Report:

We present below the analysis of four different cases of HMMs to decipher some encrypted text. The first case is a standard Hidden Markov Model implementation; we add Laplace smoothing in the second case (`-laplace` flag); the third case consists of loading some pre-computed state transition probabilities (`-lm` flag); and the last case represents Laplace smoothing along with the pre-computed state transition probabilities (`-laplace -lm` flags).
We test these four cases on two different ciphers: the first one (cipher1) is a regular character substitution cipher, while the second one (cipher2) is a random combination of two character substitution ciphers.

Accuracy (%)	Cipher1	Cipher2
`python decipher.py cipher<1|2>`	09.87 %	14.98 %
`python decipher.py -laplace cipher<1|2>`	97.66 %	83.12 %
`python decipher.py -lm cipher<1|2>`	68.40 %	60.78 %
`python decipher.py -laplace -lm cipher<1|2>`	**98.27 %**	**87.79 %**

Table01: accuracy of the four HMM versions with two different ciphers.

First of all we would expect the score of cipher2 to be generally lower than cipher1 because it is a more complex cipher, in which there are two letter substitution ciphers used at random times, thus more complex to learn and to decrypt. However, we surprisingly note that the accuracy score of cipher2 is bigger than the one for cipher1 in the case of standard HMM learning. This may be caused by the fact that the standard learning method is in fact so bad, that it is close to random, and randomness happens to predict better the second type of cipher which has a random component in it.
Moreover, we can say that each step helped improving the accuracy score of our HMM. After adding Laplace smoothing the score already made a big jump almost close to the final best score we will get. This shows that a lot of unseen components were present in the test set. It is also probably due to the very small training set which made the ability to generalise quite poor without any smoothing. Further more, adding transition probabilities from the corpus seen in Assignment1 also increased a lot the accuracy score of the standard HMM. This is expected as the data seen in Assignment1 is bigger than the current training set, thus making the transition function better in this model. However, the emission probabilities and the number of unseen components remain the same. We observe that the accuracy is lower than when we added laplace smoothing, probably because of the incapacity of generalizing well due to a lot of unseen words in the test set.
Eventually, combining both laplace smoothing and pre-trained transition probabilities improved the accuracy of both ciphers to a maximum score. In the case of cipher1 we almost reach perfect accuracy with a score of 98.27%, while cipher2 decription accurary reached 87.79%. As previously said this is due to the fact that cipher2 is more complex to learn.