Battula Venkata Sai Ankit
2019AAPS0331H

ECE F344 Information Theory and Coding
Assignment 2

Fixed Length Binary Source Encoder and Decoder

Theory and Definition:

Fixed Length encoding refers to using an encoding definition to map the source code to its definition counterpart. Decoding is just opposite to the encoding process by reverse mapping the encoding definition. A code that converts a fixed number of source symbols to a fixed number of output symbols. It's commonly in the form of a block code.

Commands Used:

1. strlength: Used to calculate length of string.
2. append : combines two strings and concatenates them
3. containers.Map : Data Structure used to store key value pairs.

Approach:

1. We have created a Map data structure which is used to store key value pairs of alphabets and binary code word counterparts.
2. For Binary Encoder, the map is (letters, binaries). For encoding the source we take each alphabet from the string and map it to its binary counterpart and append to a string.
3. For decoding, we use a different map container, this time swapping the key and value as (binaries, letters). We take every 5 digits of the binary encoded string and map it to its letter counterpart using the map container, and append the decoded alphabet to the string.
4. We have created two different functions binaryEncoder which takes in input of the source code and gives an output of the encoded value, and binaryDecoder which takes in input of the encoded value and gives an output of the decoded value.

Battula Venkata Sai Ankit
2019AAPS0331H

Code:

Binary Encoder

```
function [encoded] = binaryEncoder(sourceCode)
   %Binary Encoder Encodes the sourceCode ( fixed length alphabetical code
   %to binary)
   letters =
{'a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s',
't','u','v','w','x','y','z'};
   binaries = {'00000','00001','00010','00011',...
       '00100','00101','00110','00111',...
       '01000','01001','01010','01011',...
       '01100','01101','01110','01111',...
       '10000','10001','10010','10011',...
       '10100','10101','10110','10111',...
       '11000','11001'};

   encoderMap = containers.Map(letters, binaries);
   encoded = '';
   for v = 1:strlength(sourceCode)
       encoded = append(encoded,encoderMap(sourceCode(v)));
   end
end
```

Binary Decoder

```
function [decoded] = binaryDecoder(encodedString)
   %binaryDecoder Decodes the fixed length encoded string back to
   %alphabets

   letters =
{'a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s',
't','u','v','w','x','y','z'};
   binaries = {'00000','00001','00010','00011',...
       '00100','00101','00110','00111',...
       '01000','01001','01010','01011',...
       '01100','01101','01110','01111',...
       '10000','10001','10010','10011',...
       '10100','10101','10110','10111',...
       '11000','11001'};
   decoderMap = containers.Map(binaries, letters);

   decoded = "";
   for k = 1:5:strlength(encodedString)
       code = "";
       for l = k:k+4
           code = append(code,encodedString(l));
       end
       decoded = append(decoded,decoderMap(code));
   end
end
```

Battula Venkata Sai Ankit
2019AAPS0331H

Main Code

```
clc; clear;
prompt = 'Please enter String\n';
str = input(prompt,"s");
encoded = binaryEncoder(str);
disp('Your encoded String is');
disp(encoded);
decoded = binaryDecoder(encoded);
disp('Your decoded string is');
disp(decoded);
```

Results:

```
Please enter String
>> ankit
Your encoded String is
0000001101010100100010011
Your decoded string is
ankit

Please enter String
>> ramaraju
Your encoded String is
1000100000011000000010001000000100110100
Your decoded string is
ramaraju
```

Conclusion:

Advantage of Fixed Length Encoding is that we know beforehand how many bits can be used to depict the encoded data, and it is easy to decode the encoded value at the received end. The disadvantage of fixed length coding is that we can’t detect the error and correct the error. Sometimes the encoded data takes a huge space which might not be cost efficient.

Battula Venkata Sai Ankit
2019AAPS0331H

Binary symmetric channel

Theory and Definition:

A transmitter wishes to send a bit (a zero or a one) and the receiver wishes to receive a bit in this model. With a "crossover probability" of p, the bit will be "flipped" and otherwise received correctly

Commands Used:

1. bsc: This command is used to simulate the Binary Symmetric Channel. We give the error probability as an argument.
2. biterr: This command compares the binary data between two numeric arrays, and returns the number of error bits and ratio of error bits to total bits.
3. randperm: This command is used to generate random permutation of numbers

Approach:

1. We are creating the random permutation of numbers using the randperm command and we are creating these random numbers for different sizes starting with 100 until 20,000 with step size of 100. We are including the step size inside the for loop.
2. Sending the data through a binary symmetric channel is simulated using the bsc command and we are including the error probability as an argument (here 0.7).
3. We are using the biterr command to calculate the errors occurred between the received data and source code.
4. We are storing all the error code length, total code length and ration of error code to total length values inside three matrices.
5. We are plotting all the required values accordingly from those matrices.

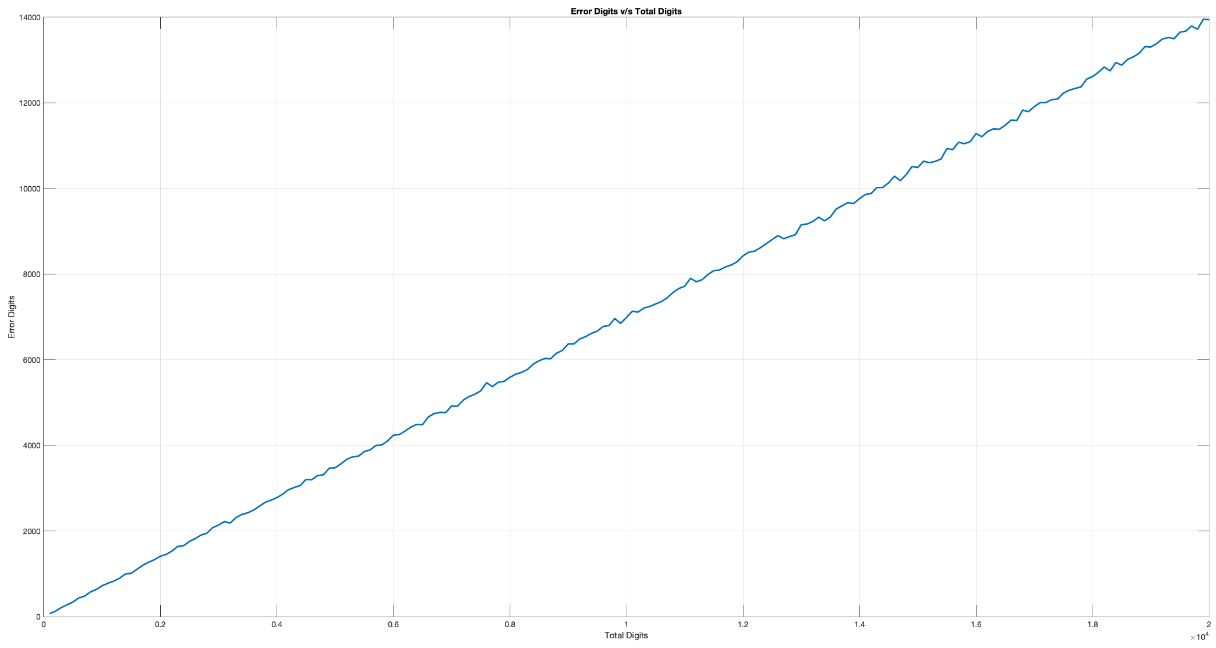
Battula Venkata Sai Ankit
2019AAPS0331H

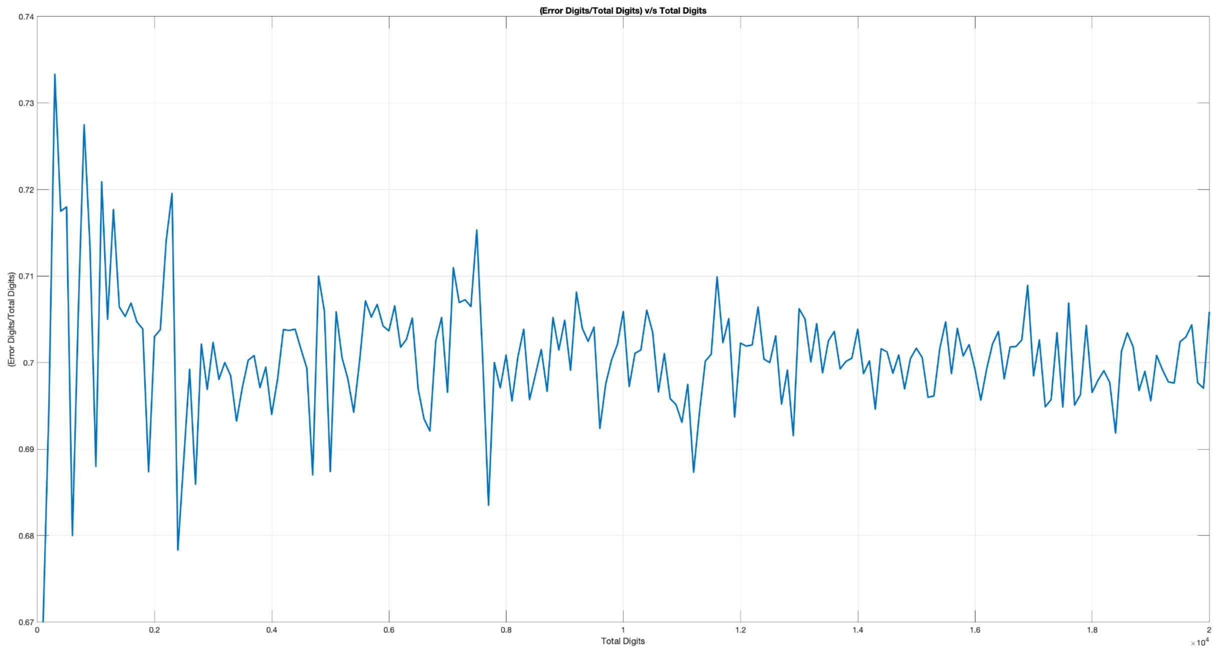
Code

```
numerr = [];
pcterr = [];
x = [];
ratio = [];
for k = 100:100:20000
   nums = mod( reshape(randperm(1*k), 1, k), 2 );
   x(end+1) = k;
   nz = bsc(nums, 0.7);
   [numerrs, pcterrs] = biterr(nums,nz);
   numerr(end+1) = numerrs;
   pcterr(end+1) = pcterrs;
   ratio(end+1) = numerrs/k;
end
f1 = figure;
plot(x,numerr);
grid on;
title('Error Digits v/s Total Digits');
xlabel('Total Digits');
ylabel('Error Digits');
f2 = figure;
plot(x,ratio);
grid on;
title('(Error Digits/Total Digits) v/s Total Digits');
xlabel('Total Digits');
ylabel('(Error Digits/Total Digits)');
```

Battula Venkata Sai Ankit
2019AAPS0331H

Results:





Conclusion:

Generally seeing the number of error bits transmitted we can see that they are increasing as the total number of transmitted bits are increasing, but when we plot the ratio of error bits to total transmitted bits we can observer that as the number of transmitted bits are increasing we can see that the ratio decreases.

Battula Venkata Sai Ankit
2019AAPS0331H

Linear Block Code + Binary Symmetric Channel

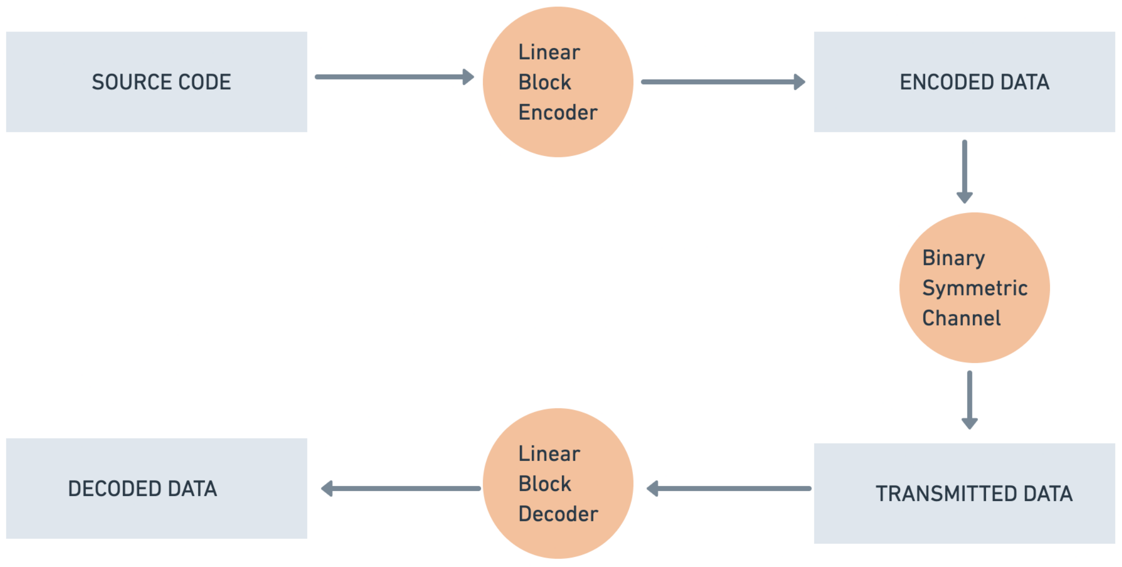
Theory and Definition:

Linear Block Coding refers to the coding of source code using a generator matrix which is G = [I : P], where P is the parity matrix. We are using encode and decode functions in MATLAB to simulate the linear block encoding. We can find the number of errors the system can

Commands Used:

1. randperm: This command is used to generate random permutation of numbers
2. encode: This command is used to encode the source using the generator matrix.
3. decode: This command aims to recover messages that were encoded using an error-correction coding technique and generator matrix.

Approach:



1. We created the random source data using randperm command of length 10000.
2. We created the generator matrix using the partiy matrix

   $$P = \begin{matrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{matrix}$$

3. We looped the 10,000 bits in steps of 3 each through a (6,3) linear block encoder simulated using encode command.
4. The encoded data is then sent through BSC with error probability of 0.3.
5. BSC introduces some errors into the encoded data. The encoded data is decoded in the steps of 6 each through (6,3) linear block decoder simulated using decode command.
6. To calculate the errors we used the h transpose and multiplied it with the encoded matrix, and calculated the syndrome to check if there is any error. We calculated the error bits and multiplied with 3 to get total error bits. Using the error bits we are calculating the performance of the system.

Code:

```
clear all; clc;
nums = mod( reshape(randperm(1*10002), 1, 10002), 2 )
P = [1 0 1;1 1 0;0 1 1];
G = [eye(3) P];
encodedMatrix = [];
for k = 1:3:10002
   m = nums(k:k+2);
   C = encode(m,6,3,'linear',G);
   encodedMatrix(end+1, :) = C;
end
H = [1 1 0 1 0 0;0 1 1 0 1 0;1 0 1 0 0 1];
ht = transpose(H);
error_count = 0;
synMatrix = encodedMatrix * ht;
check = [0 0 0];
for i = 1: length(synMatrix)
   if(synMatrix(i:i+2)~=check)
       error_count = error_count + 1;
   end
end
nz = bsc(encodedMatrix, 0.3);
decodedMatrix = [];
for k = 1:6:20004
   c = nz(k:k+5);
   D = decode(c,6,3,'linear',G);
   decodedMatrix(end+1, :) = D;
end
decodedMatrix = transpose(decodedMatrix(:));
[numerrs, pcterrs] = biterr(nums,decodedMatrix);
disp('Total number of errors corrected by the Linear Block Decoder are: ')
disp(error_count * 3);
disp('Ratio of number of errors corrected to total bits transmitted: ')
disp((error_count * 3/10002)*100);
disp('Total Error Bits after decoding the data from BSC: ')
disp(numerrs);
```

Battula Venkata Sai Ankit
2019AAPS0331H

Results:

Total number of errors corrected by the Linear Block Decoder are: 4047
Ratio of number of errors corrected to total bits transmitted: 40.4619
Total Error Bits after decoding the data from BSC: 4970

Conclusion:

Linear Block Decoder can correct 4047 errors according to this case, and there are other errors introduced by the Binary Symmetric channel, which are 2 bit errors that cannot be corrected by the Linear Block decoder. The total error bits after decoding are 4970 from the data that arrived from BSC, that means there are some errors introduced by the BSC decoder cannot correct. The performance of the system is ratio of number of bits correctly transmitted from the decoder, which is 100 - 40.4619 = 59.5381 %