1 Codebreaking

Continued from previous section.

1.1 Index of Coincidence

Imagine starting with some text consisting of only uppercase alphabet characters. Separate all of the letters from each other. Throw them all into in a bag to shuffle them up. Then draw two letters from it without replacement. What is the probability that the two letters you drew are the same?

We can let N be the total number of letters in the bag. For each letter α, let N_α be the number of times that α appears in the bag. Then, $\frac{N_\alpha}{N}$ is the probability that α is the first letter you draw from the bag. After you get that letter, there are $N-1$ letters left in the bag and α occurs only $N_\alpha - 1$ times, so the probability of drawing α again is $\frac{N_\alpha-1}{N-1}$. The overall probability, then, of drawing α twice is

$$\frac{N_\alpha(N_\alpha-1)}{N(N-1)}.$$

Now, notice that this probability is for *some* letter α. However, the question we asked in the first paragraph asks what the probability is of drawing two of the same letters in a row. So, we need to consider every possible letter; therefore, the probability we're looking for is

$$\sum_{\alpha\in\{A,\dots,Z\}} \frac{N_\alpha(N_\alpha-1)}{N(N-1)}.$$

(Exercise.) Above we stated that the probability of drawing α for our first letter is N_α/N and the probability of drawing α for our second letter is $(N_\alpha-1)/(N-1)$, so the probability of drawing two α's in a row is the product of these. Justify this "multiplication" of probabilities using conditional probabilities.

Let A_α be the event that the first letter is α, and let B_α be the event that the second letter is α. Notice how $\mathbb{P}[A_\alpha] = \frac{N_\alpha}{N}$ and $\mathbb{P}[B_\alpha|A_\alpha] = \frac{N_\alpha-1}{N-1}$, so it follows that

$$\mathbb{P}[B_\alpha|A_\alpha] = \frac{\mathbb{P}[B_\alpha\cap A_\alpha]}{\mathbb{P}[A_\alpha]} \implies \mathbb{P}[B_\alpha|A_\alpha]\mathbb{P}[A_\alpha] = \mathbb{P}[B_\alpha\cap A_\alpha] \implies \mathbb{P}[B_\alpha\cap A_\alpha] = \frac{N_\alpha}{N}\frac{N_\alpha-1}{N-1}.$$

Since there are 26 letters in the English alphabet, it turns out to be convenient to normalize this probability we computed above by multiplying by 26. The resulting number gets a special name:

Definition 1.1: Index of Coincidence

Let N be the length of some text and, for each letter α, let N_α be the number of times α occurs in this text. The **index of coincidence** of the text, denoted IC, is the number

$$IC = 26\sum_{\alpha\in\{A,\dots,Z\}} \frac{N_\alpha(N_\alpha-1)}{N(N-1)}.$$

Suppose that you start with a text where every letter appears equally often. What would the index of coincidence of such a text be?

If every letter appears equally often, then we know that there will be $\frac{1}{26}N$ of each letter. Then, the index of coincidence is given by

$$\begin{aligned}
IC &= 26 \sum_{\alpha \in \{A,\dots,Z\}} \frac{\frac{1}{26}N(\frac{1}{26}N-1)}{N(N-1)} \\
&= 26 \frac{\frac{1}{26}N(\frac{1}{26}N-1)}{N(N-1)} \sum_{\alpha \in \{A,\dots,Z\}} 1 \\
&= 26 \frac{\frac{1}{26}N(\frac{1}{26}N-1)}{N(N-1)} 26 \\
&= \frac{N(\frac{1}{26}N-1)}{N(N-1)} 26 \\
&= \frac{(\frac{1}{26}N-1)}{(N-1)} 26 \\
&= \frac{N-26}{N-1}
\end{aligned}$$

Notice how, as $N \mapsto \infty$, $\frac{N-26}{N-1} \approx 1$, so it follows that the index of coincidence would be approximately 1.0.

(Exercise.) What is the largest possible index of coincidence? What kind of text would result in an index of coincidence that's as large as possible?

Suppose only one letter was in some given text. Then, $N_\alpha = N$ and

$$\begin{aligned}
IC &= 26 \sum_{\alpha \in \{A,\dots,Z\}} \frac{N(N-1)}{N(N-1)} \\
&= 26 \sum_{\alpha \in \{A,\dots,Z\}} 1 \\
&= 26 \cdot 26 \\
&= 26^2.
\end{aligned}$$

So, the largest possible index of coincidence is 26 and the resulting text would only have one letter.

(Exercise.) Let N be the length of some text and, for each letter α, let N_α be the number of times α occurs in this text. Let $p_\alpha = N_\alpha / N$. Show that, if N is very large, then

$$IC \approx 26 \sum_{\alpha \in \{A,\dots,Z\}} p_\alpha^2.$$

Note that

$$\begin{aligned} IC &= 26 \sum_{\alpha \in \{A,\ldots,Z\}} \frac{N_\alpha(N_\alpha - 1)}{N(N-1)} \\ &\approx 26 \sum_{\alpha \in \{A,\ldots,Z\}} p_\alpha p_\alpha \qquad \text{Assume } N \text{ is very large} \\ &= 26 \sum_{\alpha \in \{A,\ldots,Z\}} p_\alpha^2. \end{aligned}$$

As we've seen, relative frequencies of letters in English are roughly stable from text to text, at least for sufficiently long texts. Correspondingly, the index of coincidence is also roughly stable from text to text! Here is the relevant heuristic:

(Heuristic.) The index of coincidence of long English plaintext is

- typically around 1.75, and
- almost always between 1.5 and 2.

Note that, based on the exercise above,

$$IC \approx 26 \sum_{\alpha \in \{A,\ldots,Z\}} p_\alpha^2.$$

The right-hand side always exactly the same if we perform a simple substitution on our text.

(Example.) For example, suppose the letter `E` has a relative frequency of 13% in some plaintext. Then, $p_E^2 \approx 0.13^2$ will appear as a summand in the above expression for the IC of this plaintext.

Now, suppose we perform a simple substitution that converts `E` to `Q`. Then, $p_Q^2 \approx 0.13^2$ will appear as a summand of the IC for the ciphertext.

The subscript is different, but the value is exactly the same. So, since we're summing over all the letters, the value of IC does not change when we performa simple substitution.

(Exercise.) Suppose some English plaintext is encrypted using a transposition cipher (e.g., rectangular transposition). How will the index of coincidence of the plaintext compare to that of the ciphertext?

Because rectangular transposition (and any cipher that just rearranges text around) simply rearranges the letters of the plaintext around, the index of coincidence for the plaintext will be exactly the same as the index of coincidence for the ciphertext.

In particular, this means that the index of coincidence of the ciphertext will probably have the same properties as the described heuristic above.

1.2 Breaking the Vigenere Cipher

With the index of coincidence in hand, we can now break the Vigenere cipher. Although this isn't the first method that was used to break said cipher, it does use the index of coincidence.

Suppose we start with ciphertext which is known for being encrypted using the Vigenere cipher:

```
IKGWERTZONXEULTEZWHWXTYHSZKEGQOJNENRUDJGFRNKUISLDZOMKDKWZHVU
OSJLFZEJJONQHWVRFEATRYIDIKKDKKEHNAEWOEYFIRMLNIENIFMAJKELXAMH
RKKDKKEEUOIVADUNVRNLNEERNKKNJHHWNAUKESXDYLSHGRVQTKGNUFOEVAEL
OFYRVSESZYVWSLOLCLDGTTCLKWHEZQGGATZQTZKDRUKFUWRQDAJOEWLAQESH
IFMLVITTEMPVEDLIEWHAYGILMZUUJHIUGNERTZKLGLTAYHROLTKGCDDONEEW
```

```
    HWEL...
```

Then, we can try to decrypt the ciphertext like so:

1. First, make a guess for the "period" (i.e., the length of the keyword). Suppose we guess a period of 5. Then, we can break the ciphertext into rows of that length.

   ```
   IKGWE
   RTZON
   XEULT
   EZWHW
   XTYHS
   ZKEG...
   ```

   If the period we've guessed is correct, **each column of this rectangle was encryped using a Caesar cipher with the same shift**. This means that we should be able to detect that our guess for the period is correct by examining indices of coincidence. If every column has an index of coincidence in the range that's expected of English text, our guess for the period is probably correct. Otherwise, we should try a different period.

   In our example above, if we guess a period of 5, it turns out that the indices of coincidence of the five columns are 1.13, 1.11, 1.18, 1.15, and 1.15, respectively. That's outside the range that's expected of English text, so our guess of 5 is probably is wrong. If we try a period of 6, we find that the indices of coincidence are 1.75, 1.77, 1.73, 1.72, 1.79, and 1.74, respectively – all in the right range. So, the period of 6 is probably correct.

2. Once we find a probable guess for the period, we're in the clear. Then, we just have to figure out what the shift is for each column, which we can do using basic frequency analysis. Recall that the most frequent letter in each column should correspond to `E`, so we should guess that the shift for that column is the shift that moves `E` to the most frequent letter (If that doesn't work we might try shifting so that the second most frequent letter in that column moves to `E` instead.)