Report Part：

In this section, we evaluate the features of our improved solution. Because of the lack of diverse data sets, we can not evaluate performance as in the paper; so we simplify the experiment here and try to explore correlations between various parameters as much as possible for its reasons. Here, we integrate data gained from experiments and use Python's Matplotlib library for visual rendering.

We implement all methods in Python and conduct experiments on a desktop computer with Intel Core i5 1.4GHz CPU, 8G Memory running Mac operating system. The testing dataset consists of only synthetic ones and their values are restrict in the limited range that fits the reality.

**Parameter settings.** In experiments, we change parameters ***u***(size of user space)and**𝝐** (probability of random response mechanism) respectively in order to qualitatively explore the impact on RE. In each experiment, the range of ***u*** is {10 ^ 6, 2x10 ^ 6, 4x10 ^ 6, 8x10 ^ 6}, the range of **𝝐** is {0.1, 0.2, 0.4, 0.8} and ***k*** (options in real situations) is set to the constant value of 5.

**Datasets.** In order to be close to the actual scene, the dataset we generate in each test is random but similar so as to ensure the fidelity and randomness between tests. Each user's data is only one of the options in the current situation. In order to be processed in the improved algorithm ***PrivKVD***, it needs to be mapped into key-value pairs first.

To measure the accuracy of the estimated frequency, we use the following metric that are widely used in the literature.

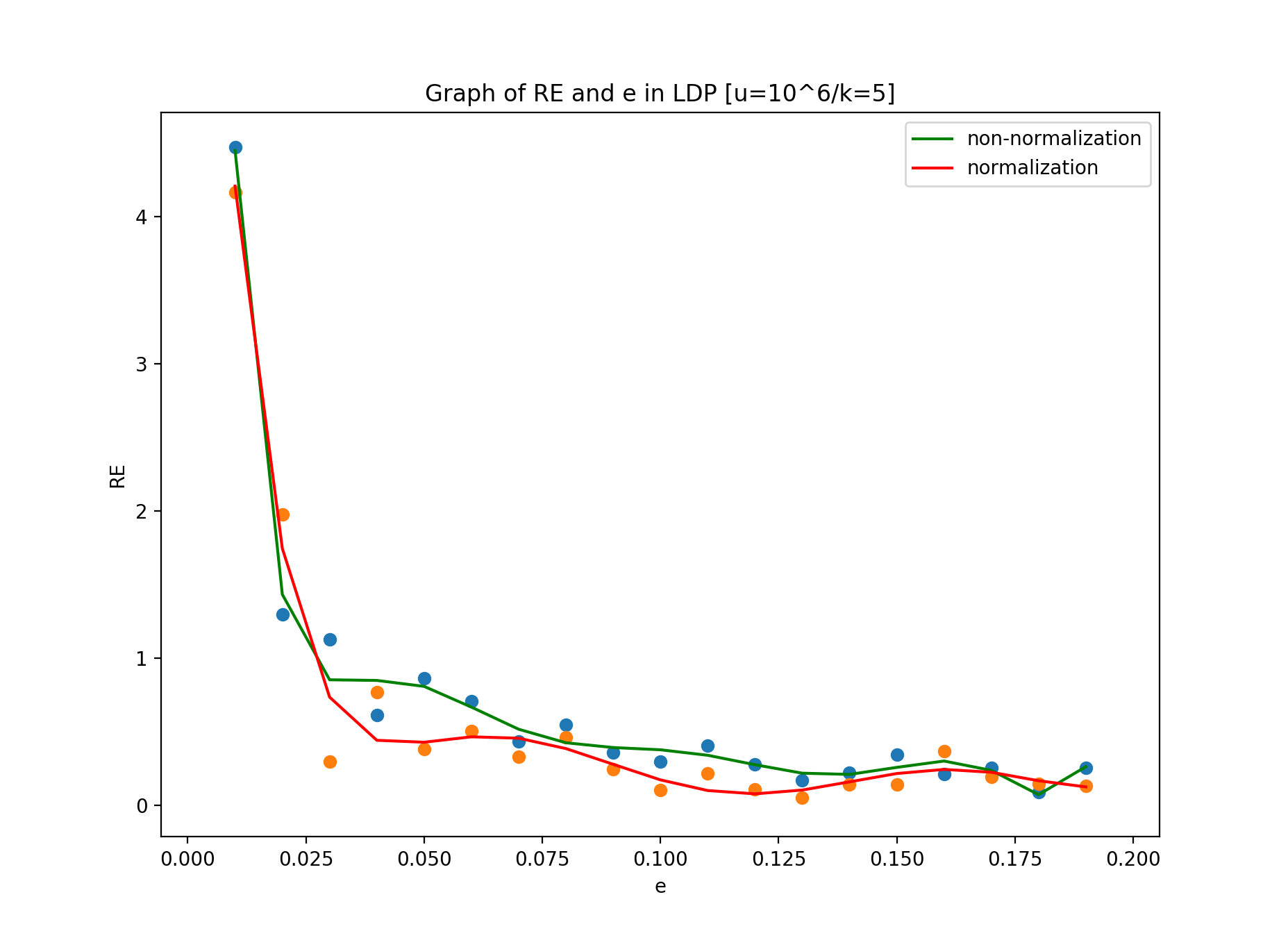
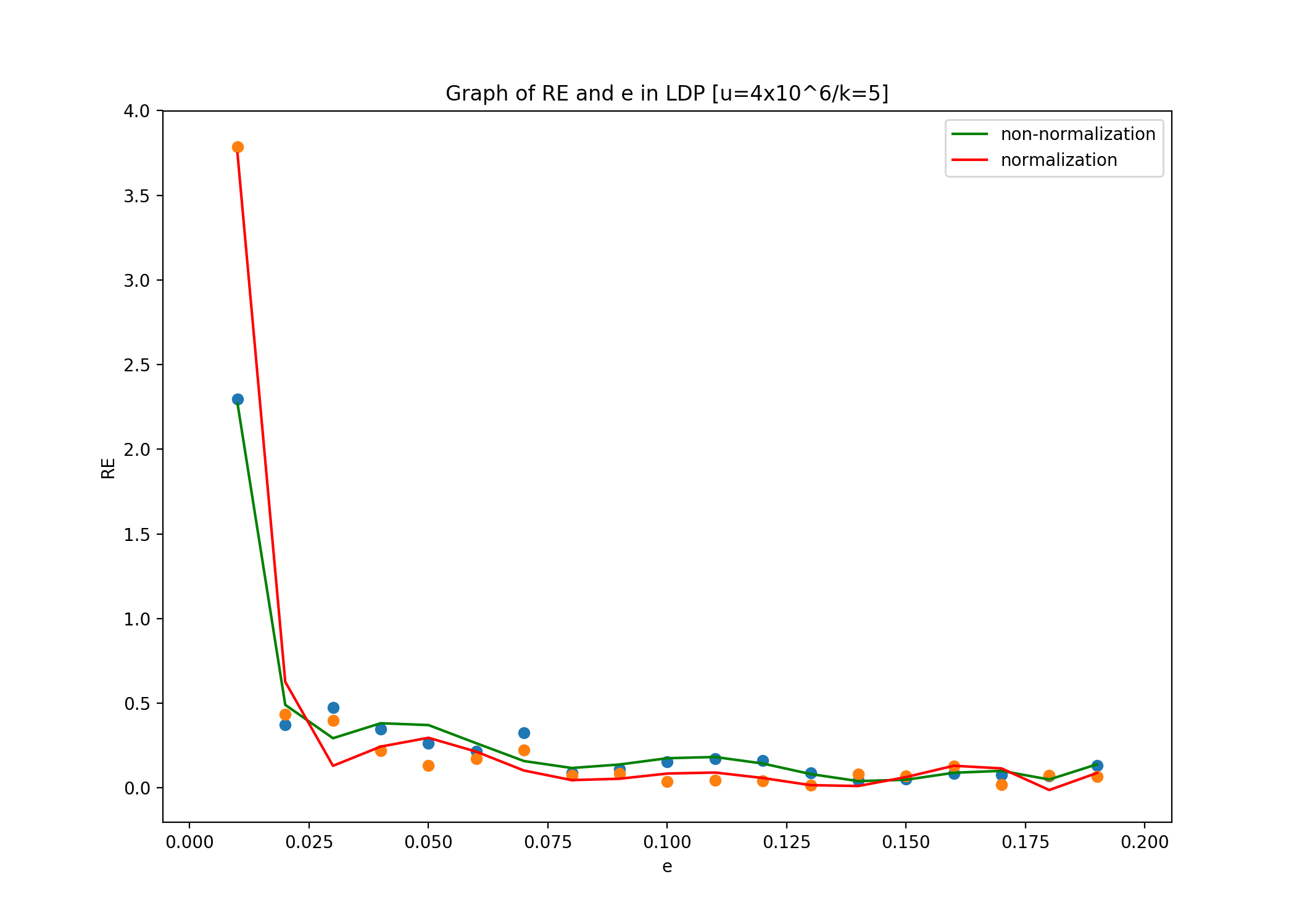
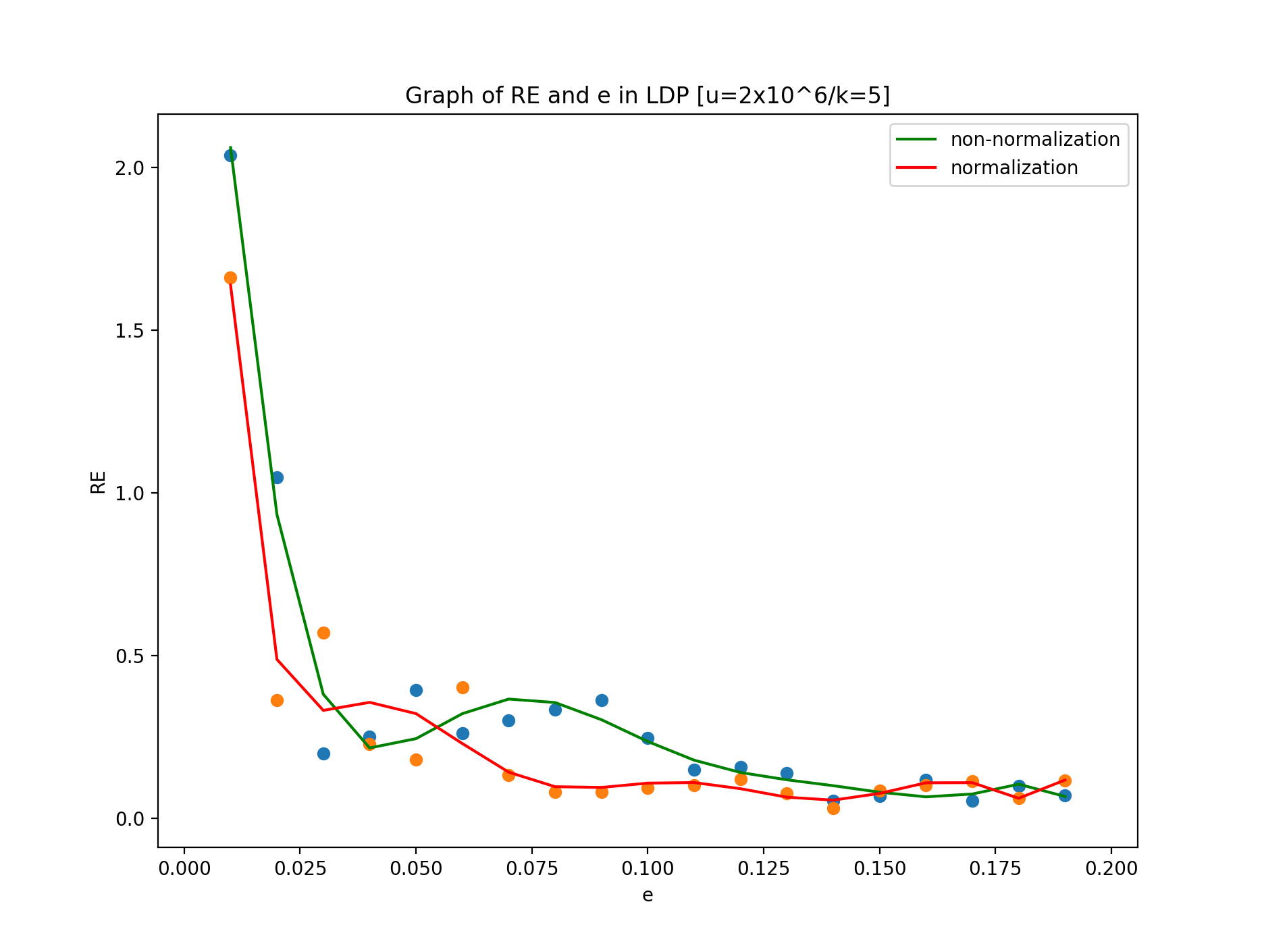
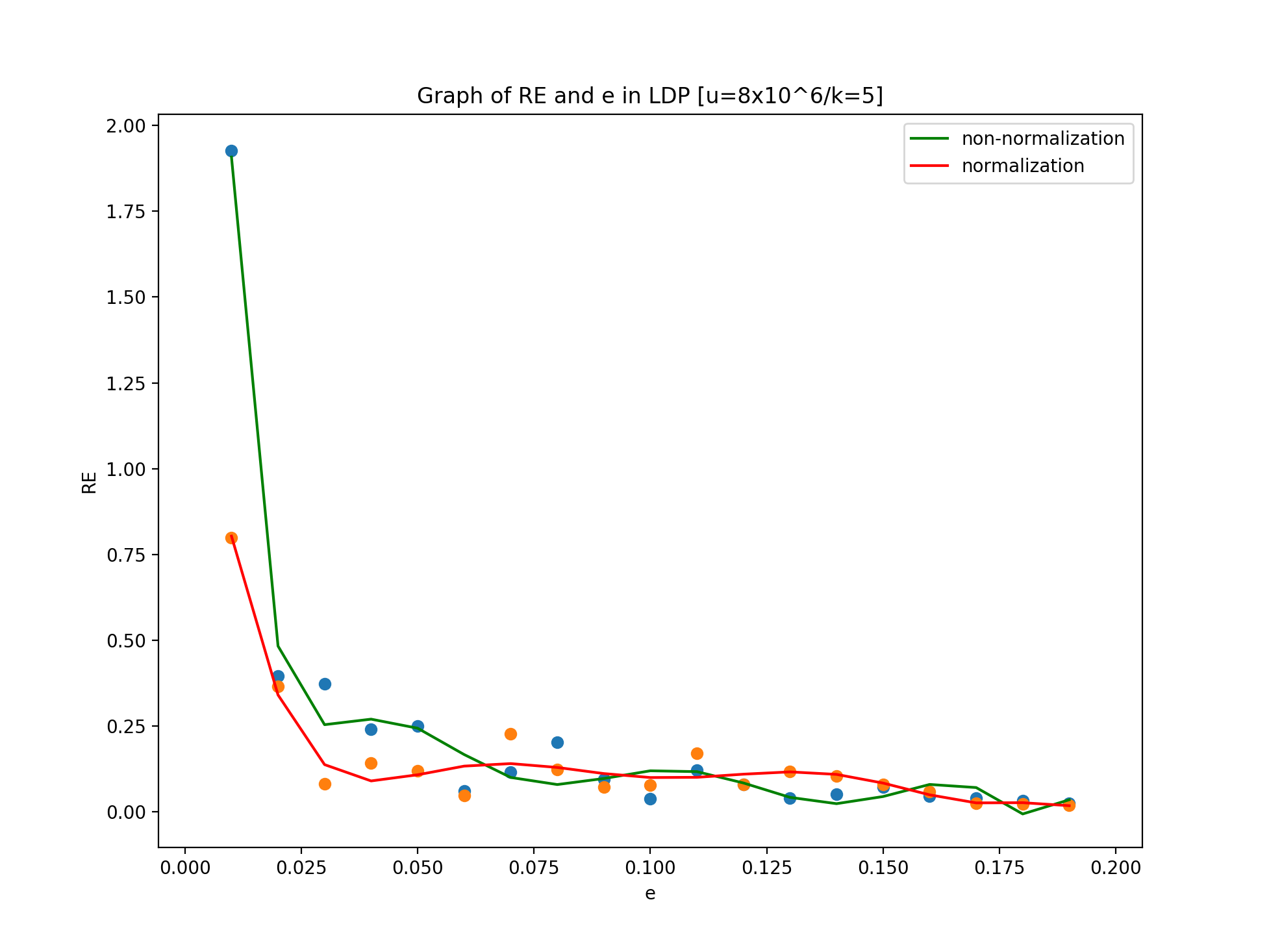
* Relative Error (RE). It measures the relative error of estimated frequencies with respect to real frequencies of all keys. Specifically, for key k ∈ K, let fk and fk\* denote real and estimated frequency, respectively. Here, keys equal to options in real situations.

[公式]

where Stat is a statistical function such as median (default).

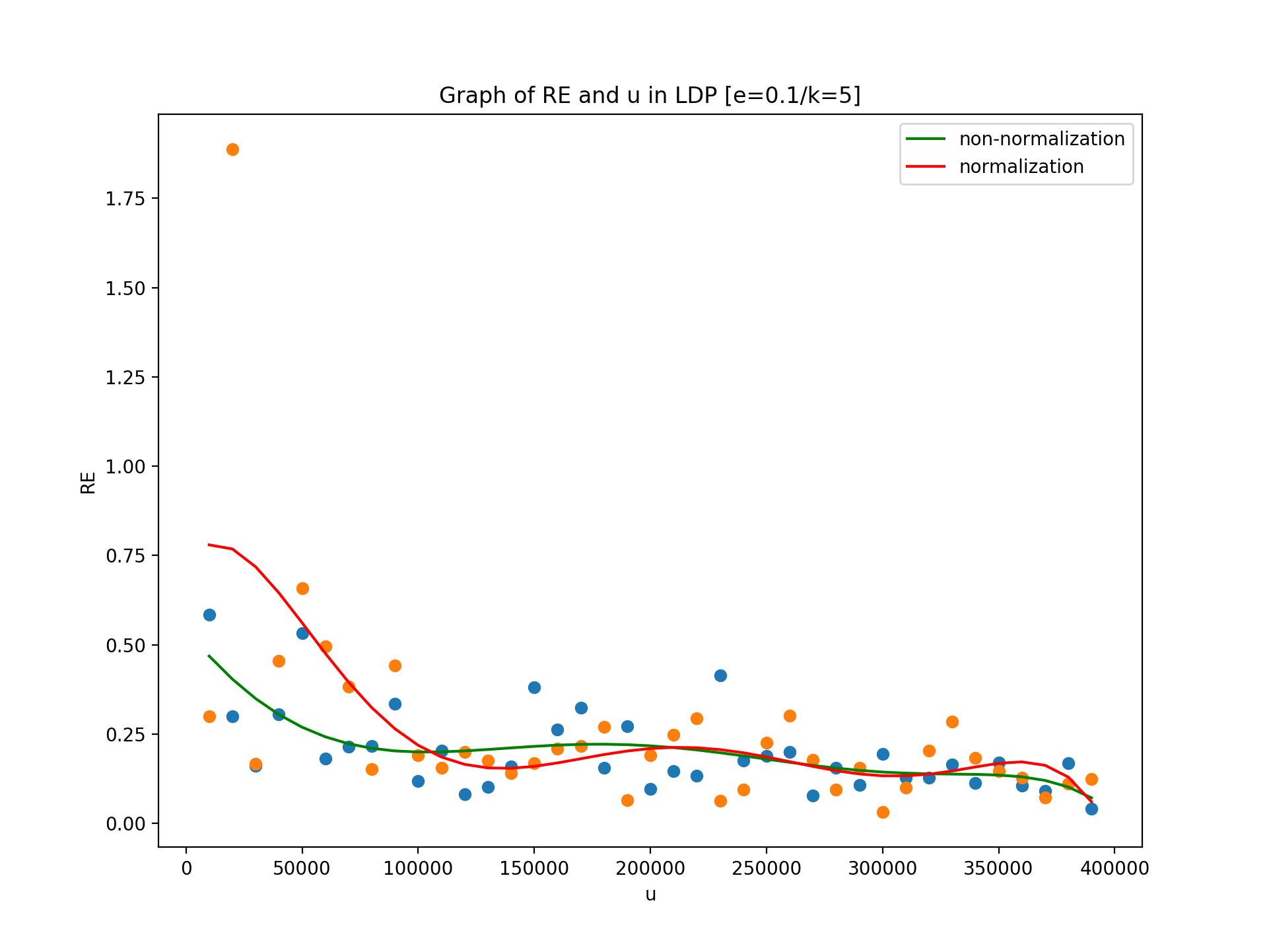
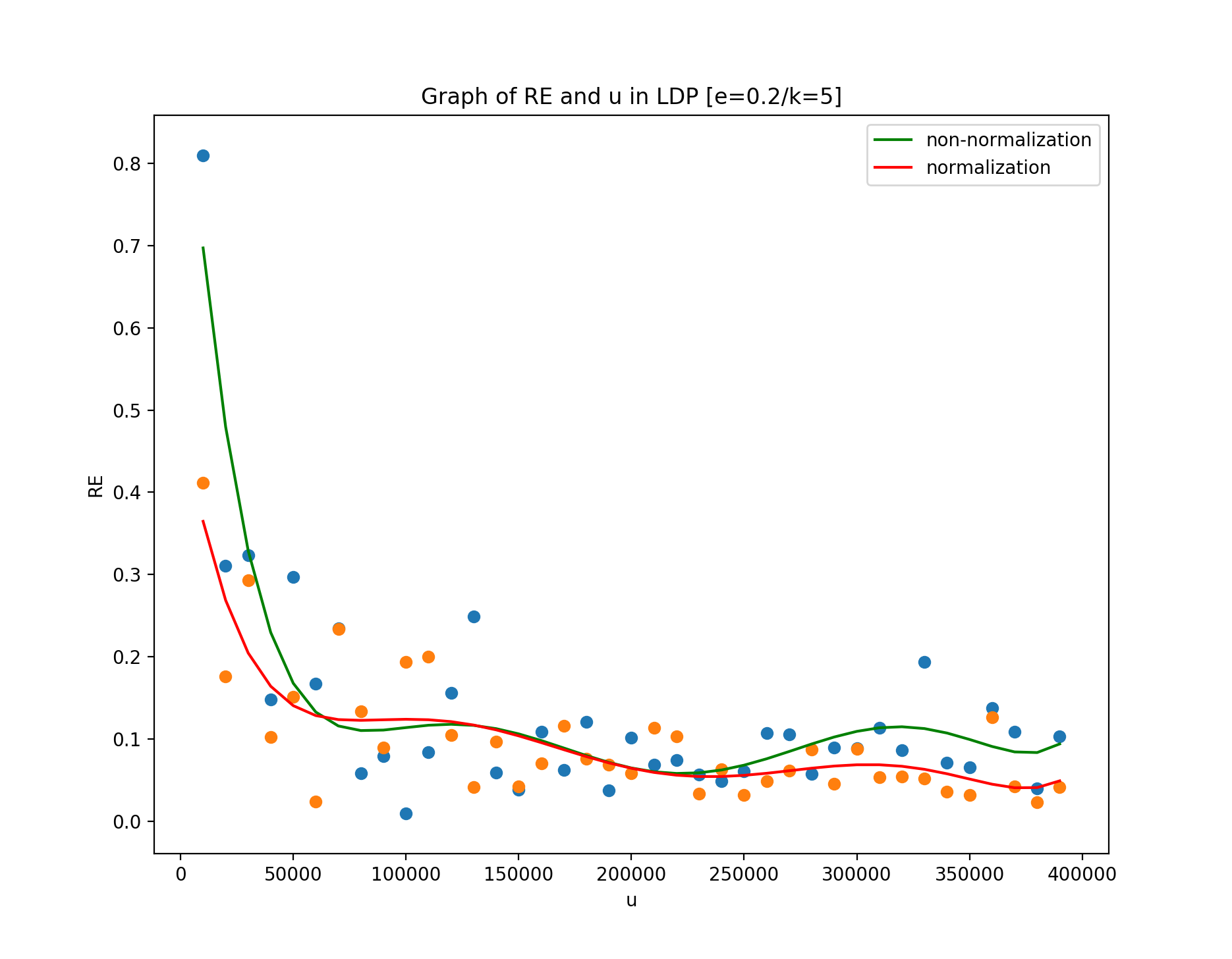
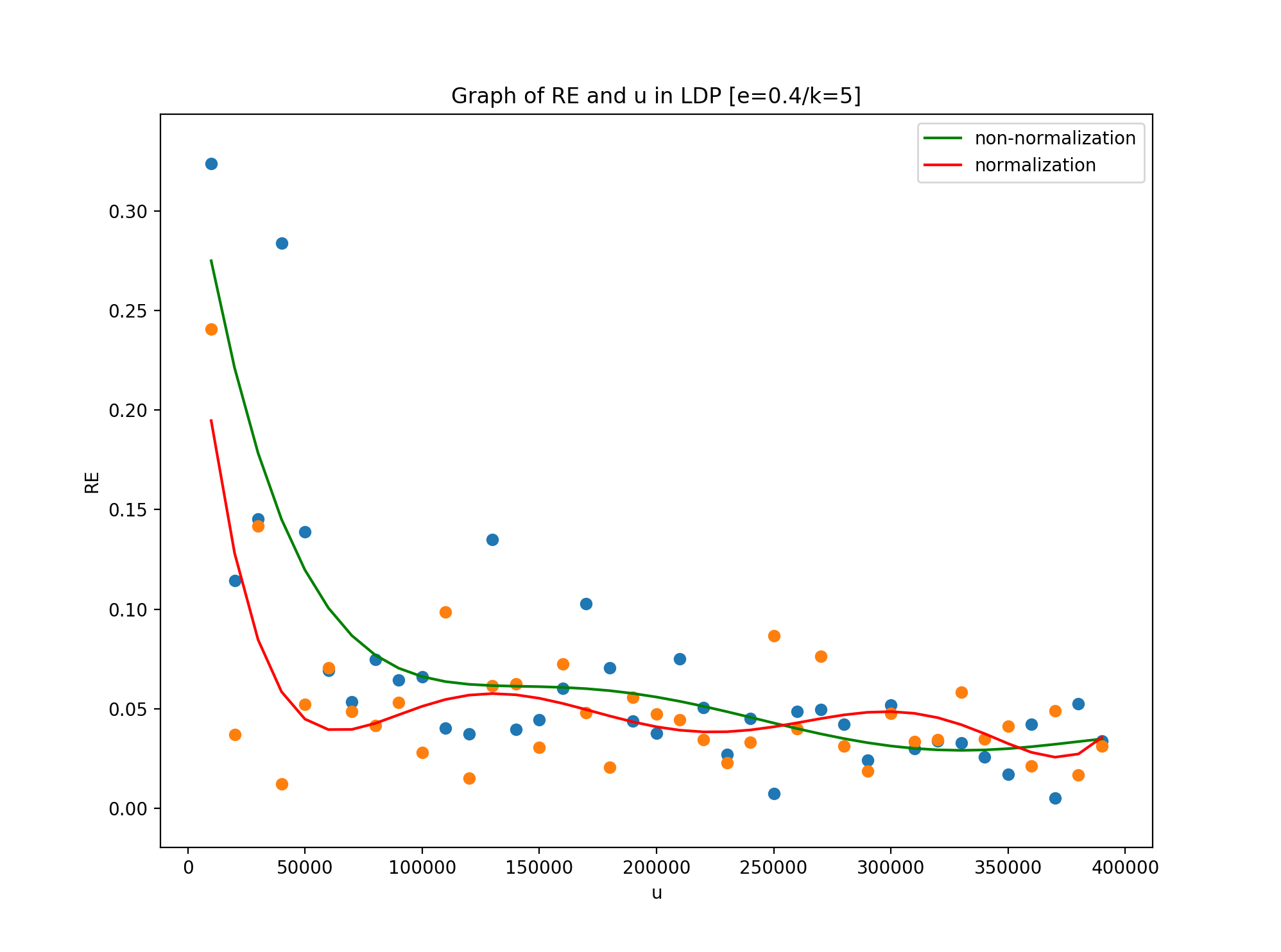
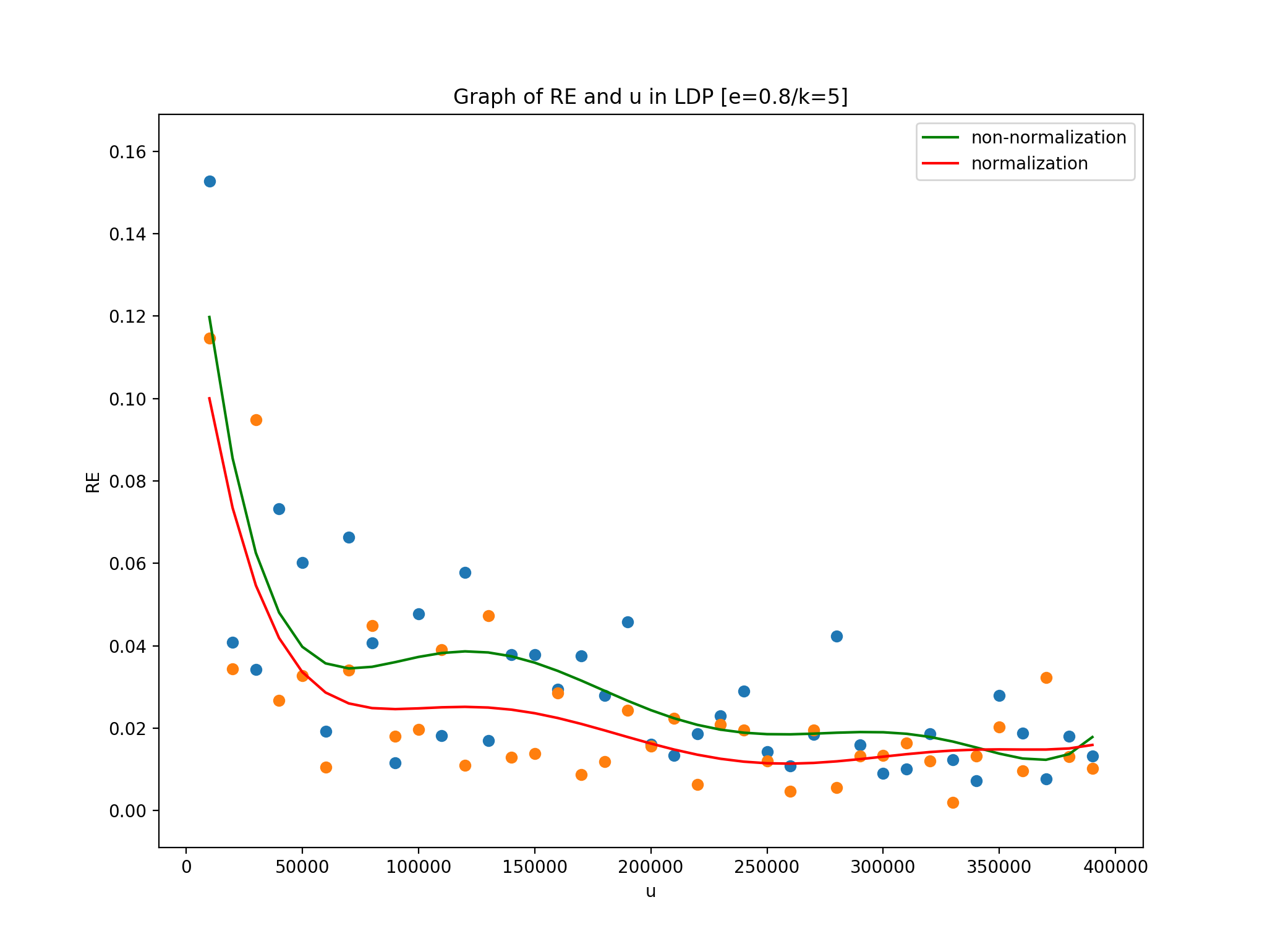
**A. 𝝐 and RE**

Formula fk\*=(p-1+fk\*)/(2\*p-1) in PrivKVD where p=e^𝝐/ (e^𝝐+1) shows that when 𝝐 tends to 0, P tends to 1/2, the denominator of the formula tends to 0, leading to fk\* tends to infinity, which will cause a large difference between it and the real frequency estimation; when 𝝐 tends to infinity, p will be close to 1, the whole formula tends to fk\* itself, leading to the weakening of the correction effect, and also making a large deviation from the real frequency estimation. Therefore, as shown in the figure, fk\* will be properly corrected and close to the true value only when 𝝐 increases properly.

**B. *u* and RE**

As shown in the figure, when the user space u is small, the correlation between different users is weak and it is difficult to reflect the statistical characteristics as a whole so the evaluation difference of frequency estimation fk\* will be larger; With the growth of user space, as the whole of the data set, it highlights the statistical characteristics, and its frequency estimation is closer to the real value, so that RE will decline. In addition, we can also know from the graph that an appropriate increase of 𝝐 will help to get a more accurate estimate. The gain improves with increasing user sizes due to the law of large numbers. When this size reaches 10^5, the accuracy is high enough for most data collection tasks.

**C. Time comparison in different LDP**

In this experiment, we compare the efficiency of four versions of LDP algorithm, ***PrivKVD***, ***PrivKV***, ***PrivKVM*** and ***PrivKVM+***. The parameter settings are shown in the figure. According to the previous experiments, we can know that the accuracy is high enough for data collectors when **𝝐** reaches 0.2. Therefore, ***u***=10^3 and **𝝐**=0.2 are set here while ***k*** continues the value of the previous experiments.

On the whole, the overhead of algorithm ***PrivKVD*** and ***PrivKV*** is the smallest and close to each other; the overhead of ***PrivKVM*** is slightly higher than that of the former ones because it calibrates the calculated data through a loop; and the overhead of ***PrivKVM+*** is much higher because it contains adaptive calibrations and needs to constantly calculate the difference between the accuracy cost and the communication cost to determine whether to end the loop. In addition, due to the internal parameter settings and control flow of the first three algorithms basically unchanged, their overheads are relatively stable. However, the number of adaptive calibrations is different in ***PrivKVM+*** for different user data sets, so its overhead has relatively large fluctuations.

