Journal of Young Investigators

Research

Finding Probable Frequency Sums to Reduce the Key Space of Homophonic Substitution Ciphers

Floe Foxon[1*]

Homophonic substitution ciphers have long pervaded brute-force decryption attempts due to the astronomical number of possible keys. The aim of this study is to identify only those combinations of homophones most likely to represent a given plaintext character, in order to reduce the key space of the cipher and increase the efficiency of brute-force decryption attempts. Plaintext data from selected English language novels were used to calibrate a normalised frequency model of English plaintext characters with upper and lower bounds. The normalised frequencies of combinations of homophones within homophonic substitution ciphers were then compared to these bounds to determine which homophone combinations were most probably used to encrypt a given plaintext character. The key space may be reduced in this way by up to 10 orders of magnitude, increasing the efficiency of brute-force attacks, but the reduced key space is still too large for exhaustive decryption with modern hardware in a reasonable time. However, this may prove surmountable with future hardware.

INTRODUCTION

Substitution ciphers are a form of encryption whereby each character in a plaintext message is substituted for a corresponding cipher character. These cryptograms have existed in various forms for millennia (Whitman and Herbert, 2017) and, for some time, were secure means of exchanging data and information due to the nature of their encryption (Singh, 2000).

For a plaintext alphabet of 26 characters like the English language, a monoalphabetic substitution cipher key which maps each of these to different singular cipher characters (e.g. a ➝ e, b ➝ h, c ➝ z, etc.) is concealed by the vast size of the key space, or the number of possible mixed alphabet cipher keys. In this case, the key space is 26! ≈ 4×10^{26}. A single machine capable of one billion decryption attempts per second would take ~13 billion years to brute-force every solution of such a cipher, which implies trying every possible key exhaustively. This number is comparable to the age of the observable universe (Planck Collaboration et al., 2016). Clearly, brute-force decryption attempts are an inefficient means of decryption.

Fortunately, patterns in the distribution of letter frequencies in languages may be used to immediately identify which cipher characters are likely to represent which plaintext characters. For example, the letter ‘e’ is the most frequent letter in English text (Friedman, 1938), therefore the most frequent cipher character in a given monoalphabetic substitution cipher text likely represents the letter ‘e’. This frequency analysis approach may be repeated for all character frequencies in the cipher text for complete decryption.

Homophonic substitution ciphers circumvent frequency analysis by mapping each character in a plaintext alphabet to multiple cipher characters (e.g. a ➝ {e, !}, b ➝ {~, q}, c ➝ {z, X}, etc.). In this way, the ciphertext alphabet is greater than the plaintext alphabet, and the frequency distribution of ciphertext characters does not immediately resemble that of the underlying plaintext language.

For a cipher of this kind with two unique homophones mapped to each character of the plaintext English alphabet, there are ~1.2×10^{60} possible keys. A machine capable of one trillion decryption attempts per second would take ~3.8×10^{40} years to brute-force the entire key space. Decrypting such a cipher exhaustively in a reasonable amount of time is far beyond the current technological grasp of cryptanalysts (Diffie and Hellman, 1977), and consequently many homophonic substitution ciphers remain unsolved (Kahn, 2005).

Arguably, the strongest decryption method for homophonic substitution ciphers is the hill-climbing algorithm (Dhavare et al., 2013), wherein a parent key is generated and used to decrypt the ciphertext, and the fitness of this decryption attempt is measured. The key is then modified, and another decryption attempt is made with this modified key. If the fitness of the modified attempt is better than the initial attempt, the modified key is carried forward; otherwise,

Address correspondance to:
[1]School of Physics and Astronomy, University of Nottingham, Nottingham, NG7 2RD, UK
*ppyss10@nottingham.ac.uk

doi:10.22186/jyi.38.1.1-5

Acceptance date: November 2019
Publication date: July 2020

the initial key is carried forward. This process is repeated iteratively until an optimal solution is reached. The popular stochastic hill-climbing technique incorporates the random generation of the first parent key and makes random modifications thereafter for child keys. This technique is far more efficient than brute-force decryption methods. However, this approach is limited; iterations here may only lead to a locally optimal solution, as opposed to the globally optimal solution (Dhavare et al., 2013).

Despite being a less efficient approach, exhaustive homophonic substitution cipher decryption may be achieved in polynomial time, placing this in the P complexity class of decision problems, meaning they can be solved in practice. This motivates novel analyses to improve the efficiency of such methods.

This paper presents a theoretical approach at improving said efficiency by forming a subset of possible keys more likely to contain the encryption key, thus reducing the time taken for brute-force attempts significantly. The aims of this study are to calibrate a normalised frequency model of plaintext characters with statistically acquired upper and lower bounds and to compare this model with combined frequencies of homophones to identify which homophone combinations most probably represent each plaintext character. This represents a rigorous and novel approach to aiding exhaustive decryption of homophonic substitution ciphers which may be utilized with future hardware.

METHODS

Sample

To calibrate a normalised frequency model, plaintext data were taken from ten classic English language novels: 'Moby-Dick or The Whale' by Herman Melville, consisting of 935,668 characters; 'Dracula' by Bram Stoker, consisting of 639,465 characters; 'Journey to the Center of the Earth' by Jules Verne, consisting of 379,042 characters; 'Frankenstein or The Modern Prometheus' by Mary Shelley, consisting of 332,111 characters; 'The War of the Worlds' by H. G. Wells, consisting of 265,613 characters; 'At the Mountains of Madness' by H. P. Lovecraft, consisting of 199,790 characters; 'The Island of Doctor Moreau' by H. G. Wells, consisting of 186,990 characters; 'Strange Case of Dr Jekyll and Mr Hyde' by Robert Louis Stevenson, consisting of 107,198 characters; 'The Metamorphosis' by Franz Kafka, consisting of 93,340 characters and 'The Murders in the Rue Morgue' by Edgar Allan Poe, consisting of 62,224 characters. These novels were chosen because of their status as public-domain texts, making them readily analysable.

To produce homophonic substitution ciphertext, the short story 'Dagon' by H. P. Lovecraft, consisting of 9972 characters, was used. This ciphertext is written in similar writing style to the texts used to calibrate the model; therefore, this model is well-suited to the target text. If the target text were of a different style, then the model would need to be calibrated with correspondingly similar text instead.

The raw text files for each of these were filtered to remove all numbers and punctuation, as well as all whitespace characters (spaces, tabs, and paragraphing). Additionally, all characters were converted to lowercase, leaving only the 26 lowercase letters of the English alphabet in the source texts. This pre-processing was done in order to make the sources resemble typical cryptogram plaintext.

The decision to use a range of texts of various length and authorship is justified by the need for a statistical spread in the normalised frequencies used to construct a representative normalised character frequency model.

Measures

For each source text, the frequency of each plaintext character was found along with the total number of characters in the source. This enabled ten samples of normalised character frequencies to be acquired for each letter in the English alphabet. Weighted mean normalised character frequencies were then found with their weighted standard deviations, building a model of typical normalised character frequencies in English plaintexts.

For comparison, the same measurements were taken on the 'Dagon' plaintext before encryption. Two kinds of ciphertexts were generated from 'Dagon' with two and three homophones per plaintext character respectively, and all possible combinations of two and three homophones were found from the ciphertext alphabets with their normalised frequencies.

Analysis

The normalised character frequencies were given by $f_{norm} = \frac{f_a}{N}$, where f_a is the frequency (number of occurrences) of a given plaintext character, and N is the total number of characters in the plaintext.

The weighted mean normalised character frequencies were then estimated as $\bar{f}_{norm} = \frac{\sum_{i=1}^{10} w_i \cdot f_{norm,i}}{\sum_{i=1}^{10} w_i}$, where $f_{norm,i}$ are the normalised character frequencies of a given character from a plaintext source i, and w_i are their respective weights; the length of the plaintext source. This weighting is justified because a longer plaintext arguably provides a more representative statistical distribution of plaintext characters for a given language.

The standard deviations on the weighted means were then found as $SD_{\bar{f}_{norm}} = \sqrt{\frac{\sum_{i=1}^{10} w_i (f_{norm,i} - \bar{f}_{norm})^2}{\frac{(10-1)}{10} \sum_{i=1}^{10} w_i}}$.

Plotting the normalised character frequencies gave a model detailing the character distribution of the English language, which was used in comparison against ciphertext distributions.

The expected normalised frequency of an English plaintext character was then estimated with the means and standard deviations in the following way: the upper bound of the expected normalised character frequency was given by

upper = $\bar{f}_{norm} + (m \cdot SD\bar{f}_{norm})$, and the lower bound by lower = $\bar{f}_{norm} - (m \cdot SD\bar{f}_{norm})$, where $m = 2$ and $m = 3$ were tried separately. This was justified because approximating the distribution of frequencies as Gaussian, ~95% and ~99.7% of the values should fall within ± two and ± three standard deviations of the mean respectively, meaning the bounds should enclose the correct homophone combinations.

The first 'Dagon' ciphertext utilised a cipher alphabet consisting of the lowercase and uppercase English alphabet, for a total of 52 cipher characters such that each plaintext character was replaced by two unique homophones. The second utilised a cipher alphabet consisting of the lowercase and uppercase English alphabet, as well as the numbers 0 through 9 and punctuation characters, for a total of 78 cipher characters such that each plaintext character was replaced by three homophones.

To create the ciphertext, each plaintext character in 'Dagon' was pseudo-randomly assigned the appropriate number of homophones, and each instance of a given plaintext character was pseudo-randomly replaced with one of its assigned homophones to produce an apparently random ciphertext. The receiver could then decrypt the ciphertext with the key and knowledge of which pseudorandom number generator was used and with which initial seed (starting parameters).

The double and triple homophones ciphertexts were then analysed by finding the normalised cipher character frequencies and plotting these for a visual comparison to the English language model found previously.

The number of possible combinations of homophones are given by the combination formula $C(n, r) = \frac{n!}{(n-r)! \cdot r!}$ such that there are $C(52,2) = 1326$ combinations of two homophones in a 52 character cipher alphabet, and $C(78,3) = 76{,}076$ combinations of three homophones in a 78 character cipher alphabet. Nested loops were used to identify all of these possible combinations in each respective ciphertext.

The normalised frequencies for each of these combinations were then found by summing the normalised character frequencies of their constituent characters, e.g. for a double homophone mapping a ➔ {e, u}, the normalised character frequency of the double homophone {e, u} was found by summing the normalised character frequencies of 'e' and 'u' in the ciphertext. Each normalised double and triple homophone frequency was then compared to the expected normalised frequency bounds of English characters from the model. In doing so, this gives, for each possible homophone combination, the plaintext characters it is most likely being substituted for, reducing the key space of the cipher.

Due to the possibility for homophone combinations to fall between the bounds of more than one English plaintext letter's expected frequency, it is difficult to estimate the reduced key space. However, an overestimate can be obtained as the product of the number of probable combinations for each letter in the plaintext alphabet, which gives the total number of 'keys' including those with cipher characters mapped to more than one plaintext character (which would not be done in practice).

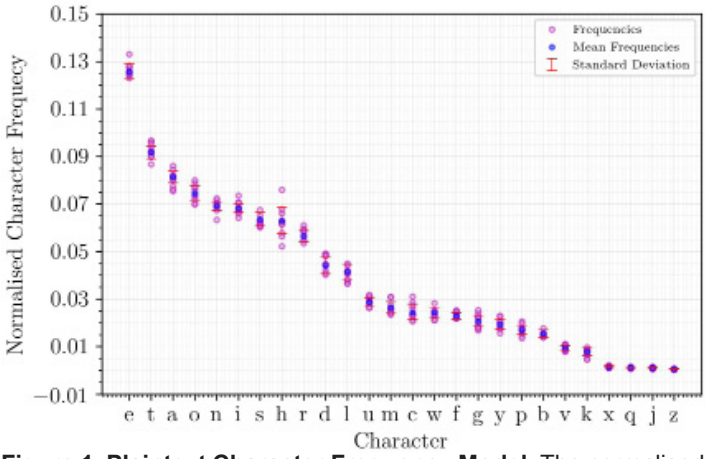

Figure 1. Plaintext Character Frequency Model. The normalised frequencies (number of instances of given character ÷ total number of characters) for English language plaintext characters, based on data from classic English language novels. The means and standard deviations are weighted by the length of their sources.

These reduced key space estimates were compared to the key spaces of double and triple homophone ciphers with no such filtering, which are $C(52,2) \cdot C(50,2) \cdot C(48,2) \cdot \ldots \cdot C(4,2) \cdot C(2,2) \approx 1.2 \times 10^{60}$ and $C(78,3) \cdot C(75,3) \cdot C(72,3) \cdot \ldots \cdot C(6,3) \cdot C(3,3) \approx 3.7 \times 10^{91}$, respectively.

All analyses demonstrated in this paper were performed with Python scripts, using the NumPy package for data storage and mathematical operations, and the Matplotlib package for plotting results. Pseudo-randomisation was achieved with the random package.

RESULTS

The model of expected normalised frequencies of English characters is shown in Figure 1. The letters of English language plaintext are distributed such that 'e' is the most frequent letter with a mean normalised frequency of 0.126 ± 0.003, and 'z' is the least frequent with a mean normalised frequency of 0.0005 ± 0.0001. Every other letter falls between these frequencies tracing a monotonic curve. Interestingly, the letter 'h' demonstrates the most spread in frequency between source texts, whereas the four least frequent characters, 'x', 'q', 'j', and 'z' demonstrate very little spread.

The normalised character frequencies of the 'Dagon' plaintext are shown in Figure 2. Most characters fit within the bounds determined by the model. This demonstrates that the model is mostly well calibrated and a cryptanalyst using this model would have a firm understanding of the underlying distribution, without knowing the true values of the frequencies.

The normalised character frequencies of the 'Dagon' ciphertexts are shown in Figure 3a and Figure 3b. Upon initial

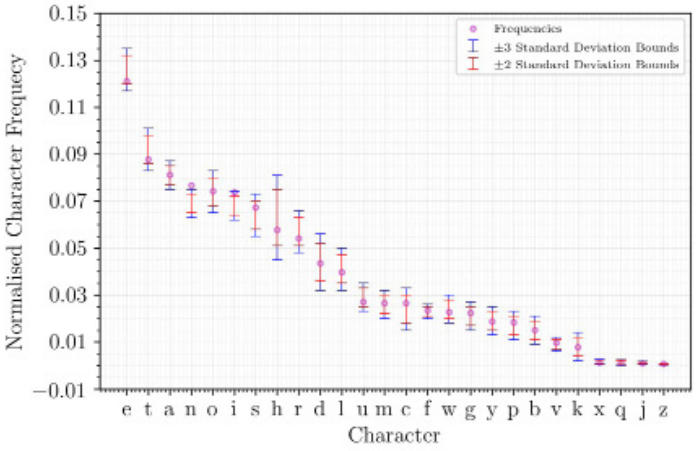

Figure 2. 'Dagon' Plaintext Character Frequency Analysis. The normalised frequencies (number of instances of given character ÷ total number of characters) for the 'Dagon' plaintext characters. The expected bounds span two and three times the standard deviation on the expected mean normalised character frequencies from the plaintext character frequency model respectively and are centred on those means.

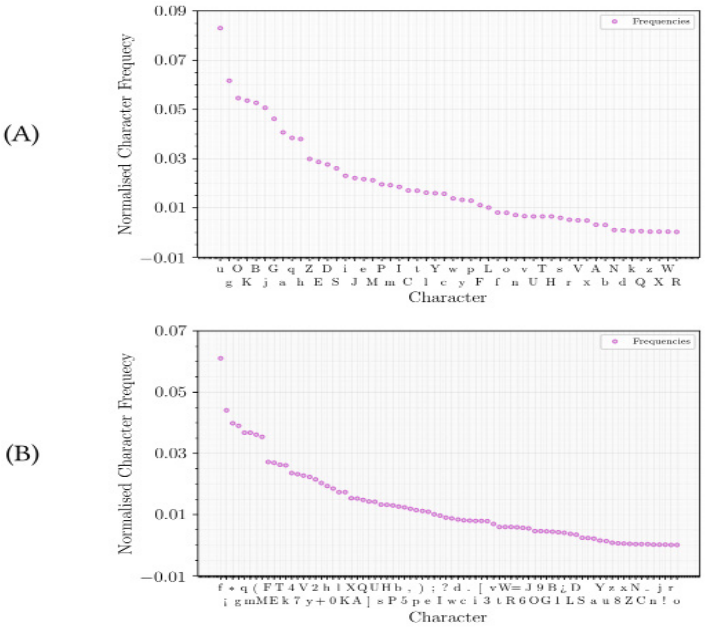

Figure 3. 'Dagon' Ciphertext Character Frequency Analyses. The normalised frequencies (number of instances of given character ÷ total number of characters) for the 'Dagon' ciphertext characters when encrypted using a. two homophones per plaintext alphabet character, and b. three homophones per plaintext alphabet character.

inspection, Figure 3a and Figure 3b somewhat resemble the underlying plaintext character distribution of Figure 2, but a closer analysis reveals the similarities are fairly limited; a cryptographer may assume the most frequent ciphertext character encrypts the letter 'e'. If correct this would only reveal the positions of a fraction of the 'e' characters in the plaintext, and it is difficult to ascertain exactly which of the other homophones are being used without knowing the pseudorandom algorithm and seed used to select them.

For the double homophones cipher, the correct homophone pair from the real key was caught in the set of probable combinations for 23 of the 26 plaintext characters when filtering with the ± 2 standard deviation bounds. The key space of the probable combinations set was $\lesssim 4.3 \times 10^{49}$ keys. This is smaller than the key space with no filtering by a factor $\gtrsim 28$ billion (at least 10 orders of magnitude). Filtering with the ± 3 standard deviation bounds allowed the correct homophone pair to be caught in the set of probable combinations for 25 of the 26 plaintext characters, but this increased the key space of probable combinations to $\lesssim 2.9 \times 10^{53}$ keys. This is smaller than the key space with no filtering by a factor of $\gtrsim 4.1$ million (at least 6 orders of magnitude).

For the triple homophones cipher, the correct homophone triplet from the real key was caught in the set of probable combinations for 23 of the 26 plaintext characters when filtering with the ± 2 standard deviation bounds. The key space of the probable combinations set was $\lesssim 1.5 \times 10^{90}$ keys. This is smaller than the key space with no filtering by a factor $\gtrsim 25$ (at least 1 order of magnitude). Filtering with ±3 standard deviation bounds allowed the correct homophone pair to be caught in the set of probable combinations for 24 of the 26 plaintext characters, but this increased the key space of probable combinations to $\lesssim 3.8 \times 10^{94}$ keys. This overestimate is greater than the key space with no filtering.

DISCUSSION

This study aimed to find, in the ciphertext of a homophonic substitution cipher, homophones whose frequency sums lay within the expected frequency bounds of a given plaintext character, thus reducing the key space of the cipher and enabling more efficient brute-force analyses.

The results presented here show that an expected frequency model calibrated with English plaintext novels is able to catch almost all of the correct homophones used to encrypt plaintext characters in sets, and that these sets provide a reduced key space which is up to 10 orders of magnitude smaller than the key space with no such filtering. However, these sets are still relatively large, and finding all the remaining keys and processing these for goodness of fit would still prove an insurmountable task for any modern processor in a reasonable quantity of time.

Increasing the size of the bounds (i.e. the number of standard deviations about the mean plaintext character frequency) used to filter the ciphertext character combinations increased the number of ciphertext character combinations correctly associated with each underlying plaintext character, however this also increases the set of probable combinations for each plaintext character, thus increasing the key space. This presents a trade-off between accuracy and efficiency.

These methods offer a complementary approach to the current best decryption method, the hill-climbing algorithm; the parent keys in the hill-climbing algorithm may be generated from a filtered set as described in this paper for additional efficiency.

Better estimates of the reduced key spaces would allow for a more thorough measure of the effectiveness of the filtering methods presented in this paper, for example the estimated reduced key space of the triple homophones cipher is greater than the unfiltered key space, but logically the reduced key space must be smaller, or at least equal to, the unfiltered. This warrants further study.

Some strengths of this study are the novel method of filtering by expected bounds, and thorough statistical measures such as employing weighted means and standard deviations in the model. However, a number of limitations should be acknowledged: Firstly, at best the correct homophones were associated with 24 out of 26 plaintext characters, meaning that brute-forcing the reduced key space would only recover partial keys and not full keys. However, this may be overcome by using goodness-of-fit measures on each key in the reduced key space to determine the best partial key (i.e. the one with the most correct combination associations). The remaining encrypted characters may then be found with algorithms which compare character strings in the partially decrypted text to an English dictionary. Furthermore, the number of English plaintexts used to calibrate the model was not exhaustive, and future studies may improve on this. Additionally, the distribution of characters in plaintext depends on a number of variables such as the author's writing style and the length of the plaintext, so a plaintext must be sufficiently long and of relevant content in order for the model comparison to work. Therefore, the model may not be generalizable to different styles of text. More significantly, the distribution of cipher characters among the plaintext characters (e.g. two homophones per plaintext character, three homophones per plaintext character, etc.) must be known to those using these methods (which they often are not), and the remaining key space, although reduced, is still large and still must be processed exhaustively. Recent developments in quantum computation such as the Sycamore processor may enable vastly reduced simulation times over the current classical hardware in future research, a concept known as quantum supremacy (Arute et al., 2019).

The use of expected frequency bounds on English plaintext characters can be used to significantly reduce the key space of a homophonic substitution cipher. However, this reduced key space is still too large for exhaustive decryption approaches with current hardware, though future hardware may make better use of these techniques, and improvements may be made to the methods presented in this paper for further efficiency. Additionally, these methods may be used in combination with hill-climbing algorithms.

ACKNOWLEDGMENTS

The author would like to acknowledge their mentor, Dr. Arielle Selya, for continued support and encouragement. The author would also like to thank Dr. Mark Stamp for some interesting discussion on key space equations.

REFERENCES

Arute, F., Arya, K., Babbush, R., Bacon, D., Bardin, J.C., Barends, R., Martinis, J.M. (2019). Quantum supremacy using a programmable superconducting processor. *Nature*, 574, 505-510, available: https://doi.org/10.1038/s41586-019-1666-5.

Dhavare, A., Low. R. M. and Stamp, M. (2013). Efficient Cryptanalysis of Homophonic Substitution Ciphers. *Cryptologia,* 37(3), 250-281, available: https://doi.org/10.1080/01611194.2013.797041.

Diffie, W. and Hellman, M. E. (1977). Exhaustive Cryptanalysis of the NBS Data Encryption Standard. *Institute of Electrical and Electronics Engineers,* 10(6), 74-84, available: https://doi.org/10.1109/C-M.1977.217750.

Friedman, W. F. (1938). *Military Cryptanalysis Part I Monoalphabetic Substitution Systems.* Washington, D.C.: United States Government Printing Office.

Kahn, D. (2005). THE MAN IN THE IRON MASK – ENCORE ET ENFIN, CRYPTOLOGICALLY. *Cryptologia,* 29(1), 43-49, available: https://doi.org/10.1080/0161-110591893753.

Planck Collaboration, Ade, P. A. R., Aghanim, N., Arnaud, M., Ashdown, M., Aumont, J., … Zonca, A. (2016). Planck 2015 results - XIII. Cosmological parameters. *Astronomy & Astrophysics,* 594, A13, available: https://doi.org/10.1051/0004-6361/201525830.

Singh, S. (2000). The Code Book : *The Secret History of Codes and Code-Breaking.* London: Fourth Estate.

Whitman, M. E. and Herbert, J. M. (2017). *Principles of Information Security.* Boston, MA: Cengage Learning.