Dealing with zero word frequencies: A review of the existing rules of thumb and a suggestion for an evidence-based choice

Marc Brysbaert   Kevin Diependaele

Ghent University

Key words: word frequency, Laplace transformation, Good-Turing algorithm, zero frequency

Address: Marc Brysbaert
Department of Experimental Psychology
Ghent University
H. Dunantlaan 2
B-9000 Ghent (Belgium)
e-mail: marc.brysbaert@ugent.be
Tel. + 32 9 264 64 07

Abstract

A critical review of the heuristics used to deal with zero word frequencies shows that four are suboptimal, one is good, and one may be acceptable. The four suboptimal strategies are: discarding words with zero frequencies, giving words with zero frequencies a very low frequency, adding one to the frequency per million, and making use of the Good-Turing algorithm. The good algorithm is the Laplace transformation, which consists of adding 1 to each frequency count and increasing the total corpus size by the number of word types observed. A strategy that may be acceptable is to guess the frequency of absent words on the basis of other corpora and increasing the total corpus size by the estimated summed frequency of the missing words. A comparison with the lexical decision times of the English Lexicon Project and the British Lexicon Project suggests that the Laplace transformation gives the most useful estimates (in addition to being easy to calculate). Therefore, we recommend it to researchers.

One of the thorny issues in word recognition studies arises when researchers want to use words not present in their preferred word frequency list. Although it is tempting to assign such words a frequency of 0, this creates problems when one needs the logarithms of the frequencies, because the logarithm of 0 goes to minus infinity and, therefore, is not given by most calculators or software packages. As usual, when confronted with this type of mathematical nuisance psychology researchers have developed a number of heuristics, which are passed on from one generation to the next without much justification. The practice commonly elicits probing questions from new, critical students, but they rapidly learn to adapt when they realize that finding answers is not trivial and risks detracting them from their real research. One would expect the providers of word frequency lists to give some guidance, but to our knowledge this has not happened so far.

It might be argued that the problem of zero frequencies is likely to disappear in the near future, given that word frequency measures are calculated on increasingly large collections of materials. Indeed, one would not expect an interesting word to be absent from a corpus of more than one hundred billion words, such as the Google Books corpus (Michel et al., 2011). This is true, but analyses have indicated that frequency measures based on such large (internet-based) corpora are not the best to predict word processing times in psycholinguistic studies. More variance in word processing performance is accounted for by frequency estimates from smaller corpora that are more representative of the language participants of psychology experiments have been exposed to (Brysbaert, Keuleers, & New, 2011). Although frequency measures based on very large corpora provide estimates for all words, they do not provide very good estimates.

There are two reasons why word frequencies from very large corpora are not doing very well. The first is that these corpora are often based on sources unlikely to be read regularly by undergraduate students, such as encyclopedias (in particular Wikipedia) and non-fiction works. The second reason is that very large corpora tend to contain relatively more rare words than participants are likely to have encountered in their life. Kuperman and Van Dyke (in press) showed by means of simulations that large corpora tend to overestimate the frequency with which participants have encountered low-frequency words. This is particularly true for low-proficiency participants. Assuming an average input rate of 200 words per minute (speech rate is usually lower than 170 words per minute; reading rate is rarely more then 300 words per minute; Rayner & Pollatsek, 1989) for 16 hours a day, a 20-year old participant is unlikely to have come across more than 200 x 60 x 16 x 365.25 x 20 = 1.4 billion words. This then would be the maximum corpus size to be used. However, even with this corpus size zero word frequencies are bound to be present. So, researchers will keep on finding them confronted with the issue of what to do with words for which there are no frequency data.

In this paper we review the main heuristics currently in use to deal with zero frequencies and see whether we can come to an evidence-based conclusion. To make the text as accessible as possible for researchers with little mathematical background, we illustrate the various calculations with a toy corpus consisting of the 50 first words of the fairy tale “The ugly duckling”, shown in Figure 1. This corpus consists of 50 words, which are called tokens. Some of the words are the same, making that the number of different word forms (types) is smaller than the number of tokens, namely 41 types. A total of 34 word types have a frequency of 1; five word types have a frequency of 2; and two word types have a frequency of 3. We will estimate the frequency of the missing word “father”.

Insert Figure 1 here

Existing rules of thumb

In this section we review a number of heuristics we have come across in our work on word frequency. As some of them are not well-founded, we think it is better not to link them to specific authors. Readers can see for themselves which heuristic has been used in the papers they read.

Discard words with zero frequencies. A first way of dealing with zero frequencies is simply to discard words with zero frequencies. This is often done implicitly, when researchers select words on the basis of an existing frequency list, such as Kucera and Francis (1967), Celex (Baayen, Piepenbrock, & Gulikers, 1995), Elexicon (Balota, Yap, Cortese, Hutchison, Kessler, Loftis, Neely, Nelson, Simpson, & Treiman, 2007), or SUBTLEX (Brysbaert & New, 2009). Only words present in the list are selected then. Sometimes, the criterion is made explicit, for instance when researchers analyze words selected on the basis of one list with a frequency measure from another list. In order not to advantage one list over the other, researchers in such a situation sometimes decide to delete words if they are absent from either list. A less fortunate aspect of this strategy is that it discards low-frequency words, which reduces the range of the frequency values and, hence, the size of the frequency effect that can be obtained. Applied to the toy corpus, the strategy assumes that the word “father” does not exist (or is not of interest to researchers).

Add 1 to each frequency. A second heuristic, arguably the most popular, is to add 1 to each frequency value. So, words not observed in the corpus get a frequency count of 1; words observed once in the corpus get a frequency of 2; words observed twice get a frequency of 3; and so on. This transformation was originally proposed by Laplace and is known as the Laplace smoothing. What is less known to psychology researchers is that correct application of the Laplace algorithm requires correction of the corpus size. Because all observed frequencies are assumed to be underestimates, the theoretical corpus size is larger than the observed size to make room for words in the language that are not observed in the corpus. Specifically, Laplace smoothing assumes that the theoretical corpus size equals the number of word tokens plus the number of word types. Figure 2 summarizes the Laplace transformation for the toy corpus of Figure 1. On the basis of this corpus, the estimated frequency of the unobserved word "father" is 1/91 or .011.

Insert Figure 2 here

The correction of the corpus size required for Laplace smoothing is often ignored by psychologists using word frequencies. Indeed, authors regularly seem to treat the Laplace transformation as a "virtual" transformation, only needed to calculate logarithms. So, they freely combine the observed frequencies and log(frequency+1) in summary tables, not realizing that the Laplace smoothing also changes the totals and hence the assumed frequencies. Luckily, the omission is less severe than suggested by the toy corpus. Because of its small size, the number of word types relative to the number of word tokens in this corpus is high (41 vs. 50), requiring a hefty correction of the denominator. The correction is much smaller for large corpora. For instance, if 100 thousand word types are observed in a corpus of 100 million words, the corrected corpus size is only 100.1 million, which will not have a large practical impact on the frequency measures calculated as frequency per million words (pm). Still, it shows how the heuristics used in word frequency research tend to simplify algorithms in ways that are mathematically not fully acceptable.[1]

Add a small value to every frequency. Because adding 1 to each frequency changes the totals, some researchers add a very small value to the frequency or simply assume that words with frequency 0 in reality have a very low frequency, for instance .0001. The former in mathematics is known as Lidstone smoothing; the latter could be described as a psychological simplification of the Lidstone algorithm. Figure 3 applies the Lidstone transformation to the toy corpus. Notice that the Lidstone transformation assumes that unobserved words are extremely rare (in the example 10 thousand times less likely than words observed once in the corpus). Applied to the toy corpus, it assumes that the frequency of the absent word “father” is .0001.

Insert Figure 3 here

Add 1 to frequency per million. A final heuristic we have come across is to add 1 to the frequency pm. The origin of this heuristic is not clear (at least, there is no analogue in the mathematical literature). In all likelihood, it is the outcome of two simplifications: (1) to standardize frequency measures across corpora, it is better to express them as frequencies pm, and (2) to calculate logarithms, one has to add 1 to the frequencies. A further contributing factor may have been that the widely used Kucera and Francis (1967) word

[1] Translated to the Kucera and Francis (1967) list that has been used extensively in word frequency research, the corpus size to be used for Laplace-transformed word frequencies is not 1.014 million (the number of tokens in the corpus) but 1.064 million (the number of tokens plus the number of types observed).

frequencies were based on a corpus of about 1 million words, so that adding 1 was equivalent to adding 1 to frequency pm.

The thoughtlessness with respect to the heuristic becomes clear when we translate it to a currently realistic corpus size of 100 million words. If we add 1 pm to each of these frequencies, then we actually assume that words not encountered in the corpus “should have” been there 100 times (0 + 1 pm). Words observed once in the corpus, “should have” been there 101 times (1 + 1 pm), and so on. As it happens, this reduces the precision of the frequency measure to that of Kucera and Francis and thus easily decreases the percentage of variance explained by word “frequency” with 5 to 10% (see Brysbaert & New, 2009, for a comparison of the new frequency measures to those of Kucera and Francis). Also notice that this algorithm mathematically requires, just like the Laplace transformation, that the theoretical size of the corpus is increased by the number of word types times 1 pm. So, a corpus of 100 million words resulting in a word list of 100 thousand words would have to be increased by 10 million words (100 thousand word types times 100 missed observations per word). For a corpus smaller than 1 million words, like our toy corpus, the strategy turns into the Lidstone transformation, as the frequency of the missing word “father” would be estimated to be 1 pm or .000001. Because there are so many things wrong with this strategy, we will not include it in the analyses below.

The Good-Turing algorithm

Anybody interested in missing types is bound to come across the Good-Turing algorithm (or a variant of it), which is the golden standard for estimating the probability of unseen instances in frequency distributions, even though references to the algorithm in the word frequency literature are rare (but see Baayen, 2001; Jurafsky & Martin, 2009) for reasons outlined below.

The basics of the algorithm are the following:

- The observed frequencies are overestimates of the “real” frequencies. So, in the toy corpus the frequencies of the words observed once are in reality lower than 1/50.
- The corrected frequencies can be obtained by comparing the sizes of the various frequency bins. So, the correct frequency of the words seen once in the corpus is estimated on the basis of the number of words seen once relative to the number of words seen twice.
- The probability of unseen observations equals the probability of the words seen once.

Figure 4 illustrates the calculation of the basic steps of the Good-Turing algorithm for the toy corpus. Given that many words are observed only once (34/50), the algorithm postulates that the probability of missing observations is quite high, namely 34/50 or .68. So, the observed words constitute only 32% of the assumed population of words. This requires the recalculation of the observed frequencies, so that they sum up to 16 (32% of N = 50). The recalculation happens on the basis of the number of words observed in the frequency bin under consideration relative to the number of words in the next frequency bin. So, the frequency of the words observed once in the corpus is calculated on the basis of the number of words observed once and the number of words observed twice. This results in a recalculated frequency of .294 instead of 1. Similarly, the words encountered twice in the corpus have a recalculated frequency of 1.2 rather than 2. The basic version of the Good-

Turing algorithm cannot recalculate the frequency of the words observed three times in the corpus, because there is no higher frequency bin.

Insert Figure 4 here

A full-fledged version of the Good-Turing algorithm provides estimates for all frequencies (including the highest one). It also allows for missing bins and for the fact that the observed numbers are noisy estimates (i.e., subject to measurement error). These calculations are considerably more complicated (see Gale & Sampson, 1995, for an introduction), but can be circumvented by making use of existing software packages.[2] The last column of Figure 4 shows the outcome of such an algorithm. As can be seen, the recalculated frequencies of 1 and 2 are lower still (respectively.26 and .83) to make room for the recalculated frequency of words observed three times (with a recalculated frequency of 1.5).

The Good-Turing algorithm is mathematically well founded (based on the binomial distribution) and also makes sense intuitively:

- The population frequencies of the words in the corpus are lower than their frequencies in the corpus, because their inclusion was partly due to chance (they happened to be in the texts sampled).
- The population frequencies of words not observed in the corpus are higher than zero, because their absence was partly due to sampling error (they happened not to be in the texts sampled).
- The number of words observed once gives an estimate of the number of words missed. The more instances of single observations (also called hapax legomena) the

[2] We have developed an easy to use Python algorithm, which is available as supplementary materials, together with a tutorial on how to install and use it.

more likely the corpus undersampled the richness of the language. The fact that 34 of the 50 words in our toy corpus were observed only once is a sure indication that the corpus size was way too small to capture the richness of the vocabulary. In a corpus of 100 billion words we would not expect the majority of words being observed only once (as a matter of fact, the larger the corpus, the fewer genuine words we would expect to observe only once).

At the same time, the Good-Turing algorithm has its limitations. For a start, it does not give a direct answer to the most pressing question: Which frequency value should be assigned to words not observed in the corpus? The Good-Turing algorithm says which probability mass must be set aside for unobserved types, but remains silent about the number of unobserved types over which the probability mass must be distributed. Applied to the toy corpus, the Good-Turing assigns the absent word "father" to the missing probability mass of .68, but does not say over how many words this mass must be distributed and, hence, what the probability of each word is. There are other algorithms to estimate this number (e.g., Wang, 2011), but they give quite divergent estimates. Furthermore, there are good arguments to assume that the human lexicon in principle is infinite (Kornai, 2002), which brings us back to the starting problem (given that probability mass divided by infinity is 0).

A second, related problem with the Good-Turing algorithm is that many hapax legomena in corpora turn out to be typos and transcription errors. So, the critical mass of words observed only once in a corpus (on which both the probability mass of the unobserved types and the recalculated frequencies of the hapax legomena are based) is suspect. It most certainly is larger than warranted by the diversity of the language. This means that some cleaning needs to be done (which is time-consuming and open to criticism).

Finally, the Good-Turing algorithm depends on the assumption that words are randomly distributed over the corpus. Given that corpora always consist of text pieces (with words occurring in clusters) this is another concern, although it is less of an issue when a corpus includes very many short excerpts from a great variety of sources.

For the above reasons, a pattern usually observed in word frequency research is one of initial attraction to the Good-Turing algorithm, followed by some disillusion and the realization that the algorithm raises as many questions as it solves (at which point the researcher moves on and leaves the question of what frequency to use for unobserved words unanswered).

Figures 5 and 6 show the outcome of the algorithm when it is applied to lowest frequencies of KF and SUBTLEX. While interpreting the data, it is important to keep in mind that Kucera and Francis worked with a heavily edited corpus (i.e., no typos) and that SUBTLEX only included entries that passed a spell checker. Also notice that the Good-Turing algorithm does not give us an estimate of the frequency of the words not observed in the corpus. It only provides us with an estimate of the probability mass of missing word types.

Insert Figures 5 and 6 here

A pragmatic alternative?

As long as there was only one frequency list, the issue of zero word frequencies had to be solved in a principled way. However, now that it is easy to collect very large corpora, we can start comparing estimates. For instance, in hindsight we can estimate how large the 0 word frequencies would have been if in 1967 Kucera and Francis had had access to the 50+ million word SUBTLEX corpus or to the 100+ billion word Google Books American English corpus.

To do this analysis, ideally one would have access to a list of all word types in American English. Clearly, this does not exist. A more realistic alternative is to work with as many words as possible. In the remainder we will work with a list of 66,516 American English words. This list started from the SUBTLEX-US list from which all proper nouns and entries with Arabic numbers were discarded. Subsequently it has been updated every time we came across a word used in psycholinguistic research that was not in the original SUBTLEX-US list. At the time of the analysis, there were 8,054 extra words. For comparison purposes, 35,958 words in the list were absent from the KF corpus, and 3 were absent from Google Books American English (herbivorously, photoengrave, and scarpers). The total number of counts of all words in the list was .946 million in KF, 48.12 million in SUBTLEX, and 118.49 billion in Google Books American English.

Figure 7 shows the relationship between the KF frequencies and the SUBLTEX and Google frequencies for the first 10 KF frequency values. As can be seen, the relationship is largely linear, meaning that we can use linear regression analysis to estimate the average frequency of the missing words in KF. According to SUBTLEX, it is .31; according to Google it is .15. Part of the difference arguably is due to the fact that SUBTLEX (and, hence, our word list) contains colloquial words rarely used in books.

Insert Figure 7 here

Similarly, we can look at which Google frequencies the words not in the SUBTLEX-US list have. Figure 8 shows the outcome. Here we see something unexpected, namely that the words with 0 frequency in SUBTLEX have the same Google frequency as the words with frequency 1 in SUBTLEX. Two likely contributing factors are (i) the fact that some written words are rarely used in social interactions (on which the subtitles are based), and (ii) the fact that the word list was largely derived from the SUBTLEX-US list, meaning that sampling error played a large role in determining which words were included once or not in our list. Indeed, according to the binomial distribution (which forms the basis of frequency distributions), chances of observing an instance with a low frequency once in a large corpus are nearly the same as not observing it [3]. A reasonable approximation, therefore, might be to give the 0 frequencies in SUBTLEX a frequency of 1 (and to assume that the theoretical SUBTLEX corpus size should be increased by 8,054 – the number of words missing in SUBTLEX-US).

Insert Figure 8 here

[3] The probability of observing a word with probability *p* once in a corpus of *n* words according to the binomial distribution is $n*p*(1-p)^{n-1}$. Chances of not observing it are $(1-p)^n$. If p equals 1/n (i.e., is expected to be observed only once in the corpus), then chances of observing it once are $[(n-1)/n]^{n-1}$ and chances of not observing it $[(n-1)/n]^n$ or $[(n-1)/n]^{n-1} * [(n-1)/n]$. The larger n, the more [(n-1)/n] approaches unity and the closer both probabilities come.

How well do the various estimates predict lexical decision times?

To see how good the various estimates are, we can compare them to the word processing times from megastudies. There are two such studies: The English Lexicon Project (Balota et al., 2007) and the British Lexicon Project (Keuleers, Lacey, Rastle, & Brysbaert, 2012). Figure 9 shows the data for the three main transformations of KF; Laplace, Lidstone, and Google-based. The Good-Turing algorithm is not included, as it does not give an estimate of the zero frequency.

Insert Figure 9 here

It is clear from Figure 9 that the Lidstone transformation is about the worst one can do, because it sets the zero frequencies too far apart from the other frequencies. Laplace seems to be the best option. The zero frequency estimated on the basis of another corpus would do better if it were based on SUBTLEX rather than on Google. Given that the SUBTLEX estimate was .31 (rather than .15) this would have brought the log10 value to -.5 instead of -.8.

Figure 10 shows the data for the various transformations of the SUBTLEX-US frequencies. Again, the Laplace transformation is doing well, certainly for the ELP data. For BLP the frequency of the words not present in the corpus seems to be underestimated. For this database, the frequency based on Google estimates is better. Again, the Lidstone transformation is to be avoided.

Insert Figure 9 here

Discussion

A review of the literature of strategies to deal with zero frequencies reveals four suboptimal techniques. The exclusion of words with zero frequencies cannibalizes on possibly interesting rare words, in particular when frequency measures from several (small-scale) sources are compared. The Lidstone transformation puts the words with zero frequencies so far from the other words that they become outliers for statistical analyses. More worryingly, if not excluded from the analysis, such outliers can have a strong impact on the outcome (killing a correlation opposite to the outliers' effect or boosting the significance of a negligible correlation in line with the outliers). The "add 1 pm" approach turns into a Lidstone transformation for word corpora smaller than 1 million words and otherwise levels the strong frequency effect observed for words with frequencies smaller than 1 pm (Brysbaert, Buchmeier, Conrad, Jacobs, Bölte, & Böhl, 2011; Keuleers, Diependaele, & Brysbaert, 2010). The latter is particularly of concern when researchers must match low-frequency words across conditions. Given that words with frequencies of .1 pm take about 100 ms longer to process in lexical decision experiments than words with frequencies of 1 pm (Keuleers et al., 2010, 2012), it is important to make sure that the words are carefully matched on log frequency. This is the case when the logs of the raw frequencies are used or when the Laplace transformation is used, but the frequency difference becomes substantially smaller when log(frequency pm + 1) is used. Finally, the Good-Turing algorithm does not provide a straightforward estimate for unobserved words, even though it re-estimates the other frequencies in a sensible way.

Two other approaches produce a more acceptable output. The first is the Laplace transformation. It consists of adding one to each frequency count and increasing the corpus size by the number of word types. The resulting frequencies provide a good predictor of the lexical decision times from the English and the British Lexicon Projects (Figures 9 and 10). Finally, there seems to be some mileage in empirically estimating the zero frequencies on the basis of another (larger) corpus. A caveat to this strategy is Kuperman and Van Dyke's (in press) recent demonstration that very large corpora tend to overestimate the frequencies of low-frequency words, particularly for students with less extensive language exposure.

All in all, it looks like nothing beats the Laplace transformation at present. In addition, it is easy to calculate and does not require access to an alternative, large-scale corpus. This then is our best buy advice for the moment. It is also the frequency measure we will make available in the coming releases of our SUBTLEX word frequency measures (i.e. it will be integrated in the frequencies pm we give and in the log frequencies). Indeed, it is often forgotten that a valid solution of the 0-frequency problem requires the (slight) recalculation of all other frequency measures as well (see the introduction).

References

Baayen, R. H. (2001). *Word Frequency Distributions*. Dordrecht: Kluwer Academic Publishers.

Baayen, R.H., Piepenbrock, R., & Gulikers, H. (1995). *The CELEX Lexical Database* [CD-ROM]. Philadelphia, PA: Linguistic Data Consortium.

Balota, D.A., Yap, M.J., Cortese, M.J., Hutchison, K.A., Kessler, B., Loftis, B., Neely, J.H., Nelson, D.L., Simpson, G.B., & Treiman, R. (2007). The English Lexicon Project. *Behavior Research Methods, 39*, 445-459.

Brysbaert, M., Buchmeier, M., Conrad, M., Jacobs, A.M., Bölte, J., & Böhl, A. (2011). The word frequency effect: A review of recent developments and implications for the choice of frequency estimates in German. *Experimental Psychology, 58*, 412-424.

Brysbaert, M., Keuleers, E., & New, B. (2011). Assessing the usefulness of Google Books' word frequencies for psycholinguistic research on word processing. *Frontiers in Psychology, 2*:27.

Brysbaert, M., & New, B. (2009). Moving beyond Kucera and Francis: A critical evaluation of current word frequency norms and the introduction of a new and improved word frequency measure for American English. *Behavior Research Methods, 41*, 977-990.

Gale, W.A., & Sampson, G. (1995). Good–Turing frequency estimation without tears. *Journal of Quantitative Linguistics, 2*, 217–37.

Jurafsky, D., & Martin, J.H. (2009). *Speech and Language Processing: An Introduction to Natural Language Processing, Speech Recognition, and Computational Linguistics* (2nd edition). New York: Prentice-Hall.

Keuleers, E., Diependaele, K. & Brysbaert, M. (2010). Practice effects in large-scale visual word recognition studies: A lexical decision study on 14,000 Dutch mono- and disyllabic words and nonwords. *Frontiers in Psychology 1*:174. doi: 10.3389/fpsyg.2010.00174

Keuleers, E., Lacey, P., Rastle, K., & Brysbaert, M. (2012). The British Lexicon Project: Lexical decision data for 28,730 monosyllabic and disyllabic English words. *Behavior Research Methods, 44*, 287-304.

Kornai, A. (2002). How many words are there? *Glottometrics, 4*, 61-86.

Kucera, H., & Francis, W. (1967). Computational analysis of present-day American English. Providence, RI: Brown University Press.

Kuperman, V., & Van Dyke, J.A. (in press). Reassessing word frequency as a determinant of word recognition for skilled and unskilled readers. *Journal of Experimental Psychology: Human Perception and Performance*.

Michel, J.B., Shen, Y.K., Aiden, A.P., Veres, A., Gray, M.K., Pickett, J.P., Hoiberg, D., Clancy, D. et al. (2011). Quantitative Analysis of Culture Using Millions of Digitized Books. *Science 331*, 176–182.

Rayner, K., & Pollatsek, A. (1989). *The Psychology of Reading*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Wang, J.P. (2011). SPECIES: An R package for species richness estimation. *Journal of Statistical Software, 40*, 1-15.

Figure 1 : Toy corpus used to illustrate the calculations

Corpus:

> "Once upon a time down on an old farm, lived a duck family, and Mother Duck had been sitting on a clutch of new eggs. One nice morning, the eggs hatched and out popped six chirpy ducklings. But one egg was bigger than the rest, and it did not hatch."

Number of words in the corpus (tokens): 50

Number of different word forms observed in the corpus (types): 41

Frequency spectrum:

- Number of word types with frequency 1: 34 (an, been, bigger, …, time, upon, was)
- Number of word types with frequency 2: 5 (duck, eggs, on, one, the)
- Number of word types with frequency 3: 2 (a, and)

Missing word for which the frequency will be estimated: "father"

Figure 2: Laplace smoothing for the toy corpus

Tokens in corpus: 50

Types in corpus: 41

Original	times	Sum	Freq/	Transformed	Sum	Freq/
frequency	observed		total	frequency	Laplace	total
0	0	0 x 0 = 0	0/50	1		1/91
1	34	1 x 34 = 34	1/50	2	2 x 34 = 68	2/91
2	5	2 x 5 = 10	2/50	3	3 x 5 = 15	3/91
3	2	3 x 2 = 6	3/50	4	4 x 2 = 8	4/91
			------------			-------------
Total freq*obs			50			91

Figure 3: Lidstone smoothing for the toy corpus

Tokens in corpus: 50

Types in corpus: 41

Original frequency	times observed	Sum	Freq/ total	Transformed frequency	Sum Lidstone	Freq/ total
0	0	0 x 0 = 0	0/50	.0001		.0001/50.0041
1	34	1 x 34 = 34	1/50	1.0001	1.0001 x 34 = 34.0034	1.0001/50.0041
2	5	2 x 5 = 10	2/50	2.0001	2.0001 x 5 = 10.0005	2.0001/50.0041
3	2	3 x 2 = 6	3/50	3.0001	3.0001 x 2 = 6.0002	3.0001/50.0041
			------------			-------------
Total freq*obs			50			50.0041

Figure 4: Good-Turing smoothing for the toy corpus

Tokens in corpus: 50

Types in corpus: 41

Recalculated_frequency = $(original\_frequency + 1) * \frac{Number\_of\_observations_{bin\_one\_higher}}{Number\_of\_observations_{current\_bin}}$

Original frequency	times observed	Sum	Freq/ total	frequency basic Good-Turing	frequency algorithm GT
0	0	0 x 0 = 0	0/50	(p = 34/50 = .68)	(p = . 68)
1	34	1 x 34 = 34	1/50	(1+1)*5/34 = .294	.26
2	5	2 x 5 = 10	2/50	(2+1)*2/5 = 1.2	.83
3	2	3 x 2 = 6	3/50		1.50
		------------		-------------	-------
Total freq*obs			50	16	16

Figure 5: The outcome of the Good-Turing algorithm for the first 20 frequencies of Kucera and Fancis (1967)

Zero mass: p = .022

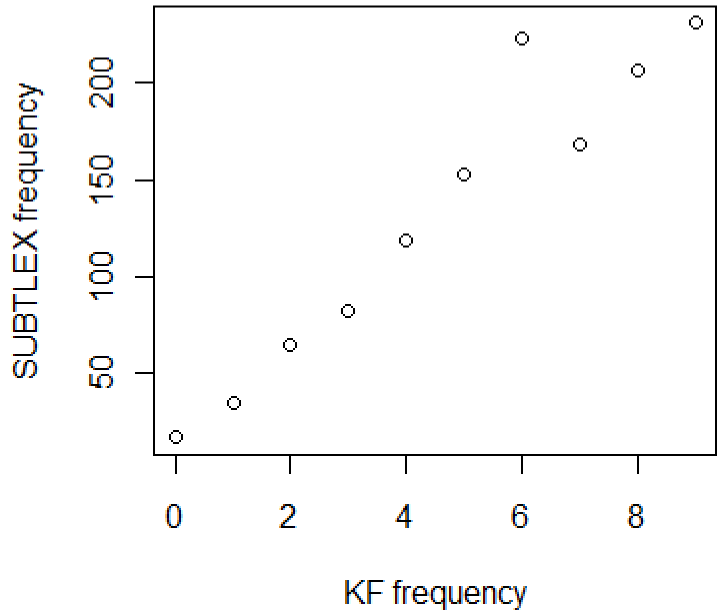
KF_Freq	Nobs	GT_estimate
1	22529	0.64
2	7230	1.64
3	3946	2.50
4	2463	3.71
5	1820	4.24
6	1279	5.22
7	1121	6.21
8	825	7.20
9	695	8.20
10	559	9.19
11	498	10.19
12	434	11.18
13	390	12.18
14	324	13.18
15	301	14.18
16	314	15.18
17	255	16.18
18	221	17.18
19	198	18.18
20	207	19.18

Figure 6: The outcome of the Good-Turing algorithm for the first 20 frequencies of SUBTLEX (Brysbaert & New, 2009)

Zero mass: p = .00028

SUBTLEX_Freq	Nobs	GT_estimate
1	13902	0.99
2	6870	1.99
3	4554	2.84
4	3231	4.02
5	2599	4.88
6	2114	5.87
7	1773	7.12
8	1578	7.45
9	1396	8.44
10	1232	9.44
11	1186	10.43
12	1078	11.43
13	897	12.43
14	862	13.42
15	761	14.42
16	749	15.42
17	667	16.42
18	638	17.42
19	593	18.42
20	589	19.42

Figure 7: Estimating the KF frequencies on the basis of SUBTLEX and Google Books



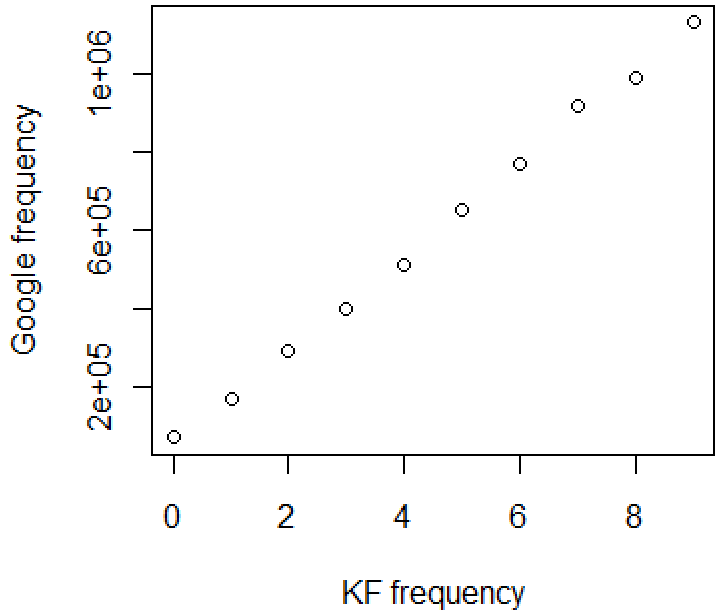


Figure 8: Estimating the SUBTLEX frequencies on the basis of Google Books

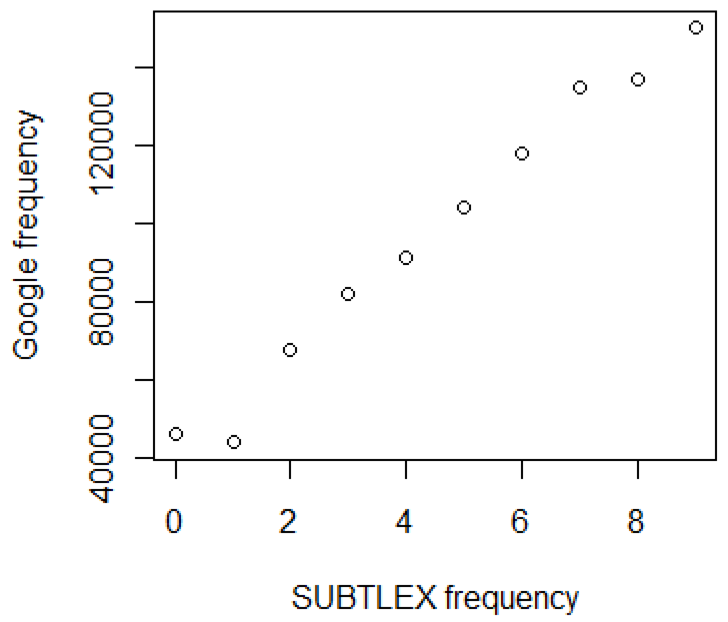


Figure 9: The relationship between KF frequency transformations (log10) and lexical decision times in megastudies (the circles are ELP times; the diamonds are BLP times, which are considerably faster)

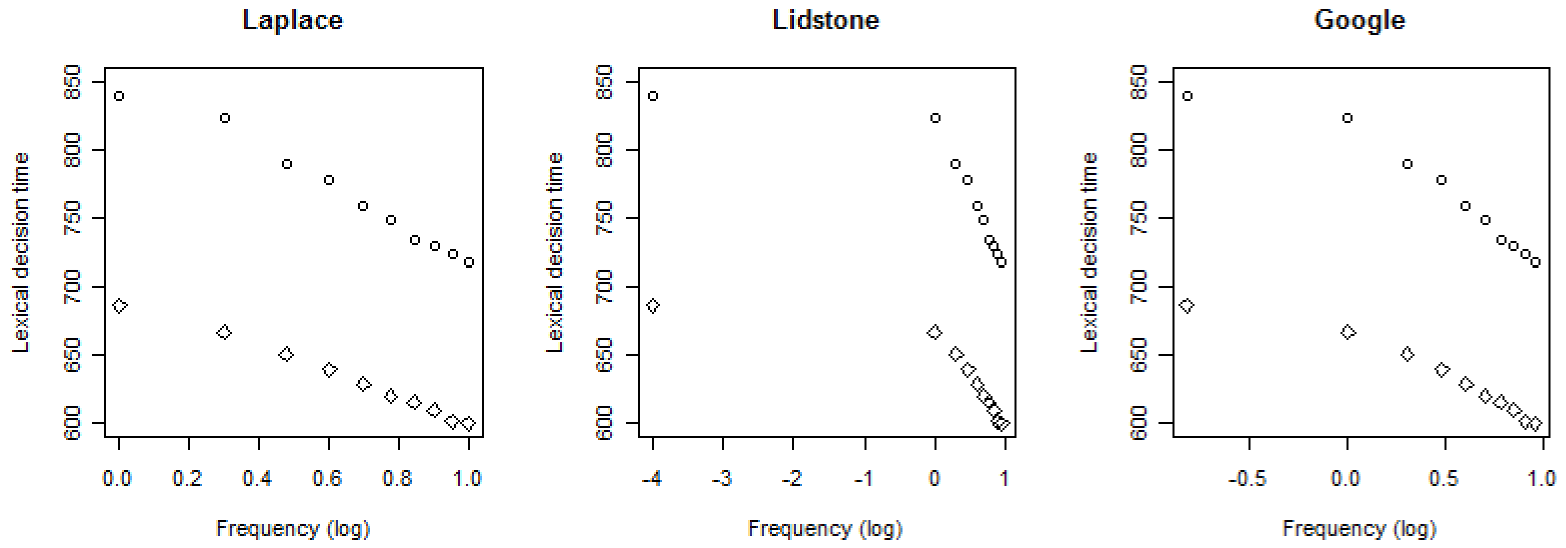
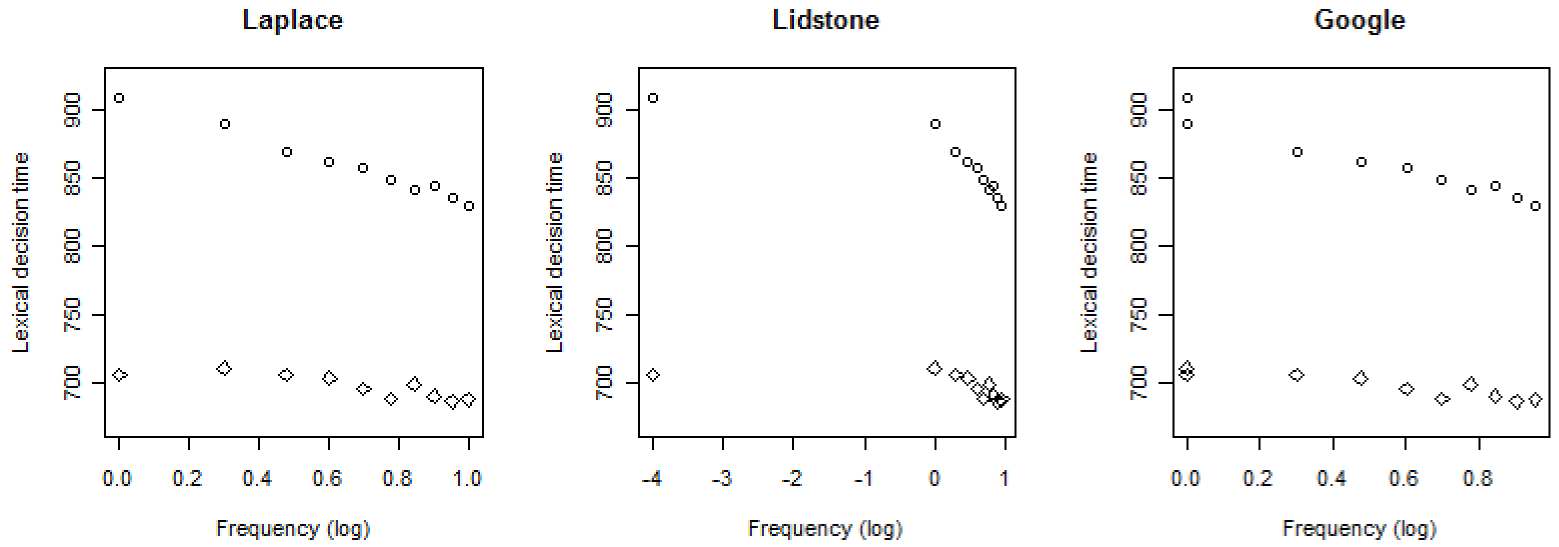


Figure 10: The relationship between SUBTLEX frequency transformations (log10) and lexical decision times in megastudies (the circles are ELP times; the diamonds are BLP times, which are considerably faster)


Laplace
Lidstone
Google
Lexical decision time
Frequency (log)