Cryptanalysis of the Vigenère Cipher

Tu KS MS

1 Introduction

For around three hundred years, the Vigenère Cipher was known as *le chiffre indéchiffrable*: the indecipherable cipher. First described by Giovan Battista Bellaso in 1553, and later credited to Blaise de Vigenère by mistake, it was finally broken in the 19th century by Charles Babbage and, in 1863, by Friedrich Kasiski. In this article, we define the cipher precisely and show how Kasiski's test and William Friedman's index of coincidence can be used to break it.

2 Mathematical Prerequisites

2.1 Cryptosystem

Below is the formal definition of a cryptosystem that we will use throughout the text. The reason we define a cryptosystem so precisely is because it allows us to compare different ciphers, find common properties they share, or perhaps, as what mathematicians like to do, prove results generally. It also describes the necessary ingredients to contriving your own cryptosystems.

Definition 2.1: Cryptosystem

A **cryptosystem** is a 5-tuple $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ satisfying the following conditions:

1. $\mathcal{P}$ is the finite set of all plaintexts, called the plaintext space.
2. $\mathcal{C}$ is the finite set of all ciphertexts, called the ciphertext space.
3. $\mathcal{K}$ is the finite set of all possible keys, called the keyspace.
4. For each key $K \in \mathcal{K}$, there is an **encryption rule** $e_K \in \mathcal{E}$ and a corresponding **decryption rule** $d_K \in \mathcal{D}$.
   Each $e_K : \mathcal{P} \to \mathcal{C}$ and $d_K : \mathcal{C} \to \mathcal{P}$ are functions such that $d_K(e_K(x)) = x$ for every plaintext element $x \in \mathcal{P}$.

Remark. The fundamental property of a cryptosystem is rule 4. It is neat that cryptosystems are determined by functions. The existence of the decryption rule relies on the corresponding encryption rule to be invertible.

2.2 Modular arithmetic

Throughout this text, given positive integer m, let $\mathbb{Z}_m = \{0, 1, \ldots, m-1\}$.

Addition in $\mathbb{Z}_m$ can be explicitly written, given integers $a, b \in \mathbb{Z}_m$, as $(a+b) \bmod m$, which is the remainder of $(a+b)$ divided by m. The remainder of any integer divided by m is always in $\mathbb{Z}_m$. Subtraction is defined in the same way, as $(a-b) \bmod m$, taking the remainder in $\mathbb{Z}_m$ even when $a-b$ is negative.

2.3 Numerical representation of the alphabet

It is difficult to work with letters mathematically. However, we can easily establish a one-to-one correspondence between an m-letter alphabet and $\mathbb{Z}_m$. Take the 26-letter English alphabet, for example:

A	B	C	D	E	F	G	H	I	J	K	L	M
0	1	2	3	4	5	6	7	8	9	10	11	12

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
13	14	15	16	17	18	19	20	21	22	23	24	25

Remark. We will be focusing on the cryptanalysis of encrypted English messages, which means that we will be working in $\mathbb{Z}_{26}$. It is clear, though, that this one-to-one correspondence between the alphabet and numbers can be extended to other languages.

3 Two Classical Ciphers

Definition 3.1: Shift (Caesar) cipher

Let $\mathcal{P} = \mathcal{C} = \mathcal{K} = \mathbb{Z}_{26}$. For any key $K \in \mathcal{K}$, define

$$e_K(x) = (x + K) \bmod 26$$
$$d_K(y) = (y - K) \bmod 26$$

Note: $x \in \mathcal{P}$, $y \in \mathcal{C}$

The Shift Cipher is one of the most notable ciphers (since it is easily understood), where the plaintext letters are simply "displaced" in their alphabet. Although distinct from the Vigenère Cipher, it will turn out to be useful in its cryptanalysis.

For example, the plaintext `irlylovecryptography` with the shift (key) of 3 gives the ciphertext

`LUOBORYHFUBSWRJUDSKB`

Definition 3.2: Vigenère cipher

Let m be a positive integer. Let $\mathcal{P} = \mathcal{C} = \mathcal{K} = (\mathbb{Z}_{26})^m$
For any key $K = (k_1, \ldots, k_m) \in \mathcal{K}$, define

$$e_K(x_1, \ldots, x_m) = (x_1 + k_1, \ldots, x_m + k_m)$$
$$d_K(y_1, \ldots, y_m) = (y_1 - k_1, \ldots, y_m - k_m)$$

where all results are reduced modulo 26.
Note: $(x_1, \ldots, x_m) \in \mathcal{P}$, $(y_1, \ldots, y_m) \in \mathcal{C}$

In words, the Vigenère Cipher takes an m-letter string as plaintext input and encrypts it by adding the plaintext with the key under $\mathbb{Z}_{26}$. This produces an m-letter ciphertext, which can in turn be decrypted into the plaintext by subtracting the same key from it. A plaintext longer than m letters is split into blocks of m letters, each encrypted with the same key, so the key repeats periodically.

For example, the plaintext `irlylovecryptography` with the key `math` gives the ciphertext below; checking this is left as an exercise for the reader.

`UREFXOOLORRWFOZYMPAF`

4 Cryptanalysis of the Vigenère Cipher

The first step in breaking the Vigenère Cipher is often to determine the keylength m. We describe two techniques that will allow us to do this: One is called the **Kasiski test** and the other uses the **index of coincidence**.

4.1 Kasiski test

The Kasiski test is based on the observation that whenever the distance between identical segments of plaintext is a multiple of m, they will encrypt to the same segments of ciphertext, as the key repeats periodically.

For a rather large chunk of text, it is likely the case that this will happen for three-letter segments called **trigrams**, because trigrams like `THE`, `ING`, `AND` are statistically common in the English language.

If the ciphertext contains identical trigrams, there is a good chance they correspond to identical plaintext trigrams as well. This happens when the same segments of the key are used to encrypt the plaintext trigrams. In light of this, we can record the distance between pairwise identical ciphertext trigrams $\delta_1, \ldots, \delta_n$. The distance δ_i is likely a multiple of m, so taking the greatest common factor of $\delta_1, \ldots, \delta_n$ will likely give m or a multiple of m. This reduces the workload of exhaustively searching for the keylength m, which, of course, is easily impractical by hand.

We summarize the Kasiski test below:

Technique: Kasiski test

Given a piece of ciphertext, if it contains identical trigrams that appear pairwise with distances $\delta_1, \ldots, \delta_n$, then it is *likely* that

$$m \mid \gcd(\delta_1, \ldots, \delta_n)$$

That is, the keylength m is a *likely* factor of $\gcd(\delta_1, \ldots, \delta_n)$.

Remark. Note that if the ciphertext contains any n-gram for $n \geq 3$, then we can apply this test. However, this test has severe limitations (note the emphasis on "likely"). It is important to recognize that the Kasiski test provides very promising information about m when it works, but I have encountered many instances where the ciphertext either did not contain identical trigrams or produced false values of δ by coincidence.

The Kasiski test is a good starting point, but it does not always work, and even when it does, we still have to find the key itself.

4.2 Index of Coincidence

This is where the index of coincidence comes in: it gives a more reliable way to find the keylength, and then the key itself.

Definition 4.1: Index of coincidence

Suppose $\boldsymbol{x} = x_1 \ldots x_n$ is an n-letter string. The **index of coincidence** (IOC) of $\boldsymbol{x}$, denoted by $I_c(\boldsymbol{x})$, is defined as the probability that two elements of $\boldsymbol{x}$, chosen at random from two different positions, are identical.

For any text $\boldsymbol{x} = x_1 \ldots x_n$ with frequencies of A, ..., Z denoted by $f_0, \ldots, f_{25}$, respectively, we have

$$I_c(\boldsymbol{x}) = \frac{\sum_{i=0}^{25} \binom{f_i}{2}}{\binom{n}{2}} = \frac{\sum_{i=0}^{25} f_i(f_i - 1)}{n(n-1)}$$

Furthermore, it is well known that some English letters occur more frequently in literature than others, captured in their relative frequencies, generated by frequency analysis of many English texts. Hence, for an "infinitely" long English string, if the relative frequencies of A, ..., Z are denoted by $p_0, \ldots, p_{25}$, respectively, then we can compute its expected IOC:

$$\sum_{i=0}^{25} p_i^2 \approx 0.065$$

To use the IOC for cryptanalysis of the Vigenère Cipher, we exploit the fact that the key repeats periodically every m-letters. Therefore, if we divide the entire ciphertext into m-letter sub-ciphertexts and arrange them in rows, the string formed by grouping letters in one column will be encrypted using the same key, equivalent to a Shift Cipher. We thus obtain m lots of Shift Cipher encrypted columns.

4.2.1 Determining the keylength

Suppose we have a ciphertext string $\boldsymbol{y} = y_1 \dots y_n$. Also suppose that we guess a keylength m. From the procedure outlined above, we divide $\boldsymbol{y}$ into m of sub-ciphertexts $\boldsymbol{y}_1, \dots, \boldsymbol{y}_m$, each corresponding to a unique key from the key tuple:

$$\boldsymbol{y}_i = y_i \, y_{m+i} \, y_{2m+i} \dots \qquad \text{for } i = 1, \dots, m.$$

Note: if the ciphertext is written in rows of m letters, each $\boldsymbol{y}_i$ is a column. The original ciphertext is recovered by reading the rows in order, taking one letter from each $\boldsymbol{y}_i$ in turn.

If the keylength we used is correct, for each sub-ciphertext $\boldsymbol{y}_i$, since it is encrypted using the Shift Cipher, $I_c(\boldsymbol{y}_i)$ will be close to the expected IOC of English language. This is because the individual terms are simply approximate permutations of the p_i^2 terms.

However, if the keylength we used is incorrect, the sub-ciphertext strings will be encrypted using different key values and hence appear statistically much more random. A uniformly random string has IOC:

$$26 \cdot \frac{1}{26^2} = \frac{1}{26} \approx 0.038$$

which is considerably different from 0.065.

We summarize this test below:

Technique: IOC test for keylength m

Let m be a positive integer. Divide the ciphertext into m sub-ciphertexts:

$$\boldsymbol{y}_i = y_i \, y_{m+i} \, y_{2m+i} \dots \qquad \text{for } i = 1, \dots, m.$$

For each $\boldsymbol{y}_i$, if

$$I_c(\boldsymbol{y}_i) \approx 0.065$$

then m is very likely the correct value. Otherwise, m is very likely incorrect.
If m is the correct keylength, its multiples $2m, 3m, \dots$ will also pass this test, so we take the smallest m that does.

Remark. This test is much more reliable than the Kasiski test in the sense that we can always statistically analyze a "long enough" ciphertext string this way. Numbers are also much easier to compare. If you implement this in code, you might be surprised how revealing this test is.

4.2.2 Determining the key

We now introduce a "displaced" version of the IOC that will allow us to determine the key value used to encrypt this sub-ciphertext.

Let $f_0, \dots, f_{25}$ denote the frequencies of A, ..., Z in $\boldsymbol{y}_i$, respectively. If $\boldsymbol{y}_i$ has l letters, then the relative frequencies of A, ..., Z are $\frac{f_0}{l}, \dots, \frac{f_{25}}{l}$ respectively.

We define the quantity M_g that I will call the "displaced" IOC (DIOC); this is not a standard term, but it is a helpful name:

$$M_g = \sum_{i=0}^{25} p_i \frac{f_{i+g}}{l}$$

To determine the key value k_i from M_g, we simply traverse through all values of $g \in \mathbb{Z}_{26}$. The largest value of M_g, which should be close to the expected value of 0.065, indicates the correct shift, and then $k_i = g$. This is because the relative frequency distribution of $\boldsymbol{y}_i$ (addition reduced modulo 26)

$$\frac{f_g}{l}, \ldots, \frac{f_{25+g}}{l}$$

now corresponds to the natural relative frequency distribution of the English language, $p_0, \ldots, p_{25}$.

Thus, M_g is nearly the expected IOC of English language, which is 0.065.

On the other hand, if we use the wrong g, then the resultant M_g will be considerably below 0.065. We provide an informal justification of this below, using a result known as the **rearrangement inequality**.

Proof. Let $p_1 \geq \cdots \geq p_n$ be a probability distribution. Let $q'_1, \ldots, q'_n$ be a permutation of probability distribution $q_1, \ldots, q_n$.

We claim that

$$\sum_{k=1}^{n} p_k q'_k$$

is maximized when $q'_1 \geq \cdots \geq q'_n$.

Consider an arbitrary sum $\sum_{k=1}^{n} p_k q'_k$. We claim we can do at least as well by swapping two q values "out of order".

More precisely, if $q'_s < q'_t$ for $s < t$, then swap q'_s and q'_t. Name the resultant distribution as

$$q''_1, \ldots, q''_n$$

Then

$$\begin{aligned}
\sum_{k=1}^{n} p_k q''_k - \sum_{k=1}^{n} p_k q'_k &= \sum_{k=1}^{n} p_k (q''_k - q'_k) \\
&= p_s(q'_t - q'_s) + p_t(q'_s - q'_t) \\
&= (p_s - p_t)(q'_t - q'_s) \\
&\geq 0
\end{aligned}$$

This means each swap can only ever increase the resultant sum. By performing this kind of swap we ultimately terminate at the final state

$$q'_1 \geq \cdots \geq q'_n$$

Applying this in context: the correct shift lines up the frequencies of $\boldsymbol{y}_i$ with $p_0, \ldots, p_{25}$ in the right order, giving the largest value, close to $\sum p_i^2 \approx 0.065$. Any other shift lines them up in a different order, and so can only give a smaller value; in practice it is usually much closer to 0.038. □

We summarize the IOC test below:

Technique: IOC test for key $(k_1, \ldots, k_m)$

Suppose we know the keylength m.

Divide the ciphertext $\boldsymbol{y}$ into m sub-ciphertexts:

$$\boldsymbol{y}_i = y_i\, y_{m+i}\, y_{2m+i} \ldots \qquad \text{for } i = 1, \ldots, m.$$

For each sub-ciphertext $\boldsymbol{y}_i$ of length l, find the corresponding key value $k_i = g$ by choosing the g that makes the DIOC largest, close to the expected value 0.065:

$$M_g = \sum_{i=0}^{25} p_i \frac{f_{i+g}}{l}$$

We eventually obtain $(k_1, \ldots, k_m)$ as the key.

Unlike the Kasiski test, this test almost always works and can be easily implemented as code. It is also pretty efficient, for m and g are typically small.

5 Conclusion

The Vigenère Cipher resisted attack for centuries, yet its weakness is simple: because the key repeats, the ciphertext splits into m Shift Ciphers, each of which still carries the letter frequencies of English. Once we know this, the Kasiski test and the index of coincidence let us find first the keylength and then the key. Modern ciphers are designed precisely so that no statistical pattern like this survives encryption.

6 Challenge

Try to cryptanalyse this quote from a famous mathematician using the techniques in this article (without AI). It was encrypted with the Vigenère Cipher, with all spaces and punctuation removed:

Ciphertext

```
RMMTUYVAGZCUAAFRKRRPMIANNRBIABORNRSNDAWEE
IOPNKTQRAMRFUZSBAGNNRAJADEZIAECVRYAAYWTGY
AZTUYRRFZTUSOYLAHJEFHRSJRRDAPEJCCHVUEMSNG
JTUVMMTVWRAAFNFHRICHRIHMNQBJSAFMMTRLRAYKO
IOEEFIGYBGTVXNAFRNPSBBRSCRTFEEHBAEVLUKRFH
TBCAETYIWGRISUNPYRDRRSIENLUEFJWUTUNRMRKHM
NJIADFKHQMNNQEZRTUCVUWSCRTFEEHBLVBEFHRJJI
AKEDSBLCHRGOQTFGDSGSENENOCISLLFHRCMENJLUK
RNQEPFLAUEMXRGYEIOEXBMHJTRIGNXGRKHQRVHJHN
IMANVIDSJRYNENOCYVJTTESCASGKEETGBNRRZSZOC
YAMNEEZTCFJCRZNFHRQXRYUFARHAUYZRTTEZUCIPJ
```

The solution can be found at the end of this edition.

References

"Cryptosystem." *Wikipedia*, Wikimedia Foundation. `https://en.wikipedia.org/wiki/Cryptosystem`.

"Letter Frequency." *Wikipedia*, Wikimedia Foundation. `https://en.wikipedia.org/wiki/Letter_frequency`.

Lyons, James. "English Letter Frequencies." *Practical Cryptography*. `http://practicalcryptography.com/cryptanalysis/letter-frequencies-various-languages/english-letter-frequencies/`.

"Modular Arithmetic." *Wikipedia*, Wikimedia Foundation. `https://en.wikipedia.org/wiki/Modular_arithmetic`.

Stinson, Douglas, and Maura Paterson. *Cryptography: Theory and Practice*. 4th ed., Taylor & Francis Group, 2018.

"Vigenère Cipher." *Wikipedia*, Wikimedia Foundation. `https://en.wikipedia.org/wiki/Vigen%C3%A8re_cipher`.