HW1

1. Shingling of Documents.

(1) The python code: see file. There are 1911 2-shingles. The 2-shingles are saved sorted according to the frequency.

(2) There are 10602 2-shingles. The 3-shingles are saved sorted according to the frequency.

(3) The results:

1.1.3_result.txt

```
The # of 1-shingles is 63
The # of 2-shingles is 1238
The # of 3-shingles is 8370
The # of 4-shingles is 29222
The # of 5-shingles is 66247
The # of 6-shingles is 113561
The # of 7-shingles is 162200
The # of 8-shingles is 207151
The # of 9-shingles is 243970
The # of 10-shingles is 271408
```

(4) Theoretically, assuming a document has n characters and $n \geq k$ (there exists at least one shingle), there are $n - k + 1$ shingles if all shingles are unique. All shingles can be unique if and only if $n - k + 1 \leq 36^k$. Then in the worst case, we have $\min(n - k + 1, 36^k)$ shingles, and each shingle takes k bytes to sore, then the space need is $\min(k(n - k + 1),\ k36^k)$ in the worst case.
To store each character we need 1 byte, so for each *k*-shingle, it costs *k* byte(s) to store, in the worst case. The space needed for the set of *k*-shingles is:

1.1.4_result.txt

```
The space taken for 1-shingles is 63 bytes.
The space taken for 2-shingles is 2476 bytes.
The space taken for 3-shingles is 25110 bytes.
The space taken for 4-shingles is 116888 bytes.
The space taken for 5-shingles is 331235 bytes.
The space taken for 6-shingles is 681366 bytes.
The space taken for 7-shingles is 1135400 bytes.
The space taken for 8-shingles is 1657208 bytes.
The space taken for 9-shingles is 2195730 bytes.
The space taken for 10-shingles is 2714080 bytes.
```

(5) The results are:

1.1.5a_result.txt

```
The # of 1-shingles is 63
The # of 2-shingles is 1238
The # of 3-shingles is 8595
The # of 4-shingles is 30732
The # of 5-shingles is 70444
The # of 6-shingles is 122523
The # of 7-shingles is 177721
The # of 8-shingles is 230859
The # of 9-shingles is 276895
The # of 10-shingles is 313889
```

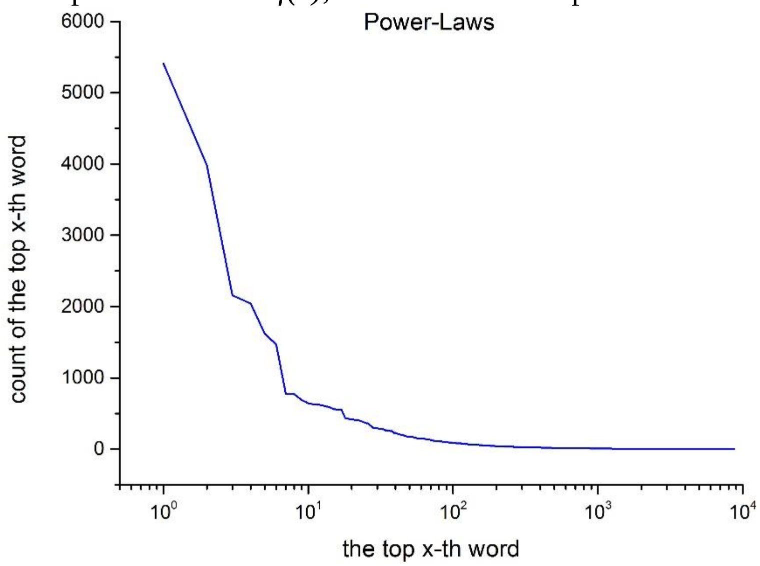
1.1.5b_result.txt

```
The space taken for 1-shingles is 63 bytes.
The space taken for 2-shingles is 2476 bytes.
The space taken for 3-shingles is 25785 bytes.
The space taken for 4-shingles is 122928 bytes.
The space taken for 5-shingles is 352220 bytes.
The space taken for 6-shingles is 735138 bytes.
The space taken for 7-shingles is 1244047 bytes.
The space taken for 8-shingles is 1846872 bytes.
The space taken for 9-shingles is 2492055 bytes.
The space taken for 10-shingles is 3138890 bytes.
```

2. Power-Laws

(1) The unique words are saved in file 1.2.1_hapaxes.txt. The frequency of each word in this document is stored in parameter count in the program, (not outputted here).

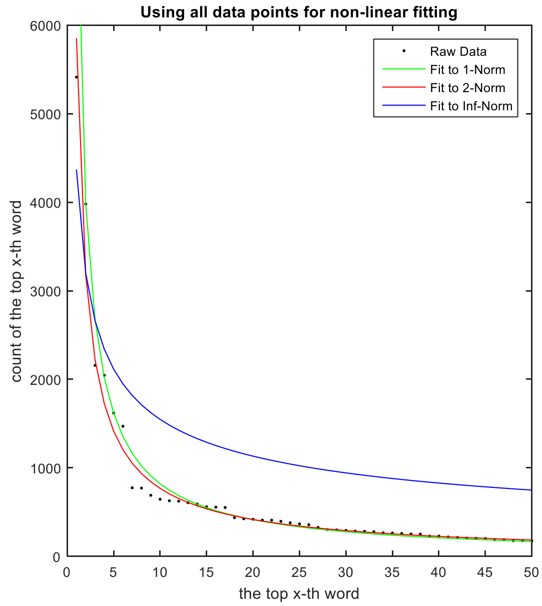
(2) D = 8807. The plot of function *f(x)*, the count of the top *x*-th word is:

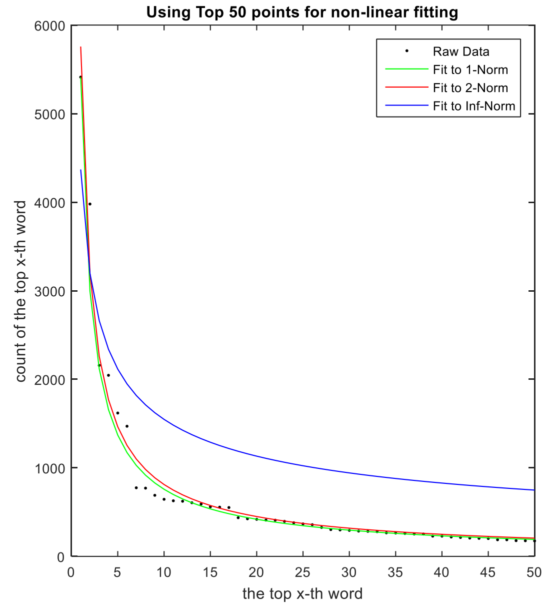


(3) Fit the function $f(x)$ by exponential law $g(x) = cx^{-a}$: I used Mathematica to find the best fit based the evaluation of 1-Norm, 2-Norm, and Inf-Norm. I compared the fitting results by using the whole data points for fitting with these by using the top 50 data points for fitting. The fitted parameters are as follows:

	whole data points		**Top 50 data points**	
	a	**c**	**a**	**c**
1-Norm	0.981599	7857.14	0.853588	5412.00
2-Norm	0.881837	5851.22	0.850784	5757.52
Inf-Norm	0.450842	4369.64	0.450842	4369.64

A plot of the fitting results is shown below:





From this figure, we can see that the fitting based on least squares is the best one, compared with the other two; there's not much difference between the fitting curves using all data points and these using the top 50 data points. Meanwhile, it can also be found that there's a huge uncertainty on the parameter c, but not a.

3. **Bonferroni's Principle**

Two people both deciding to visit a hotel on any given day is

$$0.01 \times 0.01 = 10^{-4}$$

Two people will visit the same hotel with probability

$$0.01 \times 0.01/200000 = 5 \times 10^{-10}$$

The chance that they will visit the same hotel on three different days is

$$(5 \times 10^{-10})^3 = 1.25 \times 10^{-28}$$

The number of pairs of people is

$$\binom{2 \times 10^9}{2} \approx \frac{(2 \times 10^9)^2}{2} = 2 \times 10^{18}$$

The number of pairs of three different days is

$$\binom{2000}{3} = \frac{2000 \times 1999 \times 1998}{3 \times 2 \times 1} \approx 1.33 \times 10^9$$

The expected number of suspected pairs is

$$2 \times 10^{18} \times 1.33 \times 10^9 \times 1.25 \times 10^{-28} = 0.3325$$

4. **Continued.**

The number of pairs of people is

$$\binom{100 \times 10^6}{2} \approx \frac{(100 \times 10^6)^2}{2} = 5 \times 10^{15}$$

The number of possible 10 items of the 1000 items is

$$\binom{1000}{10} \approx 2.63 \times 10^{23}$$

The chance of given two people that buy exactly the same set of 10 items is

$$1 \Big/ \binom{1000}{10} \approx 3.80 \times 10^{-24}$$

The chance of given two people to buy exactly the same set of 10 items in one year is

$$1 \Big/ \binom{1000}{10} \times 100 \times 100 \approx 3.80 \times 10^{-20}$$

The number of suspicious pars are

$$5 \times 10^{15} \times 3.80 \times 10^{-20} = 1.9 \times 10^{-4}$$

We would expect any such people found were truly terrorists, since Bonferroni principle says that we may only detect terrorists by looking for events that are so rare that they are unlikely to occur in random data.