# 1 Codebreaking

Continued from previous section.

## 1.1 Known-Plaintext Attack on Simple Substitution

So far, we've studied a few techniques for conducting ciphertext-only attacks on various ciphers, i.e., ciphers where the only information Eve knows to begin with is which cipher was used to encrypt a message and has no information about the message itself.

Now, let's suppose Eve *does* have some partial information about the plaintext itself. Specifically, we'll consider the situation where Eve already knows that a certain word appears at least once in the plaintext. As it turns out, we can use the $G$-test statistic to help us make good guesses here.

Suppose, for example, Eve intercepts some ciphertext, known to be encrypted using a **simple substitution**:

```
CUXQAUDRERAUFJUCKRACUXQAUDRXFGAUFJUCKRACUXQAUDRQWRFJXCAOFKCUXQAUDRQWR
FJJFFSCADZRAACUXQAUDRRNFYDFJERSCRJCUXQAUDRRNFYDFJCZYGROMSCUPCUXQAUDRA
RQAFZFJSCWDUCUXQAUDRARQAFZFJOQGTZRAACUXQAUDRANGCZWFJDFNRCUXQAUDRXCZUR
GFJORANQCGXRDQORLRGPUDCZWERJFGRMAXRDQOZFUDCZWERJFGRMAXRXRGRQSSWFCZWOC
GRYUUFDRQLRZXRXRGRQSSWFCZWOCGRYUUDRFUDRGXQPCZADFGUUDRNRGCFOXQAAFJQGSC
TRUDRNGRARZUNRGCFOUDQUAFKRFJCUAZFCACRAUQMUDFGCUCRACZACAUROFZCUAERCZWG
RYRCLROJFGWFFOFGJFGRLCSCZUDRAMNRGSQUCLRORWGRRFJYFKNQGCAFZFZSPUDRGRXRG
RQTCZWXCUDQSQGWRHQXQZOQIMRRZXCUDQNSQCZJQYRFZUDRUDGFZRFJRZWSQZOUDRGRXR
GRQTCZWXCUDQSQGWRHQXQZOQIMRRZXCUDQJQCGJQYRFZUDRUDGFZRFJJGQZYRCZEFUDYF
MZUGCRACUXQAYSRQGRGUDQZYGPAUQSUFUDRSFGOAFJUDRAUQURNGRARGLRAFJSFQLRAQZ
OJCADRAUDQUUDCZWACZWRZRGQSXRGRARUUSROJFGRLRGCUXQAUDRPRQGFJFMGSFGOFZ...
```

Now, more importantly, suppose Eve already knows that the word `LONDON` occurs at least *once* in the plaintext. How does the codebreaking process change?

1. Notice how, in the word `LONDON`, the first 4 letters are all distinct, the 5th letter is the same as the 2nd letter, and the 6th letter is the same as the 3rd letter. This pattern will be **preserved** by simple substitution, so somewhere in the ciphertext, we should find that same sequence. If we can find such a sequence, it suggests that the letters `L`, `O`, `N`, `D` should be matched.

   There are 18 sequences in this ciphertext[1] that fits this pattern

   ```
   SCUPCU      GKROKR
   RACZAC      MUFJUF
   OFGJFG      SXQAXQ
   ZUDRUD      SFZOFZ
   ZUDRUD      UQZOQZ
   SFZOFZ      SFZOFZ
   ARZURZ      RZUCZU
   UDCZDC      GCZWCZ
   RAFJAF      QZWCZW
   ```

   If we were just brute-forcing through all possible substitutions, the number of possibilities for matching the letters `L`, `O`, `N`, and `D` would be

   $$26 \cdot 26 \cdot 24 \cdot 23 = 358800,$$

   which is already significantly less than if we didn't know about the pattern beforehand.

[1] Not all patterns may show up above in the ciphertext; most of the ciphertext has been omitted for brevity.

2. The key observation is that the relative frequencies of the letters `L`, `O`, `N`, and `D` in some sample of plaintext English should match the relative frequencies of the corresponding ciphertext letters. We can measure the deviations between frequencies using $G$, so we can compute $G$ for each of the 15 possible matchings and use that to help guide our choices. How do we calculate $G$? Let's take a look at one of these calculations in detail.

   - Start by looking at some sample English text (e.g., the Declaration of Independence) and throwing out all the letters *except* `L`, `O`, `N`, and `D`. If we do so, we find the following distributions:

| | `L` | `O` | `N` | `D` | Total |
|---|---|---|---|---|---|
| Count in Sample | 228 | 513 | 483 | 252 | 1476 |
| Percent in Sample | 15.4% | 34.8% | 32.7% | 17.1% | 100% |

   Let's suppose we think the first sequence (out of all possible sequences above), `SCUPCU`, corresponds to `LONDON`. This would mean that `S` is matched to `L`, `C` to `O`, `U` to `N`, and `P` to `D`. We can then go through our ciphertext and count the instances of `S`, `C`, `U`, and `P` (throwing out all other letters as usual) and then fill in the first row of the table below:

| | `S` | `C` | `U` | `P` | Total |
|---|---|---|---|---|---|
| Observed in Ciphertext | 146 | 293 | 429 | 83 | 951 |
| Percent in Sample | 146.9 | 330.5 | 311.2 | 162.4 | 951 |

   Here, the expected values were obtained by taking the percent in sample from the first table and then multiplying those percentages by the observed values in the ciphertext. To see why this is the case, note that

   - We saw that 15.4% of the letters `L`, `O`, `N`, `D` should be `L`.
   - If `L` corresponds to `S` in the ciphertext, then we would expect

$$0.154 \cdot 951 \approx 146.9$$

     letters of the ciphertext to be `S`.

   In any case, we can now compute the $G$-value;

$$\begin{aligned} G &= 2 \sum O \ln\left(\frac{O}{E}\right) \\ &\approx 2\left(146 \ln\left(\frac{146}{146.9}\right) + 293 \ln\left(\frac{293}{330.5}\right) + 429 \ln\left(\frac{429}{311.2}\right) + 83 \ln\left(\frac{83}{162.4}\right)\right) \\ &\approx 91.6. \end{aligned}$$

   (Exercise.) The values of $G$ are approximated by a chi-square distribution with $k$ degrees of freedom. What is $k$? What is the value of the integral

$$\int_{91.6}^{\infty} f_k(x)dx,$$

   i.e., the $p$-value? Based on this, does it seem likely that `SCUPCU` is a correct match for London?

   We have four possible observations to make; either one for `S`, `C`, `U`, or `P`. Let $n = 4$ so that $k = n - 1 = 4 - 1 = 3$.

   We can then do analogous calculations of $G$ for the other 14 possibilities. The results are as follows:

| Ciphertext Match | $G$ |
|---|---|
| SCUPCU | 91.6 |
| RACZAC | 602.9 |
| OFGJFG | 34.5 |
| ZUDRUD | 442.5 |
| SFZOFZ | 7.8 |
| ARZURZ | 160.5 |
| UDCZDC | 354.4 |
| RAFJAF | 597.0 |
| GKROKR | 490.8 |
| MUFJUF | 39.3 |
| SXQAXQ | 284.8 |
| UQZOQZ | 223.0 |
| RZUCZU | 444.8 |
| GCZWCZ | 162.4 |
| QZWCZW | 533.3 |

Recall that smaller values of $G$ mean that the distributions are closer together, so the most likely match for `LONDON` appears to be `SFZOFZ`. Of course, it's possible that `SFZOFZ` is not the correct match, but if we make that guess and then it seems like the decryption is turning out to be nonsense, we can just go back and try the one option with the next lowest value of $G$.

As we've seen, it's usually easy to guess which letters correspond to `E` and `T`. So, we now have a systematic way of identifying the letters `L`, `O`, `N`, and `D` as well.

## 1.2 Perfect Secrecy

Let's now discuss cryptosystems in some generality. The point of this section is to show what it means when we say that the one-time pad achieves perfect secrecy.

Let's introduce the general framework.

- From Eve's perspective, it makes sense to consider Alice's choice of a plaintext message as a *random variable* $M$ whose set of values $\mathscr{M}$ represents all possible plaintext messages. By saying that $M$ is a random variable, we're introducing probabilities associated to every possible plaintext message, and we're assuming that Eve knows this probability distribution.
  - For example, $\mathscr{M}$ might be the set of all strings in the letters `A-Z`, and we might deem the plaintext `QUIZ` to be less probable than the plaintext `THEN` on the basis that the former uses much more infrequent letters than the latter.
  - Alternatively, we could assign probabilities based on bigram frequencies instead of single-letter frequencies.

  It doesn't matter *how* we assign probabilities; the point is just that any plaintext message has a probability assigned to it.
- We can model known-plaintext attacks in at least two ways. Suppose, for example, Eve knows that the plaintext must contain the word `LONDON`.
  1. The first way to model a known-plaintext attack is to assign a probability of 0 to every plaintext message that *doesn't* contain the `LONDON` as a substring.
  2. The second way is to simply eliminate all strings that do not contain `LONDON` from our set of values $\mathscr{M}$.

It's more convenient to use the latter method of modeling known-plaintext attacks, since it allows us to assume that

$$\mathbb{P}[M = m] \neq 0$$

for all $m \in \mathscr{M}$, and so we'll assume this from here on out.

All of this brings us to the following definition of a cryptosystem.

**Definition 1.1: Cryptosystem**

A **cryptosystem** (on $M$) consists of the following data:

- A random variable $K$ whose set of values $\mathscr{K}$ represents the set of all possible keys.
- A set $\mathscr{C}$ of all possible ciphertext.
- An encryption function
$$E : \mathscr{K} \times \mathscr{M} \mapsto \mathscr{C},$$
which means that $E$ takes as input a key $k \in \mathscr{K}$ and a plaintext message $m \in \mathscr{M}$ and outputs a corresponding ciphertext $E(k, m) \in \mathscr{C}$.
- A decryption function
$$D : \mathscr{K} \times \mathscr{C} \mapsto \mathscr{M},$$
which means that $D$ takes as input a key $k \in \mathscr{K}$ and a ciphertext $c \in \mathscr{C}$ and outputs a corresponding plaintext message $D(k, c) \in \mathscr{M}$.

In particular, we require that

$$D(k, E(k, m)) = m$$

for any key $k \in \mathscr{K}$ and any plaintext message $m \in \mathscr{M}$, i.e., that a plaintext can be recovered from its encryption. Once we've specified the above data, we also define the random variable $C = E(K, M)$ to model an observation of the ciphertext.

(Example.) Consider the Caesar cipher. The sets $\mathscr{M}$ and $\mathscr{C}$ are both equal to the set of all possible strings in the capital letters `A` through `Z`, and the key string $\mathscr{K}$ is the set of all possible shifts, i.e.

$$\mathscr{K} = \{0, 1, 2, \ldots, 24, 25\}.$$

Given $k \in \mathscr{K}$ and $m \in \mathscr{M}$, the string $E(k, m)$ is the result of applying the Caesar cipher with shift $k$ to $m$, i.e., by adding $k$ mod 26 to the numbers 0-25 corresponding to each letter `A-Z` in $m$. To fully specify the cryptosystem, we should also assign probabilities to each of the keys; for example, we might assume that $\mathscr{K}$ is uniform.

This is not necessarily the only way to fit the Caesar cipher into the above framework. For example, if you like, you can fix an upper bound on the length of the strings involved so that $\mathscr{M}$ and $\mathscr{C}$ are finite. You could also decide that the key space is just $\{1, 2, \ldots, 25\}$, if you don't want to consider a shift of 0 to be a valid key. You could also choose a non-uniform distribution for the random variable $K$.

**Definition 1.2: Perfect Secrecy**

A cryptosystem achieves **perfect secrecy** if $M$ and $C$ are independent random variables.

In other words, for any plaintext message $m$ and any encrypted message $c$, we should have

$$\mathbb{P}[M = m | C = c] = \mathbb{P}[M = m].$$

Heuristically, this means that observing $C = c$ provides Eve with no additional information whatsoever about $M$; the best guesses that she might have made about the plaintext message before she intercepted the ciphertext do not change even after she intercepts the ciphertext.

Recall that the attacks on cryptosystems that we've seen so far exploit the fact that those cryptosystems fail to achieve perfect secrecy. When we conduct frequency analysis on simple substitution, for example, we are using the fact that certain plaintexts become more likely after observing a given ciphertext, e.g., the plaintexts in which the letter `E` appears in the same positions in which the most frequenct letter of the ciphertext appears.

How do we achieve perfect secrecy? The first observation to make here is that achieving perfect secrecy requires having a lot of keys. More precisely,

**Lemma 1.1**

Suppose a cryptosystem achieves perfect secrecy. Then, the number of keys must be at least as large as the number of "possible" ciphertexts (i.e., ciphertexts $c \in \mathscr{C}$ such that $\mathbb{P}[C = c] \neq 0$).

Having observed this, here is a result that gives us a way to achieve perfect secrecy.

**Theorem 1.1: Perfect Secrecy**

A cryptosystem achieves perfect secrecy if it satisfies all of the following conditions:

- $K$ is uniform.
- $K$ and $M$ are independent.
- For every plaintext message $m \in \mathscr{M}$ and every ciphertext $c \in \mathscr{C}$, there exists a unique $k \in \mathscr{K}$ such that $E(k, m) = c$.

We now want to show that the "one-time pad achieves perfect secrecy." Remember that the one-time pad is a special case of the Vigenere cipher. Fix a period $p$ and

- let $\mathscr{K}$ be the set of all sequences $\{a_1, \ldots, a_p\}$, where $a_1, \ldots, a_p \in [0, 1, 2, \ldots, 24, 25]$. Fix a message length $r$, and
- let $\mathscr{M}$ and $\mathscr{C}$ be the set of all strings in the letters `A-Z` of length $r$.
- Also note that if Eve intercepts a ciphertext of length $r$ that she knows to be encrypted using a Vigenere cipher, she also knows the plaintext must have length $r$, so she can assign probability 0 to all messages of other lengths. In other words, it's reasonable to eliminate all messages of other lengths from our message spaces.

We then have the Vigenere encryption and decryption functions $E : \mathscr{K} \times \mathscr{M} \mapsto \mathscr{C}$ and $D : \mathscr{K} \times \mathscr{C} \mapsto \mathscr{M}$. Applying Vigenere encryption to a message of length $r$ will always just use the first $\min\{r, p\}$ of the $p$ numbers in our key sequence, so might as well replace $p$ with $\min\{r, p\}$ in order to assume that $p \leq r$.

**Lemma 1.2**

Using the notation we've just introduced, we have $p = r$ if and only if, for every plaintext message $m \in \mathscr{M}$ and every ciphertext, there exists a unique $k \in \mathscr{K}$ such that $E(k, m) = c$.

With this in mind, recal lsome of the assumptions we made when we were discussing the one-time pad when we first introduced it.

- The key sequence needs to be "totally random." We can now formalize this by saying that we want our random variable $K$ to be uniform.

- The key sequence must be "unrelated to the plaintext." We can now formalize this by saying that $K$ and $M$ be independent random variables.
- The key sequence must be as long as the plaintext. In the notation we just introduced, this is the requirement that $p \geq r$, but we already assumed that $p \leq r$ so in fact this is equivalent to saying $p = r$ and, in the lemma above, we saw that this is equivalent to saying that, for any $m \in \mathscr{M}$ and $c \in \mathscr{C}$, there exists a unique $k \in \mathscr{K}$ such that $E(k, m) = c$.

In other words, these three assumptions we made coincide exactly with the conditions that appear in the Perfect Secrecy Theorem! We have now proved what we want to prove:

**Corollary 1.1**

The one-time paid achieves perfect secrecy.