Title : program for coding and decoding of linear block codes

Experiment No.:10

```
% Given H Matrix
H =  [1 1 1 0 1 0 0;
        0 1 1 1 0 1 0;
        1 1 0 1 0 0 1]

k = 4;
n = 7;

% Generating G Matrix

% Taking the H Matrix Transpose
P = H';

% Making a copy of H Transpose Matrix
L = P;

% Taking the last 4 rows of L and storing
L((5:7), : ) = [];

% Creating a Identity matrix of size K x K
I = eye(k);

% Making a 4 x 7 Matrix
G = [I L]

% Generate U data vector, denoting all information sequences
no = 2 ^ k

% Iterate through an Unit-Spaced Vector
for i = 1 : 2^k

% Iterate through Vector with Specified Increment
% or in simple words here we are decrementing 4 till we get 1
for j = k : -1 : 1
       if rem(i - 1, 2 ^ (-j + k + 1)) >= 2 ^ (-j + k)
       u(i, j) = 1;
       else
       u(i, j) = 0;
       end

       % To avoid displaying each iteration/loop value
       echo off;
end
end

echo on;
```

```
u

% Generate CodeWords
c = rem(u * G, 2)

% Find the min distance
w_min = min(sum((c(2 : 2^k, :))'))

% Given Received codeword
r = [0 0 0 1 0 0 0];
r

p = [G(:, n - k + 2 : n)];

%Find Syndrome
ht = transpose(H)

s = rem(r * ht, 2)

for i = 1 : 1 : size(ht)
if(ht(i,1:3)==s)
       r(i) = 1-r(i);
       break;
end
end

disp('The Error is in bit:')
disp(i)

disp('The Corrected Codeword is :')
disp(r)
```

Output:

>> lbc_final

% Given H Matrix

H = [1 1 1 0 1 0 0;

0 1 1 1 0 1 0;

1 1 0 1 0 0 1]

H =

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

```
k = 4;
n = 7;

% Generating G Matrix

% Taking the H Matrix Transpose
P = H';

% Making a copy of H Transpose Matrix
L = P;

% Taking the last 4 rows of L and storing
L((5:7), : ) = [];

% Creating a Identity matrix of size K x K
I = eye(k);

% Making a 4 x 7 Matrix
G = [I L]
```

G =

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

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

```
% Generate U data vector, denoting all information sequences
no = 2 ^ k

no =

   16


% Iterate through an Unit-Spaced Vector
for i = 1 : 2^k

% Iterate through Vector with Specified Increment
% or in simple words here we are decrementing 4 till we get 1
for j = k : -1 : 1
        if rem(i - 1, 2 ^ (-j + k + 1)) >= 2 ^ (-j + k)
        else
        u(i, j) = 0;
        end

        % To avoid displaying each iteration/loop value
        echo off;
u

u =
```

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

```
% Generate CodeWords
c = rem(u * G, 2)
```

```
c =

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

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

```
% Find the min distance
w_min = min(sum((c(2 : 2^k, :))'))
```

```
w_min =

    3
```

```
% Given Received codeword
r = [0 0 0 1 0 0 0];
r
```

```
r =

    0    0    0    1    0    0    0
```

```
p = [G(:, n - k + 2 : n)];

%Find Syndrome
ht = transpose(H)

ht =

    1    0    1
    1    1    1
    1    1    0
    0    1    1
    1    0    0
    0    1    0
    0    0    1


s = rem(r * ht, 2)

s =

    0    1    1


for i = 1 : 1 : size(ht)
Warning: Colon operands must be real scalars. This warning will become an error in a future release.
> In lbc_final (line 66)

if(ht(i,1:3)==s)
```

```
end
end
if(ht(i,1:3)==s)
end
end
end
if(ht(i,1:3)==s)
end
end
end
if(ht(i,1:3)==s)
        r(i) = 1-r(i);
        break;

disp('The Error is in bit:')
The Error is in bit:
disp(i)
    4



disp('The Corrected Codeword is :')
The Corrected Codeword is :
disp(r)
    0    0    0    0    0    0    0

>>
```