**The Most Frequent Words**

António Ramos, [ajframos@ua.pt](mailto:ajframos@ua.pt), 101193, MEI

*Resumo* – Este trabalho apresenta uma análise comparativa

dos contadores exatos e space-saving count

Para o estudo, foi usado livros de grandes dimensões de

língua portuguesa, inglesa e espanhola.

*Abstract* - This work presents a computational comparation

of exact counter and space-saving count

For the study, has used large books of Portuguese language,

English, French and Spanish.

# Introduction

In nowadays exists then need to count the most frequent words, in this work is analysed the exact counts and the algorithm from frequent words called space saving count.

For this comparison, it’s used a book in English “The Bible”, a book in Portuguese “Os Maias -Eps da Vida Romântica” and a book in Spanish “Don Quijote”. And it’s counted each word of the books.

# Algorithm analysis

In this chapter is it’s showed how the algorithm was analysed and what is the approach chosen.

## A. Handle information

Due to free choice on book, I choose “The Bible” to represent English, “Os Maias -Eps da Vida Romântica” to represent Portuguese and a “Don Quijote” to represent Spanish.

Since the requisite for this work is to remove the stop words from the counting of words, it’s used the package of python nltk, that contains a list of stop words of all languages and it’s also had a tokenizer for the text. It’s need it to install the package to the program works.

The choice of the book is made on execution of the main program. So, it’s processed one book at time.

*B. Exact Counter*

To do exact counter is created a dictionary that saves the word in lowercase and the counting’s. (Figure 1).

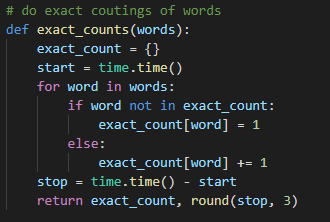


Figure 1 – Exact count

*C. Space-Saving-Count*

The algorithm of space saving count is implemented according to the slides and the author [1] (figure 2). It receives a list of all words of a document and a k, that’s chosen manually, the values are 10, 25, 50 and 70. It saves the counting’s of each word in a dictionary. For counting he checks if a word is on the dictionary and if not then verifies if the length of dictionary + 1 is bigger than the value stipulated then calculate the min counter of all words that is on dictionary and then removes it and add one count to the dictionary of that word. If the word exists counts more one value.

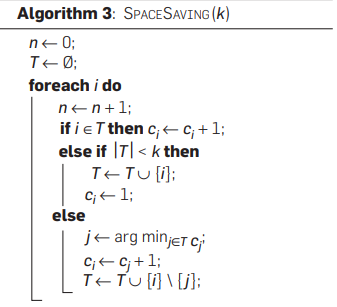


Figure 2 – Space Saving Count of [1]

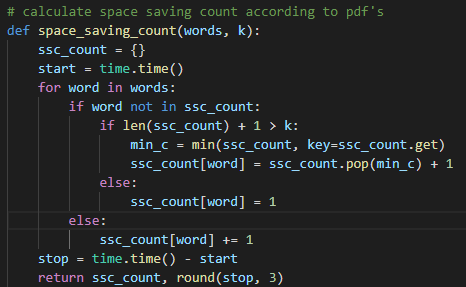


Figure 3 – Space Saving Count algorithm

# Tests Sequence Analysis

The tests performed are in the directories “results” and “results\_rel\_error”.

## Execution Time Analysis

The analysis regarding the execution time is given in seconds is in the bellow image.

The image has exact count, space saving count 10, 25, 50, 70. It’s analysed for all books, shown above. The time of execution of the counter exact is smaller than space saving counter. In space saving count when the k grown, the time also grown. This information is written to a file called “*results\_time.csv*” and to an image, for visuals proposals, called “*results\_time.png*”, in each execution of program.

1. *The Bible.*

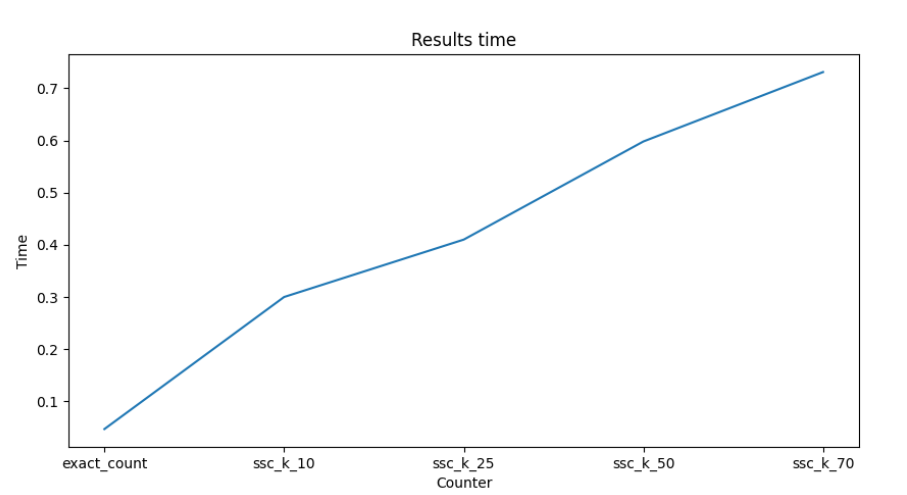


Figure 4 - Execution Time – The bible

1. *Don Quixote.*

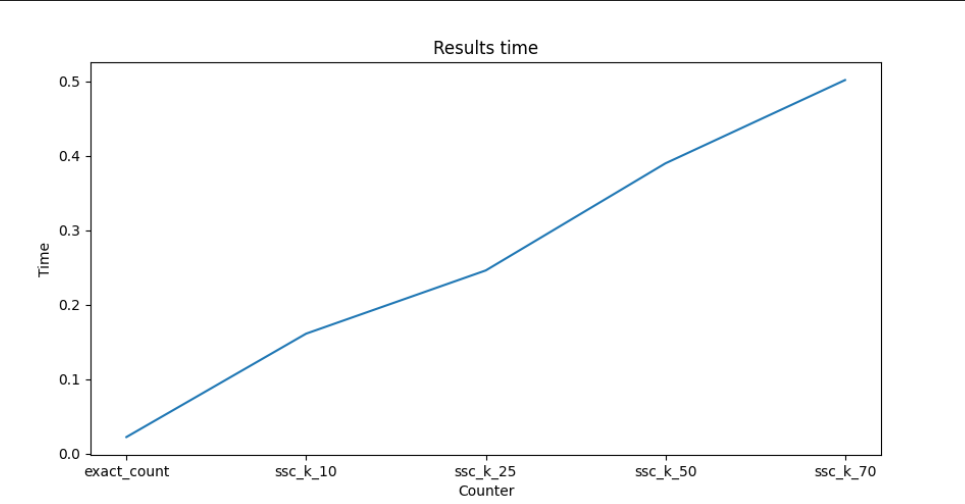


Figure 5 - Execution Time – Don Quixote

1. *Os Maias*

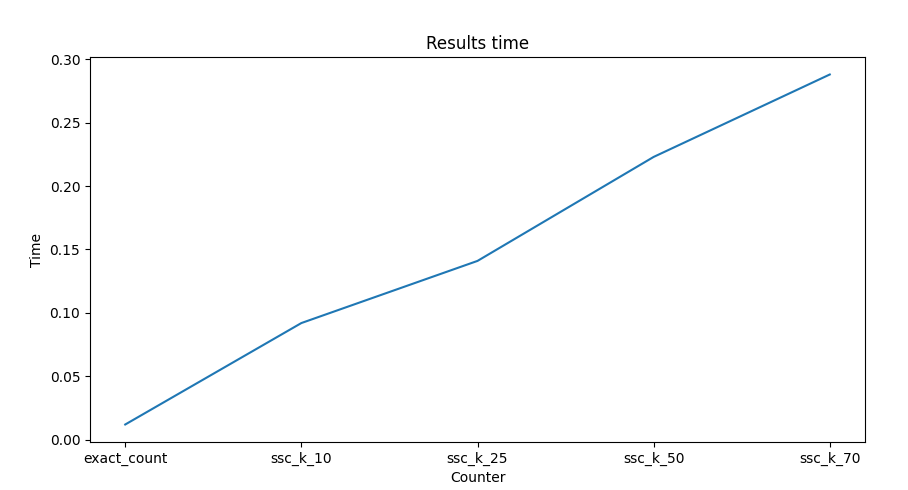


Figure 6 - Execution Time – Os Maias

1. *Relative Error*

To calculate the relative error for the algorithm space saving count, I based on expression of *[2]*

*|exact\_counter – ssc\_counter |/exact\_counter \* 100*

This information is written to a file called *“results\_rel\_error.csv”* in each execution of program*.* To visualize the information, it’s created a directory called “results\_rel\_error” that contains each k chosen. Since the k=10 is the k that haves least words, I will evaluate in each book. Regarding the values, it’s shown that the most frequent words have bigger error than the others.

1. *The Bible*

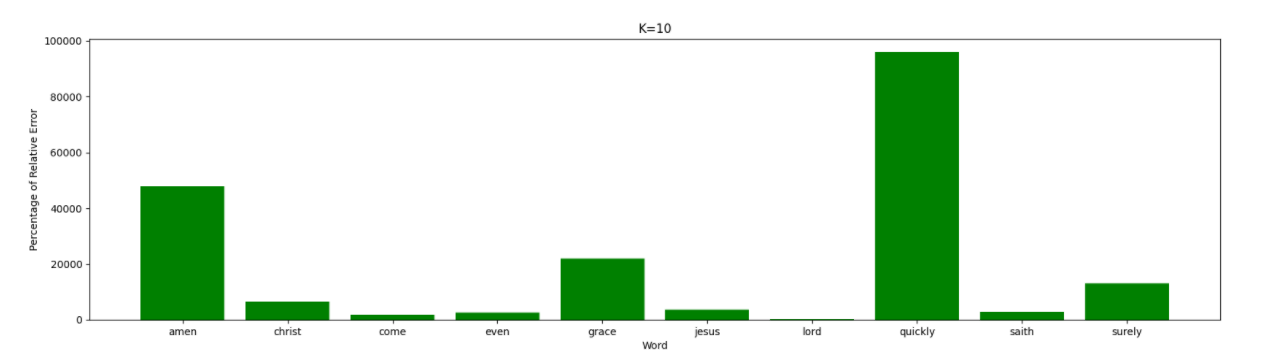


Figure 7 – Bar Chart Relative Error – The Bible

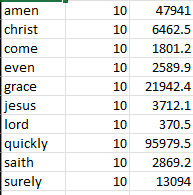


Figure 8 – Values of Relative Error – The Bible

1. *Don Quixote*

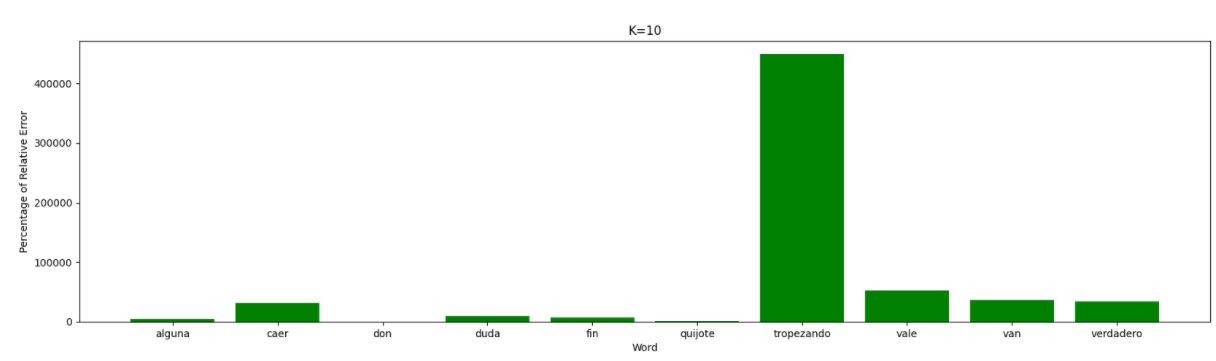


Figure 9 – Bar Chart Relative Error – Don Quixote

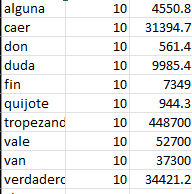


Figure 10 – Values of Relative Error – Don Quixote

1. *OS Maias*

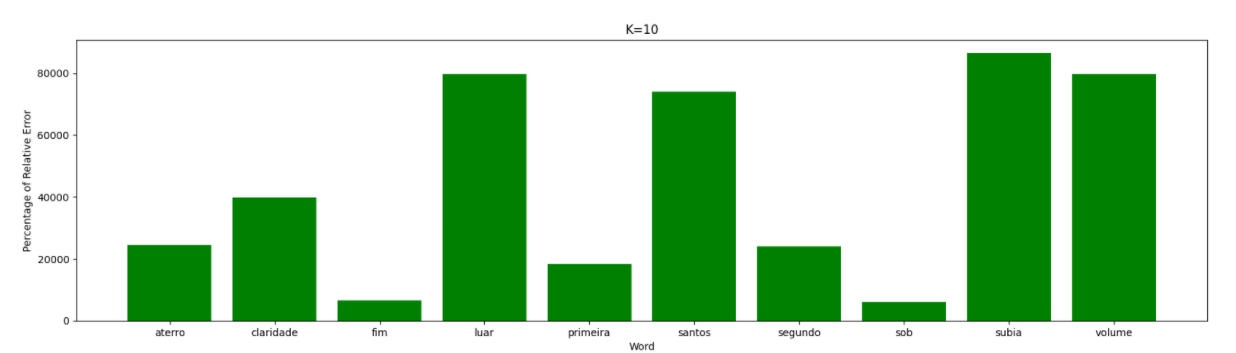


Figure 11 – Bar Chart Relative Error – Os Maias

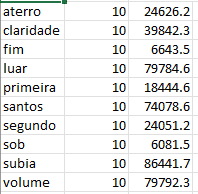


Figure 12 – Values of Relative Error – Os Maias

1. *Most Frequent Item*

Regarding the results for the most frequent item for exact count and space saving count, it has created a file “results\_count\_word”. It’s shown that if the k is bigger then he will be getting the counting’s of exact count. For example, in k = 70 the counting’s is almost equal than the exact count.

1. *The Bible*

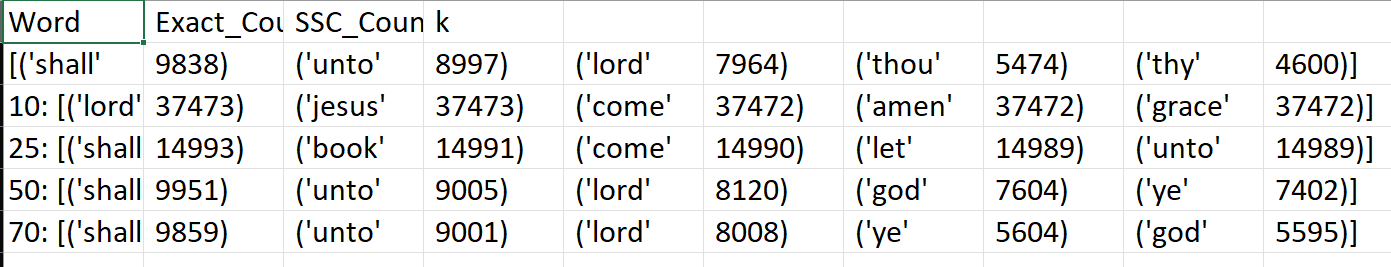


Figure 13 – Most Frequent Item – The Bible

1. *Don Quixote*

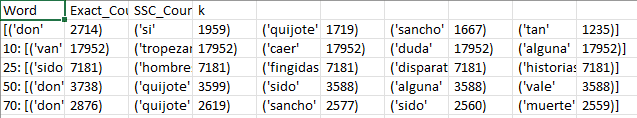


Figure 14 – Most Frequent Item – Don Quixote

1. *Os Maias*

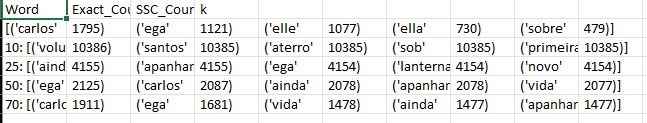


Figure 15 – Most Frequent Item – Os Maias

# Conclusion

The relative error and the most frequent item indicate that bigger k has low relative error and it almost get right the counting’s of exact count. So, the best k for these books is k = 70, even the time of execution is higher than exact count.

# References

[1] D. Cono D’elia, “Mining heavy hitters with Space-Saving,” 2013, Accessed: Jan. 24, 2022. [Online]. Available: http://developer.gnome.org/glib/.

[2] <https://www.greelane.com/pt/ci%c3%aancia-tecnologia-matem%c3%a1tica/ci%c3%aancia/how-to-calculate-percent-error-609584/>