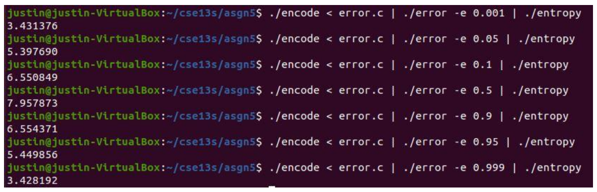
asgn5 Writeup

Graphs

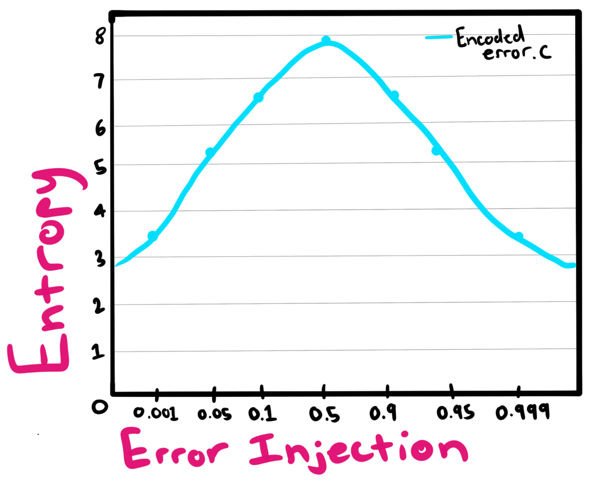
As I am unfamiliar with gnuplot, I decided to test several outputs in the terminal, then create my own representation of those outputs.

Entropy quantifies the amount of information created by a process. When running entropy with the encoding of different files, there is a clear variation between them. The variation becomes increasingly more thin depending on certain inputs of error injection. These variations and similarities are described through the two terminal input and graph representations below.

Graph 1: Encoded error.c

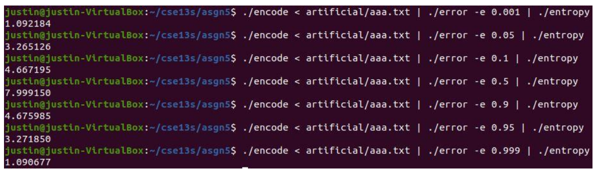


```
justin@justin-VirtualBox:~/cse13s/asgn5$ ./encode < error.c | ./error -e 0.001 | ./entropy
3.431376
justin@justin-VirtualBox:~/cse13s/asgn5$ ./encode < error.c | ./error -e 0.05 | ./entropy
5.397690
justin@justin-VirtualBox:~/cse13s/asgn5$ ./encode < error.c | ./error -e 0.1 | ./entropy
6.550849
justin@justin-VirtualBox:~/cse13s/asgn5$ ./encode < error.c | ./error -e 0.5 | ./entropy
7.957873
justin@justin-VirtualBox:~/cse13s/asgn5$ ./encode < error.c | ./error -e 0.9 | ./entropy
6.554371
justin@justin-VirtualBox:~/cse13s/asgn5$ ./encode < error.c | ./error -e 0.95 | ./entropy
5.449856
justin@justin-VirtualBox:~/cse13s/asgn5$ ./encode < error.c | ./error -e 0.999 | ./entropy
3.428192
```

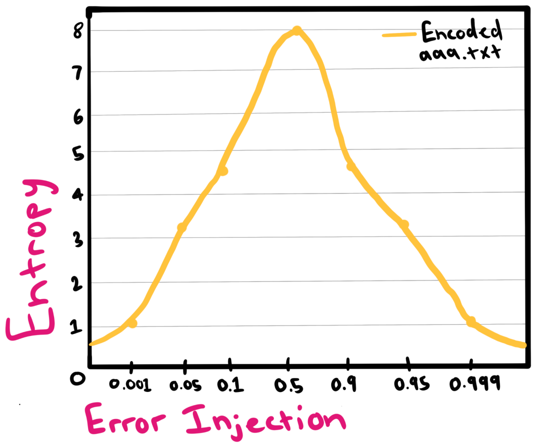


This graph was produced from encoding error.c with various error injection percentages. Entropy was then run on it to get readable data. From these two images, we are able to see that as we increase the amount of error injected into the encoding up to 50%, the amount of bits being flipped (information being created) increases. However, once the error injection rate starts to near the 100% extrema, the entropy levels become similar to the 0% extrema numbers. The reason this happens is because so many errors are being injected into the encoded file that they begin to self correct!

Graph 2: Encoded aaa.txt



```
justin@justin-VirtualBox:~/cse13s/asgn5$ ./encode < artificial/aaa.txt | ./error -e 0.001 | ./entropy
1.092184
justin@justin-VirtualBox:~/cse13s/asgn5$ ./encode < artificial/aaa.txt | ./error -e 0.05 | ./entropy
3.265126
justin@justin-VirtualBox:~/cse13s/asgn5$ ./encode < artificial/aaa.txt | ./error -e 0.1 | ./entropy
4.667195
justin@justin-VirtualBox:~/cse13s/asgn5$ ./encode < artificial/aaa.txt | ./error -e 0.5 | ./entropy
7.999150
justin@justin-VirtualBox:~/cse13s/asgn5$ ./encode < artificial/aaa.txt | ./error -e 0.9 | ./entropy
4.675985
justin@justin-VirtualBox:~/cse13s/asgn5$ ./encode < artificial/aaa.txt | ./error -e 0.95 | ./entropy
3.271850
justin@justin-VirtualBox:~/cse13s/asgn5$ ./encode < artificial/aaa.txt | ./error -e 0.999 | ./entropy
1.090677
```



This graph was produced from encoding aaa.txt with various error injection percentages. Entropy was then run on it to get readable data. This graph is quite similar to the previous graph. When comparing the two, we can see that as we get closer to the 50% injection percentage, the entropy created is extremely close. On the other hand the closer we get to each extrema, the more different the graphs are from one another. The reason behind this is due to the original messages the files contain. The file “aaa.txt” contains 200,000 “a” characters, therefore does not have much variation when encoding with a low error injection. This leaves us with a lower entropy data number due to the lack of information being created. When we get to 50% injection percentage, there are just as many encoded errors as there aren't. Because of this, no matter what file you attempt to run this on, the 50% entropy data number will always reach around the same number!