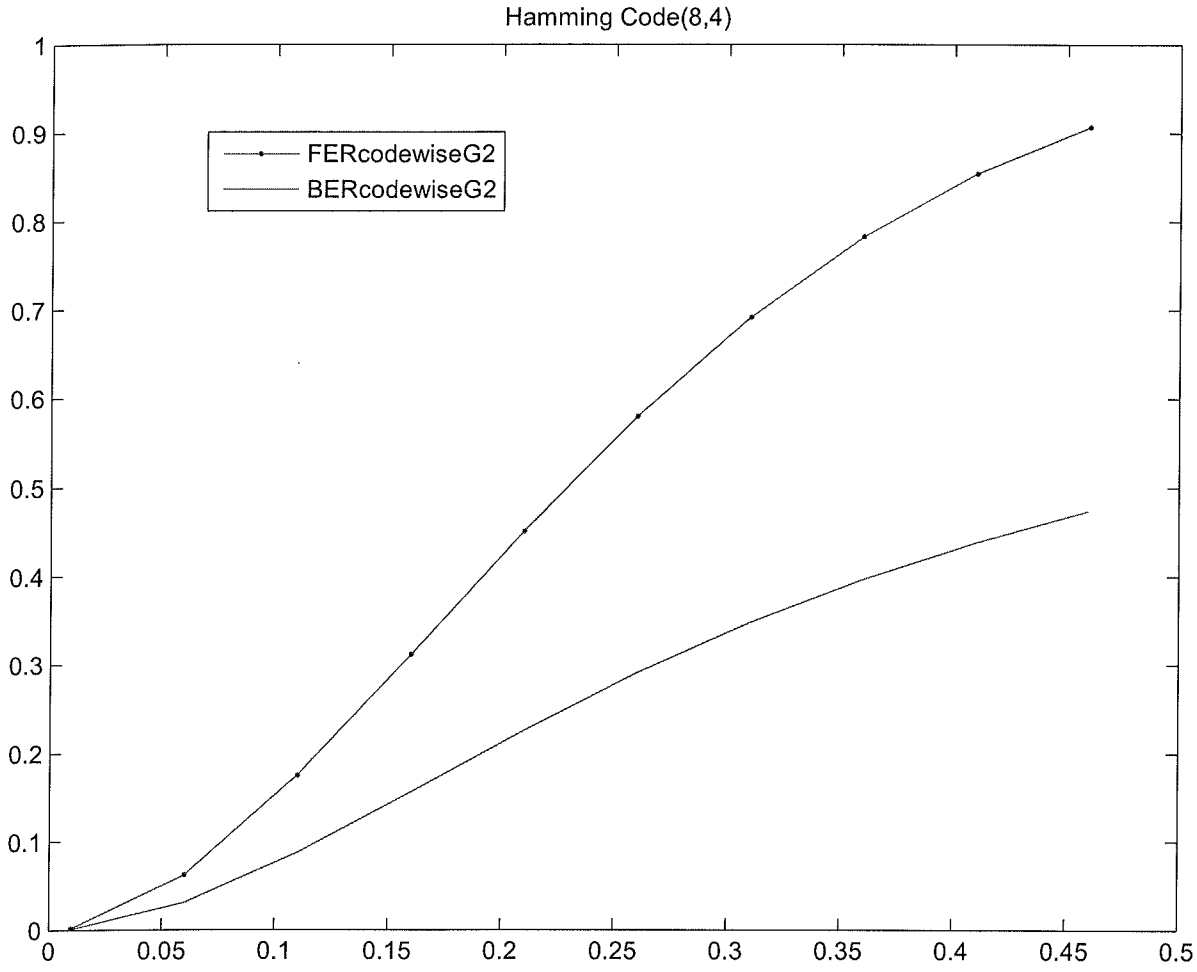
```
G=input('G')
column = length(G(1,:));
row = length(G(:,1));
X = zeros(row,2^column);
% binary counter
temp=dec2bin((0:2^column-1));
for i=1:2^column
    for j=1:column
        I(j,i)=str2num(temp(i,j));
    end
end
for x = 1:row;
    for z = 1:2^column;
        for y = 1:column;
           X(x, z) = xor(X(x, z) , and(G(x, y) , I(y, z)));
    end
    end
end
p00=input('p00')
p01=input('p01')
p10=input('p10')
p11=input('p11')
Y1= input('Yobs')
Y= Y1.';

for x=1:2^column;
np00=0;
np01=0;
np10=0;
np11=0;
    for y=1:row;
        if X(y,x)==0 &Y(y,1)==0
            np00=np00+1
        end
        if X(y,x)==0 &Y(y,1)==1
            np01=np01+1
        end
        if X(y,x)==1 & Y(y,1)==0
            np10=np10+1
        end
        if X(y,x)==1 & Y(y,1)==0
            np11=np11+1
        end
    end
    L(1,x)= (p00^np00)*(p01^np01)*(p10^np10)*(p11^np11);

end
[maxVal maxInd] = max(L)
m(1,1:column)=I(1:column,maxInd)
XML(1:row,1)=X(1:row,maxInd)
```

```
%G = [0 1 1 1;1 0 1 1;1 1 0 1;1 0 0 0; 0 1 0 0; 0 0 1 0;0 0 0 1];
 G = [1 0 0 0;1 0 0 1;1 0 1 0;1 0 1 1;1 1 0 0; 1 1 0 1; 1 1 1 0; 1 1 1 1 ];
column = length(G(1,:));
row = length(G(:,1));
X = zeros(row,2^column);
XML = zeros(row,2^row);
%p=0.1
p=0.01:0.05:0.5;
% binary counter
temp=dec2bin((0:2^column-1));
for i=1:2^column
    for j=1:column
        I(j,i)=str2num(temp(i,j));
    end
end
tempy=dec2bin((0:2^row-1));
for i=1:2^row
    for j=1:row
       Y(j,i)=str2num(tempy(i,j));
    end
end
for x = 1:row;
    for z = 1:2^column;
        for y = 1:column;
           X(x, z) = xor(X(x, z) , and(G(x, y) , I(y, z)));
    end
    end
end
for P=1:length(p(1,:))
p00=1-p(1,P);
p01=p(1,P);
FERC = 0;
BERC =0;
BERC1=0;
for i=1:2^row;

    for x=1:2^column;

    np00=0;
    np01=0;



        for y=1:row;
        if (X(y,x)==Y(y,i))
            np00=np00+1;
        end
        if (X(y,x)~=Y(y,i))
            np01=np01+1;

        end
```

```
        end


    L(1,x)= (p00.^np00).*(p01.^np01);
    end
    [maxVal maxInd] = max(L);

    XML(:,i)=X(1:row,maxInd);% at this point we have the decoded Y(i)
        for k =1:2^column
        mp00=0;
        mp01=0;

        FERi=0;
        BERi=0;
        np00=0;
        np01=0;

        for z=1:row % Hamming distance
                if X(z,k)== XML(z,i)
                    mp00=mp00+1;
                end
                if X(z,k)~= XML(z,i)
                    mp01=mp01+1;
                end

       end

            for z=1:row
                if X(z,k)==Y(z,i)
                    np00=np00+1;
                end
                if X(z,k)~=Y(z,i)
                    np01=np01+1;
                end

            end
      FL(1,k)= (p00.^np00).*(p01.^np01);

      BL(1,k)= (mp01).*FL(1,k);


       end
    BERi= (1/2^column).*(sum(BL.'));
    FERi=(1/2^column).*(sum(FL.')-(max(FL.')));
    BERC1=(BERC1+BERi);
    FERC=FERC+FERi;
end
BERC = (1./row ).* BERC1;
BERcode(1,P)= BERC
FERcode(1,P)=FERC
end
```

```
plot(p,FERcode,'.-')
axis([0 0.5 0 1])
hold on
plot(p,BERcode)
axis([0 0.5 0 1])
%legend('FERcodewiseG1','BERcodewiseG1');
legend('FERcodewiseG2','BERcodewiseG2');
%title('Hamming Code(7,4)')

title('Hamming Code(8,4)')
hold off
```


Hamming Code(7,4)
FERcodewiseG1
BERcodewiseG1
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5


Hamming Code(8,4)
FERcodewiseG2
BERcodewiseG2
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5

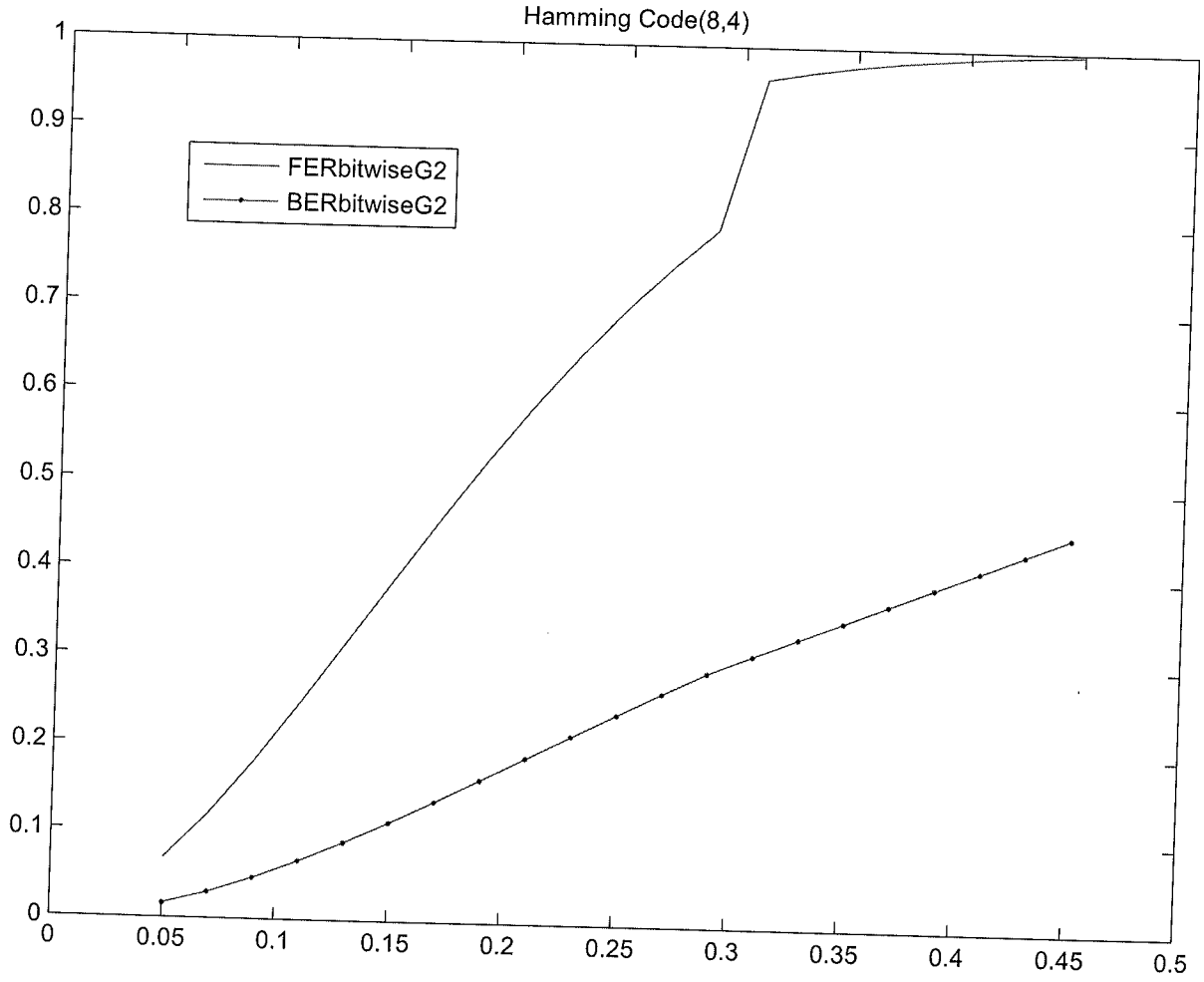
```
%G = [0 1 1 1;1 0 1 1;1 1 0 1;1 0 0 0; 0 1 0 0; 0 0 1 0;0 0 0 1];
G= [1 1 0 1;1 0 1 1;0 1 1 1; 1 0 0 0;0 1 0 0;0 0 1 0; 0 0 0 1];
column = length(G(1,:));
row = length(G(:,1));
X = zeros(row,2^column);
% binary counter
temp=dec2bin((0:2^column-1));
for i=1:2^column
    for j=1:column
        I(j,i)=str2num(temp(i,j));
    end
end
for x = 1:row;
    for z = 1:2^column;
        for y = 1:column;
           X(x, z) = xor(X(x, z) , and(G(x, y) , I(y, z)));
    end
    end
end
p00=input('p00')
p01=input('p01')
p10=input('p10')
p11=input('p11')
Y1= input('Yobs')
Y= Y1.';
for x=1:row;
    for y=1:2^column;
        np00=0;
        np01=0;
        np10=0;
        np11=0;
            for z=1:row;
                if X(z,y)==0 &Y(z,1)==0
                    np00=np00+1;
                end
                if X(z,y)==0 &Y(z,1)==1
                    np01=np01+1;
                end
                if X(z,y)==1 & Y(z,1)==0
                    np10=np10+1;
                end
                if X(z,y)==1 & Y(z,1)==0
                    np11=np11+1;
                end

            end
            if X(x,y)==1
                L(1,y)= (p00^np00)*(p01^np01)*(p10^np10)*(p11^np11);
                T(1,y)= (p00^np00)*(p01^np01)*(p10^np10)*(p11^np11);
            else
```

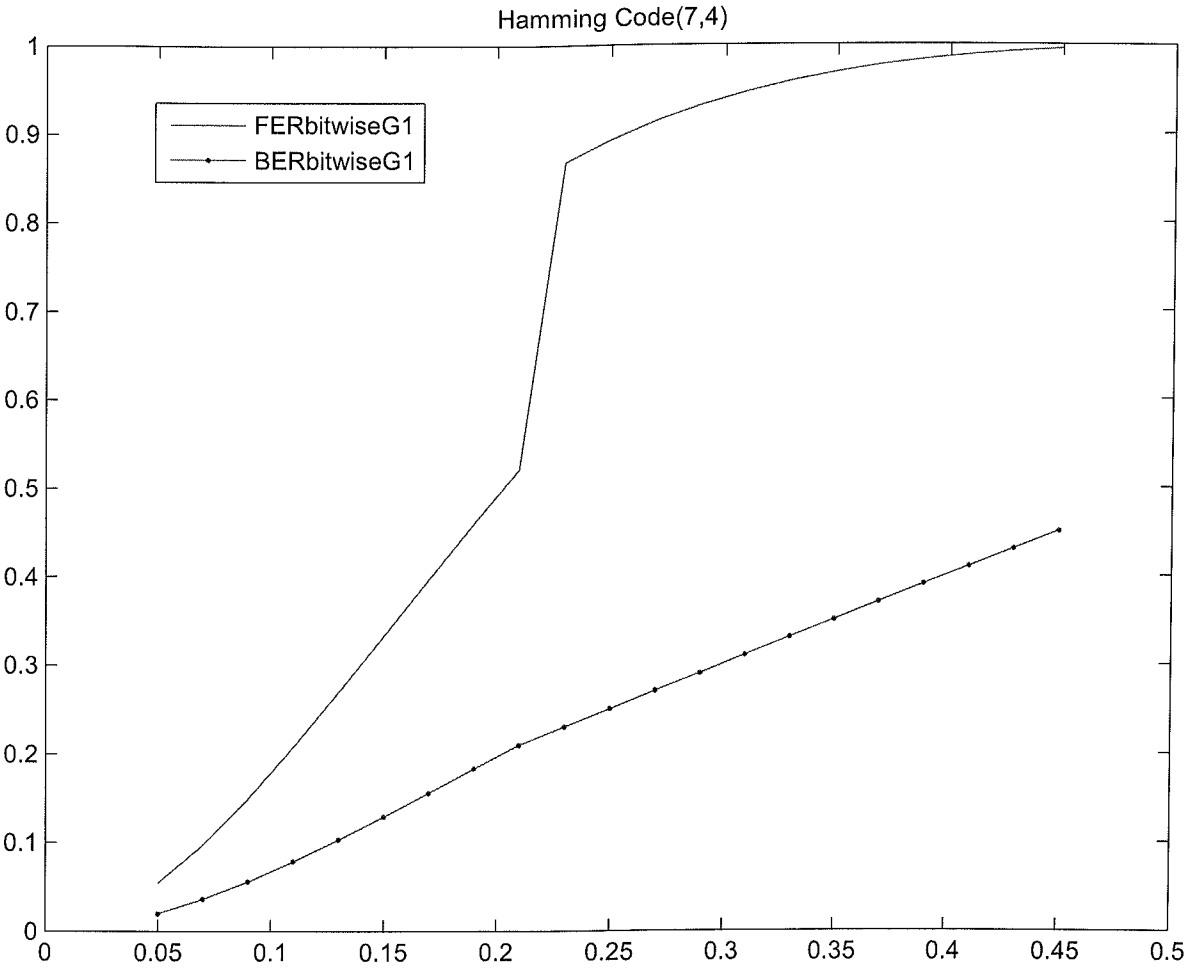
```
            L(1,y)=0;
            T(1,y)=(p00^np00)*(p01^np01)*(p10^np10)*(p11^np11);
            end


    end
    LB=sum(L.');
    LT=sum(T.');
    Detector=(LB/LT);
    if Detector < 0.5
        XML(1,x)=0;
    else
        XML(1,x)=1;
    end

end

        Output=num2str(XML)
```

```
%G = [0 1 1 1;1 0 1 1;1 1 0 1;1 0 0 0; 0 1 0 0; 0 0 1 0;0 0 0 1];
G = [1 0 0 0;1 0 0 1;1 0 1 0;1 0 1 1;1 1 0 0; 1 1 0 1; 1 1 1 0; 1 1 1 1 ];
column = length(G(1,:));
row = length(G(:,1));
X = zeros(row,2^column);



p=0.05:0.02:0.45;
FERbit=zeros(1,length(p(1,:)));
% binary counter
temp=dec2bin((0:2^column-1));
for i=1:2^column
    for j=1:column
        I(j,i)=str2num(temp(i,j));
    end
end
tempy=dec2bin((0:2^row-1));
for i=1:2^row
    for j=1:row
       Y(j,i)=str2num(tempy(i,j));
    end
end
for x = 1:row;
    for z = 1:2^column;
        for y = 1:column;
           X(x, z) = xor(X(x, z) , and(G(x, y) , I(y, z)));
    end
    end
end
for P=1:length(p(1,:))
p00=1-p(1,P);
p01=p(1,P);

FERB1 = 0;
BERB =0;
BERB1=0;
for i=1:2^row;

    %Bitwise-decoder
for x=1:row
    for y=1:2^column
        np00=0;
        np01=0;
            for z=1:row
                if X(z,y)==Y(z,i)
                    np00=np00+1;
                end
                if X(z,y)~=Y(z,i)
                    np01=np01+1;
                end
```

```
            end
            if X(x,y)==1
               L(1,y)= (p00.^np00).*(p01.^np01);
            else

               L(1,y)=0;
            end
                T(1,y)=(p00.^np00).*(p01.^np01);

    end
    LB=sum(L.');
    LT=sum(T.');
    Detector=(LB./LT);
    if Detector < 0.5
        XML(x,i)=0;
    else
        XML(x,i)=1;
    end
end % at this point XML (:,i) is decoded vector of Y(i)
for k =1:2^column
        mp00=0;
        mp01=0;
       FERBi=0;
       np00=0;
        np01=0;

        for z=1:row % Hamming distance
                if X(z,k)== XML(z,i)
                    mp00=mp00+1;
                end
                if X(z,k)~= XML(z,i)
                    mp01=mp01+1;
                end

       end

           for z=1:row
               if X(z,k)==Y(z,i)
                   np00=np00+1;
               end
               if X(z,k)~=Y(z,i)
                   np01=np01+1;
               end

           end

        if all(X(:,k)== XML(:,i))
            LEFR(1,k)=0;

      else
```

```
            LFER(1,k) = (1/2^column).*(p00.^np00).*(p01.^np01);
         end
            BERtemp(1,k)= (mp01).*(p00.^np00).*(p01.^np01);
end
        BERi=(1/2^ column).*(sum(BERtemp.'));
        FERBi= sum(LFER.');
        BERB1=(BERB1+BERi);
        FERB1=FERB1+FERBi;
end
    BERB= (1/row).*(BERB1);
   BERbit(1,P)= BERB;
   FERbit(1,P)=FERB1;



end



plot(p,FERbit)
axis([0 0.5 0 1])
hold on
plot(p,BERbit,'.-')
axis([0 0.5 0 1])
%legend('FERbitwiseG1','BERbitwiseG1');
legend('FERbitwiseG2','BERbitwiseG2');
%title('Hamming Code(7,4)')

title('Hamming Code(8,4)')
hold off
```


Hamming Code(8,4)
FERbitwiseG2
BERbitwiseG2
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5


Hamming Code(7,4)
FERbitwiseG1
BERbitwiseG1
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5

HW3

Monday, October 10, 2022 9:27 AM

Q12: Suppose $m = a \cdot b$

where $a \geq 2$ and $b \geq 2$ are the factors of m.

then $a \cdot b = a \boxdot b \mod m$

$= 0.$

$\Rightarrow \cdot$ is not a commutative group

over $\{0, 1, \cdots, m-1\} \setminus \{0\}$

$\Rightarrow$ It is not a field.

Q13: Proof: ① Closedness:

$(4 \cdot k_1) + (4 k_2) = (4k_1) \boxplus (4k_2) \mod 32$

$= 4(k_1 \boxplus k_2) \mod 32$

$= 4 \cdot ((k_1 \boxplus k_2) \mod 8)$

② Associativity.

② Associativity.

$$\left[(4k_1)+(4k_2)\right]+(4k_3)$$

$$= 4(k_1 \boxplus k_2 \boxplus k_3) \bmod 32$$

$$= (4k_1)+\left[(4k_2)+(4k_3)\right]$$

③ Identity: 0

④ Inverse: $4k_1 \xrightarrow{\text{inverse}} 4\cdot(8 \boxminus k_1)$

$$\Rightarrow 4k_1 + 4(8 \boxminus k_1)$$

$$= 4(8 \boxminus k_1) + 4k_1 = 4(k_1 \boxplus 8 \boxminus k_1) \bmod 32$$

$$= 0.$$

Q14: $X+Y+Z+W=1$

$X+Y+W=1 \Rightarrow Z=0.$

$$X + \quad Z + W = 0 \implies Y = 1$$

$$Y + Z + W = 0 \implies X = 1$$

$$\implies W = 1$$ ※