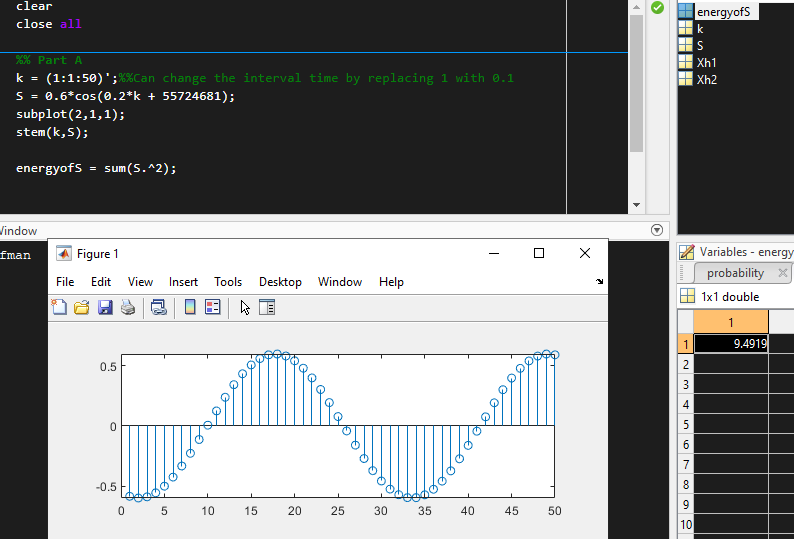
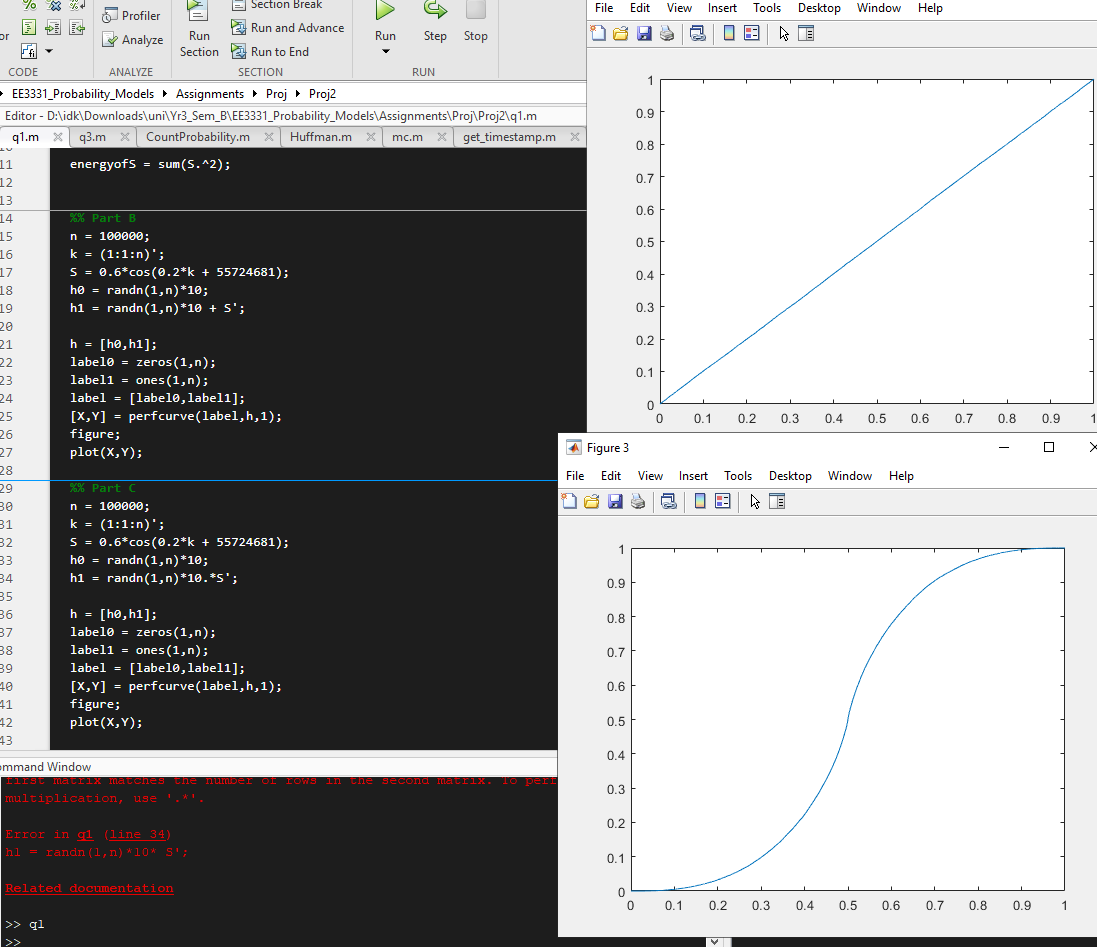
1a)



b) & c)

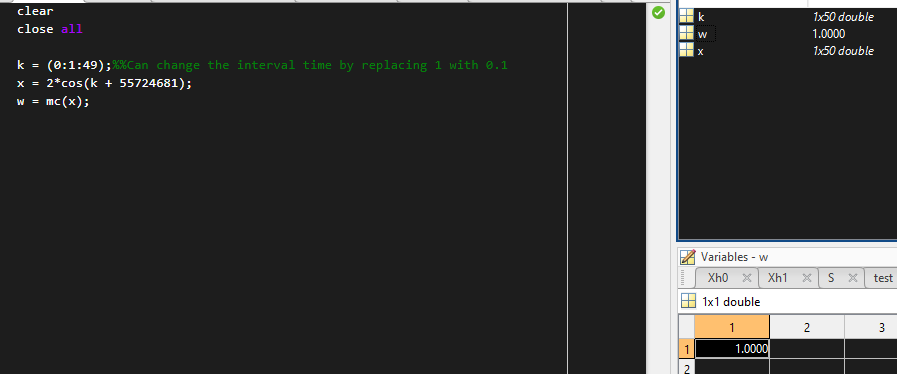


With the Y as Pd and X as Pfa, we can see that at the top figure using the instructions from part b) is a linear graph where as the bottom figure using the instructions from part c) has a more exponential and logarithmic graph.

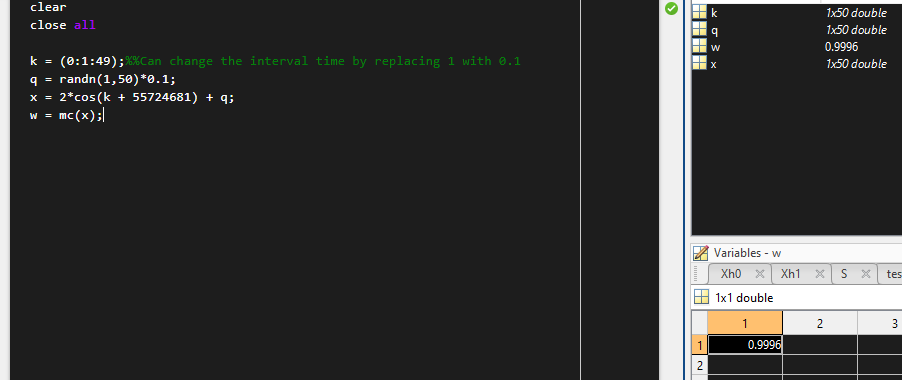
I think both detectors has the same performance as both graphs shows that they have very similar area coverage under the line.

2a)

The purpose is to keep preventing the inverse cosine function from going out of bounds.

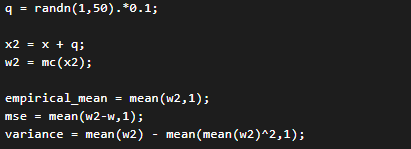
b)

w’s result is a 1.

c) 

w has a result of 0.9996

d)

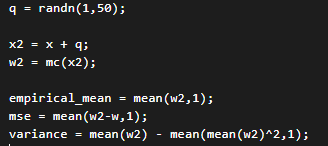


Empirical mean is 0.9997

Mse is -2.7098e-04

Variance is 2.7098e-04

e)



Empirical mean is 1.1763

Mse is 0.1763

Variance is -0.20743

f) The MSE and variance are correlated in d), but in e) it is very different.

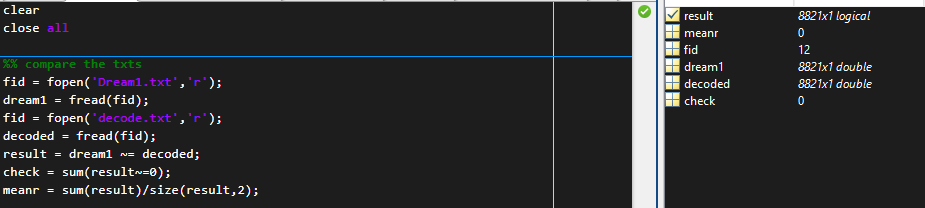
g)  

Using matlab to calculate it instead of using the estimator, we can see that it is equal or close to the what matlab calculated. So it is an indicator of an unbiased estimator, but from e) and d) we can see that the MSE is not equals to the variance, therefore MC is a biased estimator.

3a)

The ‘alphabet’ array represents the list of characters that appeared in the text and is sorted by the highest probability stored in ‘probability’ array which also included the whitespace.  
While ‘dict’ is the Huffman code dictionary which acts as a translator for which char will be translated to which numbers such as E into “110”, etc. which is generated by using the maximum variance algorithm.

b)

Essentially no difference as Huffman encoding is a lossless encoding compression method. As shown in the matlab code, check has a value of zero meaning that there is no difference from the original txt to the decoded txt.

c) 

Using the function ‘mode’ to find the max count in dream1, we can get the ascii of the char of 32 that appeared the most times, which is ‘ ‘ (just white space). Which is a given, so I decided to calculate the all counts of occurring characters by using groupcounts().

 where GC is the count the character appeared and GR is the character the count is tied to.

We get the second highest ascii char of 101 which is the alphabet ‘e’.

Then we can compute each of the character’s probabilities.





d)





e)