Karim Eltahawy

651537986

5/10/2019

ECE 311

Mini Project

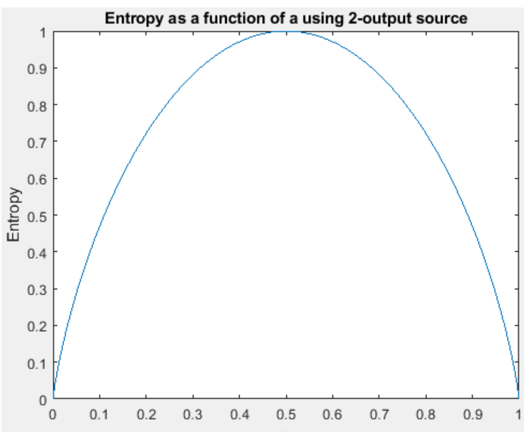
Objectives:

Part 1: To build and test a Huffman code by using a .txt file as the input signal and then derive the probability distribution of the symbols used in the input. Then, derive the codewords of the output symbols using the Huffman encoder. Finally, evaluate the compression rate.

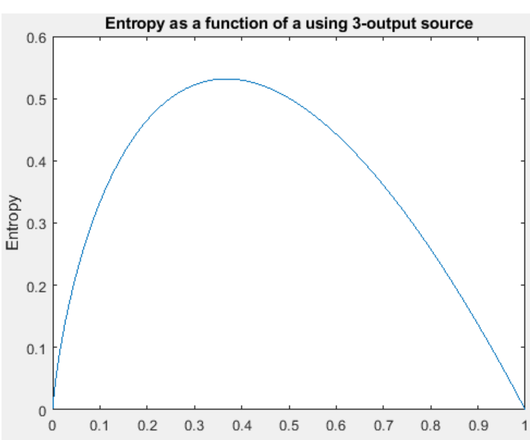
Part 2: To investigate the properties of a simple channel code (linear block code) using MATLAB. Then, encode and decode a long binary sequence with and without transmission errors.

Pre-Lab:

Observing the maximum source entropy when the source outputs are equally likely. In this case, a two output source is used with respective probabilities [a; 1-a].



Three output source with respective probabilities [a; b; 1 - a - b].



<u>Task:</u>

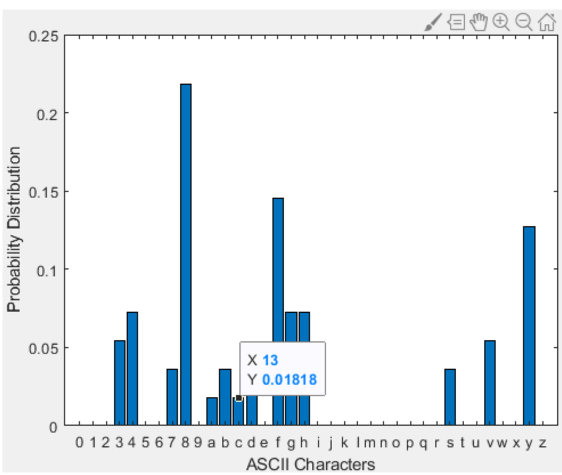
Part 1:

By adding the “count_letters” and “count_numbers” matrices together into “counts”, we can not only see how many different characters there are but also see how many times each character is shown.

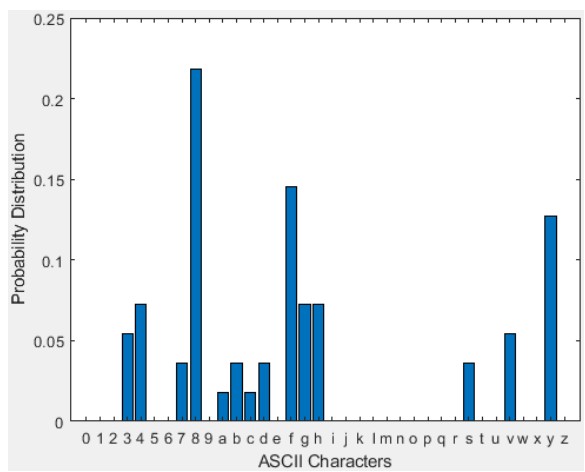
```
counts =

  Columns 1 through 18

     0     0     0     3     4     0     0     2    12     0     1     2     1     2     0     8     4     4

  Columns 19 through 36

     0     0     0     0     0     0     0     0     0     0     2     0     0     3     0     0     7     0
```

To find the number of bits needed per character if we use code with the same length for all characters, we find the lowest probability (as shown below in the graph), 0.01818, then we take -$\log_2$(lowest-probability), which in this case is -$\log_2$(0.01818) = 6. B should be 4 bits because the lowest probability bit is 0.01818. So, we need at least 6 bits for a codeword, given that probability using Shannon code. When using Huffman code, however, the encoding number of bits needed is one-less so we get B = 5 bits.

Achieving the lowest probability:



Original Graph:



Part 2:

Before channel coding, the length of the transmission sequence is 192x1 and after channel coding the length is 336x1. The relationship between the before and after transmission sequences is through this equation: $length(before) \cdot \left(\frac{n}{k}\right) = after$. In this case, it would be $192 \cdot (\frac{7}{4}) = 336$. To correct errors in the code, the biterr function in MATLAB indicates how many bit errors are in the transmission sequence and at max, the code can fix 7 consecutive errors. The “numError” variable indicates the number of errors there are.