Friedman attack on Vigenère

This attack illustrates that single-letter frequencies in natural languages combined with some maybe-not-so-intuitive mathematical manipulations can be used to break ciphers.

Kasiski had broken Vigenère, about 1880.

Strangely, Vigenère was still believed to be unbreakable in the early 20th century.

Keep in mind that an even slightly mis-used OTP degenerates into Vigenère, so a mis-used OTP is also completely broken.

Also, Vigenère provides yet another example of a cipher with a huge key space but which is nevertheless completely broken.

We'll study Vigenère with a 26-character alphabet, though the principle remains the same generally.

The Vigenère cipher

The main variant of Vigenère is a OTP with a **periodic** key.

That is, a key for Vigenère is a string of characters which is *repeated* until the string of copies is as long as the message. Then the encryption step adds the i^{th} character of the copied key to the i^{th} character of the plaintext, and reduces mod 26.

In other words, if the key

$$k = k_0 k_1 k_2 \ldots k_{n-1}$$

is of length n then the i^{th} character x_i of the plaintext is encrypted as

$$x_i \to (x_i + k_{i \,\%\, n}) \,\%\, 26$$

Yes, the subscript on the characters of the key is $i \,\%\, n$.

For example, with key $k = \texttt{dog}$ and plaintext $x = \texttt{helloworldoutthere}$ we encrypt by adding vectors modulo 26

$$\begin{aligned} E_k(x) &= && \texttt{helloworldoutthere} \\ & && + \texttt{dogdogdogdogdogdog} \\ &= && \texttt{KSROCCRFRGCAWHNHFK} \end{aligned}$$

where as usual $\texttt{a} \to 0$, $\texttt{b} \to 1$, ..., $\texttt{z} \to 25$. Decryption is by *subtraction* of vectors modulo 26 with the repeated key.

This is a **polyalphabetic substitution cipher** since a given letter of the plaintext can be encrypted in different ways depending where it falls in the message.

A Vigenère certainly messes up inter-letter statistics, and also flattens single-letter frequency statistics.

Since the key can be as long as desired, the keyspace is unlimited.

However, if the key is less than 1/2 the length of the message, *or* if more than a single message is sent with the same key, Friedman's attack breaks Vigenère.

This is so even if the key is highly random itself, since it is *repeated*.

Thus, the only way that Vigenère is safe is if it's actually a legitimate OTP, with fully random key that is never reused. But this has difficulties with both **key generation** and **key distribution**.

Friedman's Index of Coincidence

For two strings of characters of the same length

$$x = x_1 x_2 x_3 \ldots x_n$$

$$y = y_1 y_2 y_3 \cdots y_n$$

the **index of coincidence** $I(x, y)$ is

$$I(x, y) = \frac{\text{no. indices } i \text{ for which } x_i = y_i}{n}$$

If all characters were *equally* likely, with probability 1/26, then the expected number of equalities would be 1/26 of the total, which would be

$$\frac{1}{26} \sim 0.03846$$

However, the skewed statistics of natural languages change this.

For example, the index of coincidence of two chunks of English

```
downaresortsomewheretosendauthors
whohavecompletedprojectstheirowne
xecscouldvactheretootomakethewhol
eendeavorworthwhilebutyesthebilge
thespillsoverafteradissatisfyinge
ndeavorisoftenafrustration
```

```
sometimessleepingatacrucialjunctu
reseemsabletodeflectimpendingilln
essyesalsothoughithastobeabitearl
ierintheprocessgoingforarunsomeso
rtofflushingeffectexamsstrikemeas
pointlesslymeaningless
```

$$= \frac{12}{191} \sim 0.0628$$

This is the typical number for English.

Again, the expected Index of equidistributed strings, or a random string against English, is

$$I(\text{random}, \text{random}) \sim \frac{1}{26} \sim 0.03846$$

$$I(\text{random}, \text{English}) \sim \frac{1}{26} \sim 0.03846$$

The expected Index of English with itself is

$$I(\text{English}, \text{English})$$

$$= P(\texttt{a})^2 + P(\texttt{b})^2 + \ldots + P(\texttt{z})^2$$

$$\sim 0.0628$$

Of course, the Index of a string against itself is always 1, so it is pointless to do exactly this in order to test whether a string is English or random.

Rather, compute an **averaged Index**, as follows, on two strings of length n.

$$I_{\text{avg}}(x, y)$$

$$= \frac{\text{no. } \texttt{a}\text{'s in } x}{n} \cdot \frac{\text{no. } \texttt{a}\text{'s in } y}{n}$$
$$+ \frac{\text{no. } \texttt{b}\text{'s in } x}{n} \cdot \frac{\text{no. } \texttt{b}\text{'s in } y}{n}$$
$$+ \ldots + \frac{\text{no. } \texttt{z}\text{'s in } x}{n} \cdot \frac{\text{no. } \texttt{z}\text{'s in } y}{n}$$

Note that this is a **dot product** or **scalar product** of two 26-dimensional vectors telling the fraction of each character in the two strings.

If $I_{\text{avg}}(x, x)$ is close to $1/26 \sim 0.038$ then it's probably *not* English.

If $I_{\text{avg}}(x, x) \sim 0.063$ this does *not* say that the string x is probably English, but something subtler.

Imagine what would happen if we encrypted an English plaintext with a *monoalphabetic* cipher, that is, where each letter of the alphabet is encrypted the same way throughout the message. Though e itself will no longer occur .11 of the time, whatever charcter it's encrypted to will occur at that rate. The analogue is true for each letter of the alphabet.

Thus, the *set* of single-letter probabilities of monoalphabetically encrypted English is the same as English.

Thus, if $I_{\text{avg}}(x, x) \sim 0.063$ then x *is probably encrypted from English by a monoalphabetic cipher.*

Vigenère is polyalphabetic. But the repeating or periodic nature of the key means that many pieces *are* encrypted by the same monoalphabetic cipher.

For example, with key `dog`, the 0^{th}, 3^{th}, 6^{th}, 9^{th}, … characters are shifted in the alphabet by $3 \sim$ `d`, the 1^{th}, 4^{th}, 7^{th}, 10^{th}, … are shifted in the alphabet by $14 \sim$ `o`, and the 2^{th}, 5^{th}, 8^{th}, 11^{th}, … characters are shifted by $6 \sim$ `g`.

Let $x^{[t]}$ be the same string x but **shifted in positions** forward by t, with wrap-around. Then, if the key is `dog`, the 0^{th}, 3^{th}, 6^{th}, 9^{th}, … characters of **both** x and $x^{[3]}$ are all shift-encrypted by $3 \sim$ `d`, the 1^{th}, 4^{th}, 7^{th}, 10^{th}, … are shifted in the alphabet by $14 \sim$ `o`, and the 2^{th}, 5^{th}, 8^{th}, 11^{th}, … characters are shifted by $6 \sim$ `g`.

That is, with key `dog` of **length 3**, at each position in the strings the character of x is encrypted by the same shift cipher as the corresponding character in $x^{[3]}$, so

$$I(x, x^{[3]}) \sim 0.063$$

And similarly

$$I(x, x^{[6]}) \sim 0.063$$

$$I(x, x^{[9]}) \sim 0.063$$

because $3, 6, 9$ are *multiples of the key length.*

But *it happens that* for shifts $x^{[t]}$ with the position shift t *not* a multiple of the key length,

$$I(x, x^{[t]}) << 0.063$$

For example, with x the second block of English above encrypted with key `dog`

x = `VCSHHOPSYVZKHDOQUGWOIUIILORMI`
`TFHAUSYHSSVOHOSZRRKIZKFHOPDKQROQUO`
`OZTHGYBSYDZYRHNRIMKWZKOYWCHHOHLHKD`
`FRLSXLBZKSVUCIHGYJCOQULRFGUIVCSHG`
`UUHUITRXGNLBMHTLHQZHLGPGYWFONSSHOY`
`SCOQHRHGYOMSHOTLBMOSYVZEFFAHZ`

compute, noting one **bad** number:

$$
\begin{array}{rcll}
I(x, x^{[0]}) & \sim & 1.000 & \\
I(x, x^{[1]}) & \sim & 0.046 & \text{low} \\
I(x, x^{[2]}) & \sim & 0.036 & \text{low} \\
I(x, x^{[3]}) & \sim & 0.072 & \text{high} \\
I(x, x^{[4]}) & \sim & 0.036 & \text{low} \\
I(x, x^{[5]}) & \sim & 0.087 & \textbf{???} \\
I(x, x^{[6]}) & \sim & 0.072 & \text{high} \\
I(x, x^{[7]}) & \sim & 0.015 & \text{low} \\
I(x, x^{[8]}) & \sim & 0.036 & \text{low} \\
I(x, x^{[9]}) & \sim & 0.082 & \text{high} \\
I(x, x^{[10]}) & \sim & 0.041 & \text{low}
\end{array}
$$

But still the preponderance of high numbers are at multiples of 3.

To start the Friedman attack: for various physical shifts $x^{[t]}$ of the ciphertext x, compute

$$I(x, x^{[t]})$$

and conclude that the key length is the **gcd** of the values of t that give the *high* numbers.

Oops, due to probabilistic fluctuations, this will fail.

Instead, compute the **gcd** of the values of t giving high indices *allowing yourself to drop one or more.*

Getting the key

We now know (tentatively) that the key length is 3.

Let the ciphertext characters be

$$x = x_0 x_1 x_2 x_3 \ldots$$

To determine the key, we look at the **slices** of the ciphertext corresponding to the key-length 3

$$\begin{aligned} x^{(0)} &= x_0 x_3 x_6 x_9 \ldots \\ x^{(1)} &= x_1 x_4 x_7 x_{10} \ldots \\ x^{(2)} &= x_2 x_5 x_8 x_{11} \ldots \end{aligned}$$

Each character of $x^{(0)}$ is encrypted by the same shift cipher (secretly by $3 \sim$ `d`).

Each character of $x^{(1)}$ is encrypted by the same shift cipher (secretly by $14 \sim$ `o`).

Each character of $x^{(0)}$ is encrypted by the same shift cipher (secretly by $6 \sim$ `g`).

Let E_t be encryption by a plain-old shift cipher with key t.

Preliminary to determining the **first**, **second**, and **third** shifts (secretly `dog`) in the key, the Friedman attack determines the **differences** of the shifts.

Compute $I(x^{(0)}, E_{-t}(x^{(1)}))$ for $t = 0, 1, 2, \ldots$.

The values of t which give ~ 0.064 are the most likely values for the *difference*

$$1^{\text{th}}\text{shift} - 0^{\text{th}}\text{shift}$$

Compute $I(x^{(0)}, E_{-t}(x^{(2)}))$ for $t = 0, 1, 2, \ldots$.

The values of t which give ~ 0.064 are the most likely values for the *difference*

$$2^{\text{th}}\text{shift} - 0^{\text{th}}\text{shift}$$