APMA1941A - Project 1 **Junewoo Park**

0. About

This project uses Markov chain Monte Carlo to descramble given scrambled texts. I implemented a descramble algorithm using the bigram model (and the monogram model) of English in a Python (v 3.7.4) file, textdescramble.py. First, my program receives a training text file and obtains approximation to the true distribution of bigrams and monograms. Next, using Metropolis walk, the program searches for the permutation that maximizes the probability of the descrambled text. Refer to the comments in the textdescramble.py file for more detail.

1. Usage

```
Usage: textdescramble.py [trainfile] [input] [output] [bigram/monogram] [num_iter]
```

[trainfile] is the filename of the text file needed for training the model.
[input] is the filename of the scrambled text. I used “war-and-peace.txt”
[output] is the filename of the descrambled text user would like to create.
[bigram/monogram] is where the user specifies which model the program should use.
[num_iter] is the number of Metropolis walk.

2. Remark

When obtaining the energy difference of two permutations in Metropolis walk, the numerators of the approximated probability distributions are cancelled out. Thus, my program does not normalize the numbers of appearances of bigrams/monograms when training the bi/mono-gram distributions. In other words, bigram_data and monogram_data in textdescramble.py only count the numbers of bigrams and monograms, respectively.

Also, since the training text file is sufficiently long, I decided to use 0 by default (instead of $-\infty$) for the logarithms of counts of the bigrams that do not appear in the train file. This is equivalent to using $-\log(\text{Length of the training text})$, instead of $-\infty$, for the log conditional probability of bigrams that do not appear in the training file.

3. Files

textdescramble.py: Python file that trains and applies the bi/mono-gram model of English.

war-and-peace.txt: The text file used to train the model.

p_student.txt: The scrambled text file. I manually removed the first line “The coded text:” before feeding it to textdescramble.py

p_descrambled.txt: The descrambled text file of p_student.txt

p_log.txt: Log file for descrambling p_student.txt rrr_student.txt: The scrambled text file. I manually removed the first line "The coded text:" before feeding it to textdescramble.py

rrr_descrambled.txt: The descrambled text file of rrr_student.txt

rrr_log.txt: Log file for descrambling rrr_student.txt

zzz_student.txt: The scrambled text file. I manually removed the first line "The coded text:" before feeding it to textdescramble.py

zzz_descrambled.txt: The descrambled text file of zzz_student.txt obtained by applying textdescramble.py

zzz_descrambled_final.txt: The final descrambled text file of zzz_student.txt obtained by manually modifying some errors in zzz_descrambled.txt

zzz_log.txt: Log file for descrambling zzz_student.txt

README.pdf

Note: Descrambling of zzz_student.txt was difficult since the its length was not sufficiently long. I observed instability and convergence to local minimum frequently, so it took many runs to get zzz_descrambled.txt.