Martin Isaksson
Elias Al-Tai

2016-10-31

Exercise 1

Content

This report includes a description of the problem to be solved, the theory used to solve the problem, the approach to solving the problem, the results from the code zipf.py and a discussion of the results.

Zipf's Law

There is an interesting phenomenon called Zipf's law. Zipf's law is an empirical law formulated using mathematical statistics. Many types of data studied in physical and social science can be approximated to a Zipfian distribution. There are several interesting examples of when Zipfian distributions occurs in the real world. One such example is that the population of the world's largest cities form a Zipfian distribution. Another example of a Zipfian distribution that occurs, is in natural language texts. In this report this example will be evaluated. In this case the frequency of different words will be in a Zipfian distribution. The Zipfian distribution occurs in all different natural languages. The frequency count of all the words in a *corpus* (a large collection of natural language texts) in any language will be evaluated and also the graph of frequency of each word against the word rank will be compared to Zipf's first law.

$$frequency = \frac{K}{rank}$$

Equation 1: Zipf's first law. K is the proportionality constant.

In the case of words in a corpora, the most common word will have rank 1 and the second most common word will have rank 2 and so on. The most frequent word in an English corpora is "the" and according to Zipf's law the second most frequent word's frequency will be half of the frequency of "the". The third most frequent word will have a frequency of a third of the frequency of "the", and so on. Therefore the frequency of any word is inversely proportional to its rank in the frequency table. An interesting point is that this rule, or law, is true for any language as mentioned and also true for languages written thousands of years ago that humans today can't understand. The visualization of a Zipfian distribution is best presented in a two dimensional graph. In a log-log-graph the Zipfian distribution is presented as a linear curve while in an ordinary graph it is presented as a curve that is exponentially decreasing.

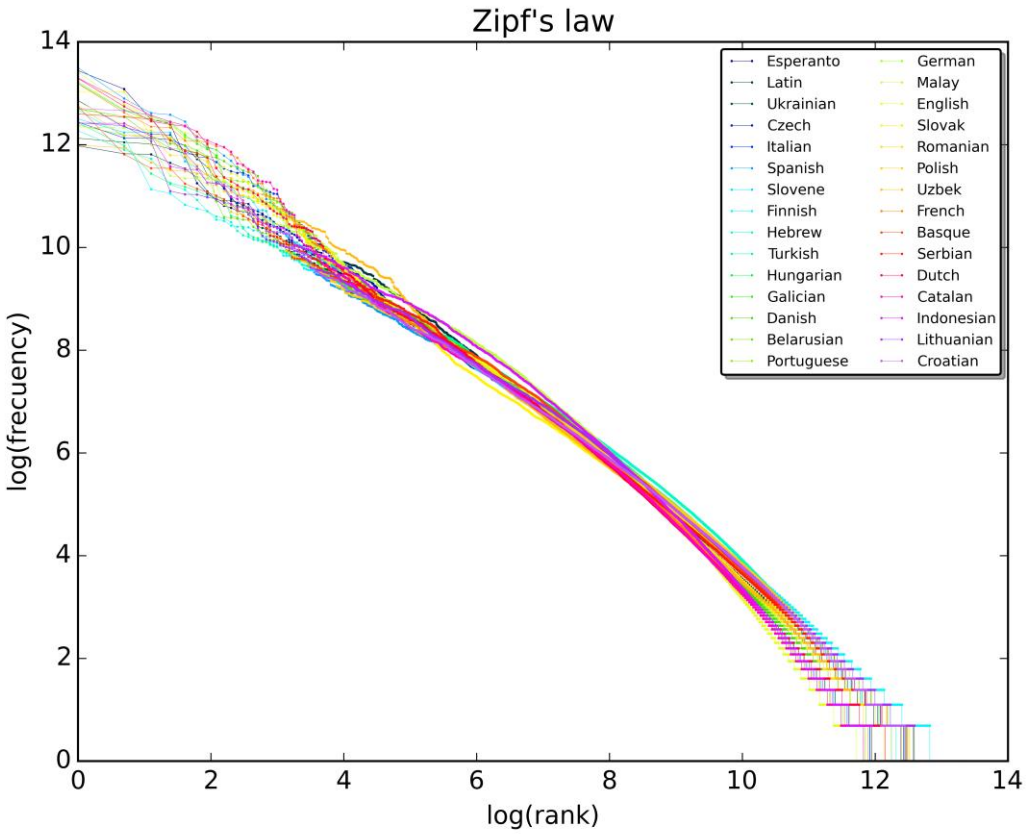

Figure 1 A plot of the rank versus frequency for the first 10 million words in 30 Wikipedias (dumps from October 2015) in a log-log scale.

Approach

In this exercise two corpora, one in English and one in Spanish, (the corpora were provided) are supposed to be analyzed. The first thing to do is to tokenize the texts, i.e. split the whole text into tokens by using the space between two tokens as a rule of where a token ends and a new one begin. The next step is to filter out tokens that are not words, for example numerical numbers. After that, count the words and plot the frequency of the words against the word rank (also in log-log to see if a linear curve is achieved). Then evaluate if the words in the corpora have a Zipfian distribution. The same procedure is later done to evaluate if single characters occur in the corpora with a Zipfian distribution.

Procedure

One approach to solving this exercise would have been to use the python library *re* and with that use regular expressions to tokenize and then extract the words from the tokens.

A more efficient way is to use the *NLTK* library, which is well described in the following links:

http://www.nltk.org/book/ch01.html - http://www.nltk.org/book/ch14.html

Martin Isaksson
Elias Al-Tai 2016-10-31

The *NLTK* library provide functions that tokenize, filter out certain tokens from the result of the tokenizer (words, characters, etc.), count word frequency, extracting a chosen number of the most common words in a list (sorting and extract e.g. 50 words), plotting and much more.

The functions mentioned above are the function from *NLTK* that were used to solve this exercise, in that order. The first function scans the text file and tokenizes it. The tokenizer function tokenizes a text into sequences of alphabetic and non-alphabetic tokens. This tokenizer function will create some errors with English text, for example the word “man’s” will be split into three tokens: [“man”, “’”, “s”]. After the corpus had been tokenized into alphabetic and non-alphabetic tokens, the alphabetic tokens were filtered out. After that all capital letters in tokens were changed to lower case letters in all of the words. The same procedure was repeated for English words and characters and Spanish words and characters.

Then plots were generated, which can be seen in ***Figure 2*** and ***Figure 3***.

To see if the formula in ***Equation 1*** is fulfilled, K is calculated for each word/character and then the mean is calculated of all those K’s when finally, the deviation from that K-mean value can be calculated for each word to see if it is approximately the same or very different.

Results

The results of the whole process can be seen in ***Figure 2*** and ***Figure 3***, and also the result of, if the law in ***Equation 1*** is fulfilled, is presented for each text (for both words and characters).

English Text

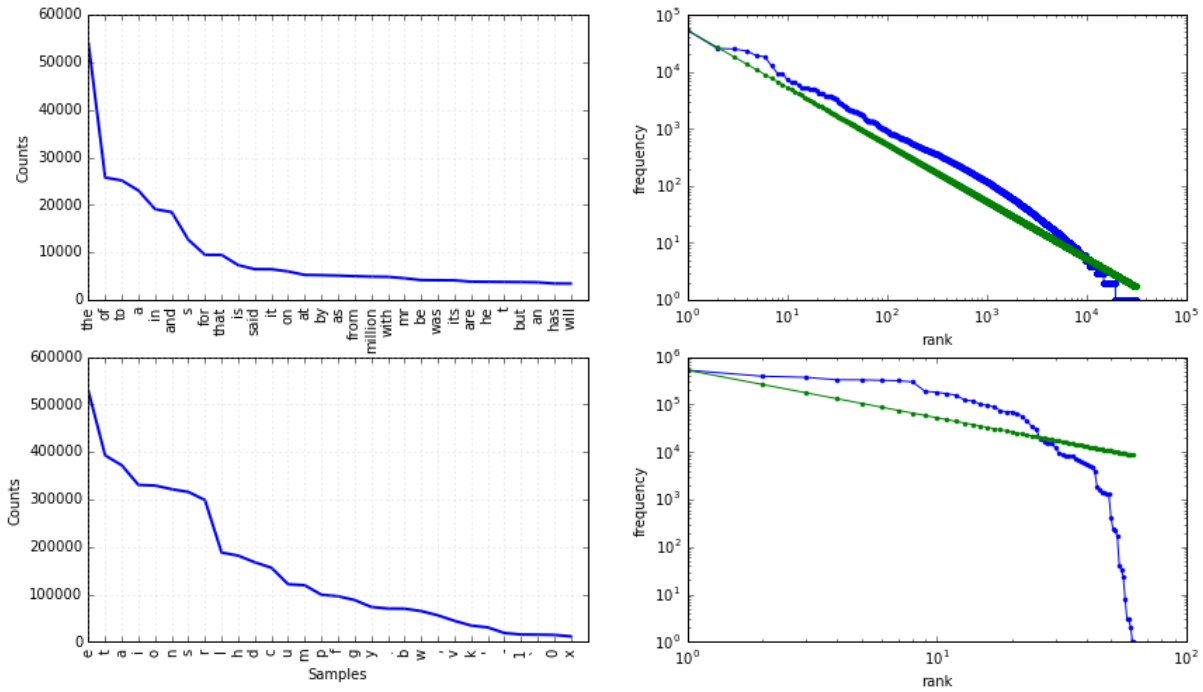

Figure 2 These are the plots for the English text. The upper plots correspond to the words and the lower plots correspond to the characters of all the words in the text. The plots to the left are the frequencies of the 50 most common elements and the plots to the right correspond to element frequency against the element rank with log-log graph. The green lines represent the Zipf’s first law where K is always the number of the most common element.

For the words

The mean of K here is **47603.5**. The K-values are in the interval **[19571,121680]**. The standard deviation is around **56.6 %**.

For the characters

The mean of K here was **690419.6**. The K-values are in the interval **[61,2390864]**. The standard deviation is around **102.0 %**.

Spanish Text

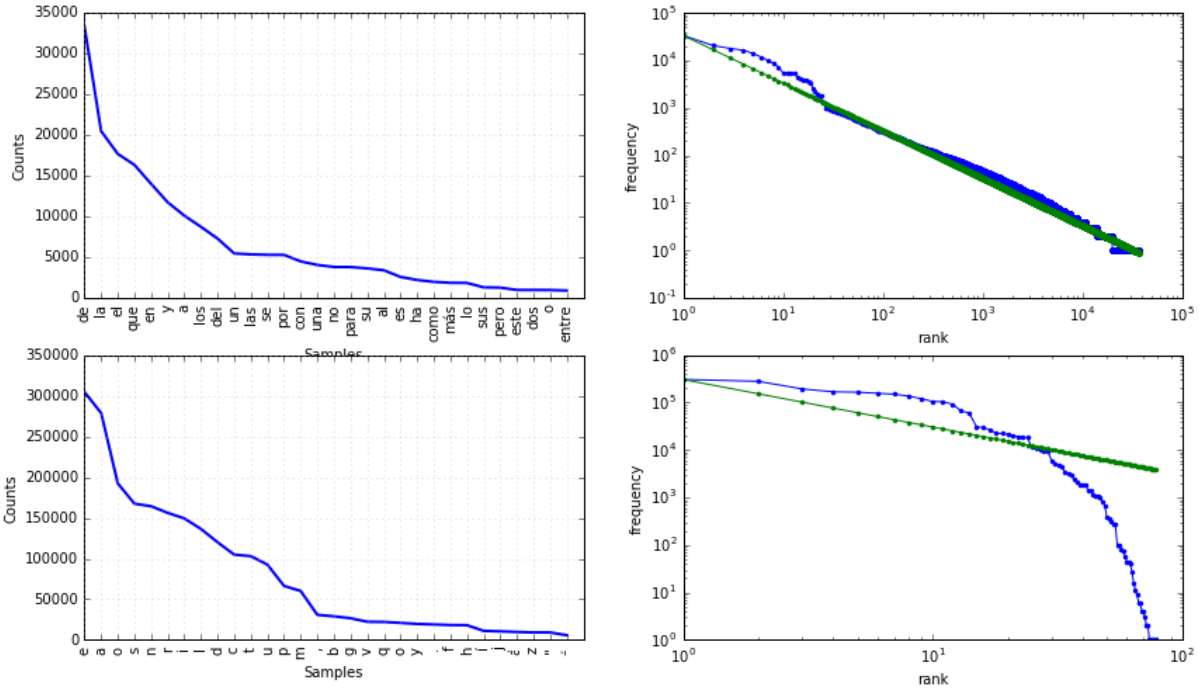

Figure 3 These are the plots for the Spanish corpora. The upper plots correspond to the words and the lower plots correspond to the characters of all the words in the text. The plots to the left are the frequencies of the 50 most common elements and the plots to the right correspond to the element frequency against element rank with log-log graph. The green lines represent the Zipf's first law where K is always the number of the most common element.

The mean of K here is **34798.2**. The K-values are in the interval **[20019, 70700]**. The standard deviation is around **22.8 %**.

For the characters

The mean of K here is **248533.5**. The K-values are in the interval **[74, 1132780]**. The standard deviation is around **134.5 %**.

Discussion

We were not sure if for example the word "man" should be counted separately when it occurs in a possessive form ("man's") but we assumed that the most logical thing would be not to count possessive form as a unique word and therefore we kept using this tokenizer.

Since some errors with the English language occurred for example the word "s" was very frequently occurring because of splitting up possessive words, some filtering functions could have been

hardcoded to filter out some of these incorrect words but we chose not do it because we wanted a general solution.

Another example of a problematic word in English with the chosen tokenizer was the word “you’re” which will be two words, “you” and “re”. These kinds of errors are mostly prevalent in for example the English language because it often uses the non-alphabetic character apostrophe in its words. This condition is however not occurring in the Spanish language that is another reason why it is interesting to keep the tokenizer as it is and compare the difference between how well Zipf’s law is being followed in the two cases where one of the language have some non-existing words.

For both of the texts the K value changes a lot from word to word. One reason for that is that Zipf’s first law ***Equation 1*** (f=K/r) isn’t bullet proof. Nowadays the law has been updated to:

$$frequency = \frac{K}{rank^s}$$

Equation 2: Zipf’s first law with a modification.

, where *s* is a number.

The Spanish corpus has more unique words, 37 507 of them, while the English corpus has 31 461 unique words. We think that this is one of the reasons that makes the distribution of words more similar to the Zipfian distribution in the Spanish corpus case than the English case. The standard deviation of the Spanish distribution is smaller than the standard deviation of the English distribution. And another reason as previously mentioned was the tokenization errors with the English corpus.

To also explore a lower level of language, the words were broken into characters, to see how well it suited Zipf’s law. The distribution of the characters were not remotely similar to a Zipfian distribution. The standard deviation of K is much higher for the characters than for words. Wentian Li demonstrated in his paper *Random Texts Exhibit Zipf's-Law-Like Word Frequency Distribution*, that words generated by randomly combining letters fit the Zipfian distribution. In his randomly generated text, the frequency distribution of word length was exponential. The words of length 1 occurred more than words of length 2 and so forth, with frequency declining exponentially with word length. Li showed mathematically that the power law distribution of frequency against rank is a natural consequence of the word length distribution. His underlying theory is that the rank distribution arises naturally out of the fact that word length plays a part — long words tend not to be very common, whilst shorter words are. It is easy to see how this has occurred in the evolution of language. Li argues that as Zipf distributions arise in randomly-generated texts with no linguistic structure, the law may be a statistical artifact rather than a meaningful linguistic property. We think that this is the major reason why characters do not occur in a Zipfian distribution. All characters have the same length, one – therefore the Zipfian distribution which as stated in the theory occurs naturally because shorter words are more common than longer ones, with the case of characters this cannot be the case.

Conclusion

The frequency of the words both in the English and Spanish corpus have a distribution that is very similar to the Zipfian distribution and therefore Zipf's law has been proven correct with this exercise.

The frequency of characters in both the English and Spanish corpus does not occur in a Zipfian distribution and therefore in this case Zipf's law cannot be applied.

References

Figure 1: https://commons.wikimedia.org/wiki/File:Zipf_30wiki_en_labels.png

Li, Wentian. "Random texts exhibit Zipf's-law-like word frequency distribution." *IEEE Transactions on information theory* 38.6 (1992): 1842-1845.