Cracking the enigma

Recall that deciphering a message encrypted by the enigma required:

- Knowledge of the plugboard configuration
- Drum setup (rotors, initial positions, notches)
- Reflector plate knowledge.

However, if we accept that permutation ciphers are insecure (something we will soon justify) then recalling that the action of the enigma is:

$$P^{-1}R_1^{-1}R_2^{-1}R_3^{-1}R\,R_3R_2R_1P$$

we observe:

If we can figure out the rotor configuration, what's leftover is simply a permutation cipher. (Not exactly, but the point is that what remains is a few unknown <u>constant</u> permutations) KEY.

So we focus on cracking the initial rotor configuration.

Early war and Polish codebreaking: (Rejewski)

The enigma machine was used as follows:
Cipher clerks, on a monthly basis, would be given a table of daily keys (not really daily, but changed at various intervals). These were instructions on how to set up the enigma, depending on the time.

- Begin with a message key. The message key was ~~#~~ a repeated 3-letter string (sent twice using the daily key settings) that gave instructions to the recipient on how to configure their machine to read the rest of the message.

  Idea: A 3-letter chunk repeated twice is too little data for you to decode the daily key. Anything after the daily key is useless ~~in~~ helping to crack the enigma, it could be encrypted with any settings at all.

This was a huge error.

Idea: The first six letters sent each day were encrypted with six different permutations. Let's name them:

$$A = P^{-1}R_1^{-1}R_2^{-1}R_3^{-1}R\,R_3\,R_2\,R_1\,P$$

$$B = P^{-1}S^{-1}R_1 S\,R_2\,R_3^{-1}R\,R_3\,R\,S_2^{-1}R_1 S P$$

$C = \dots$

$D = \dots$

$E = \dots$

$F = \dots$

} fill these in with general form, bear in mind we don't know when the rotors turn...

Suppose we intercept the following message key, sent using the daily key:

$$d\ m\ q\quad v\ b\ f$$

We know this is the same 3 letters sent twice. Say it's $x\ y\ z\ \ x\ y\ z$.

Then $A(x) = d$ and $D(x) = v$

But remember that all permutations produced by the Enigma square to give the identity! So $A^2 = id$, $B^2 = id$, etc. Therefore

$$A(x) = d \Rightarrow A(A(x)) = A(d)$$
$$\Rightarrow A^2(x) = A(d)$$
$$\Rightarrow x = A(d)$$

Substitute this into $D(x) = v$, and get

$$D(A(d)) = v.$$

So, if we intercept enough messages, we can know <u>exactly</u> what the permutation DA, EB, FC are. (In practice you needed to intercept about 60-80 messages).

<u>Q</u>: How does knowing DA, EB, FC let you figure out the rotor settings? ~~In particular, the plugboard permutation P can be anything at all~~

First, do some espionage: We analyzed the enigma as though there were 26! possible rotors, but in reality only a very small number were constructed (e.g. the navy used roughly 10 rotors at any given time). Still, the plugboard permutation P has

$25!! \sim 8 \times 10^{12}$ possible configurations... so?

Trick for "getting around" the plugboard: Cycle decomposition

A *cycle* is a special kind of permutation.

Def: A permutation $\sigma: S \to S$ is called a *cycle* if there are k elements $a_1, a_2, \dots, a_k \in S$ with

$$\left.\begin{aligned} \sigma(a_1) &= a_2 \\ \sigma(a_2) &= a_3 \\ \sigma(a_3) &= a_4 \\ &\vdots \\ \sigma(a_k) &= a_1 \end{aligned}\right\} \sigma \text{ "cycles through" these guys}$$

and $\sigma(s) = s$ for all other $s \in S$. Write: $\overset{\sigma}{\overset{\shortparallel}{(a_1\ a_2\ a_3 \cdots a_k)}}$.

Example: If

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 1 & 4 & 2 & 5 \end{pmatrix}$$

then σ is a cycle, because:

$$\left.\begin{aligned} \sigma(1) &= 3, \\ \sigma(3) &= 4 \\ \sigma(4) &= 2 \\ \sigma(2) &= 1 \end{aligned}\right\} \text{cycling through}$$

$\sigma(5) = 5$ ← stays fixed.

We write: $\sigma = (1\ 3\ 4\ 2)$.

Example: Suppose $\sigma = (1\ 3\ 5\ 2)$ and $\tau = (2\ 5\ 6)$.

Describe the "product" $\sigma \circ \tau : \{1, 2, 3, 4, 5, 6\} \longrightarrow \{1, 2, 3, 4, 5, 6\}$.

Solution:

$$\sigma \circ \tau (1) = \sigma(1) = 3$$
$$\sigma \circ \tau (2) = \sigma(5) = 2$$
$$\sigma \cdot \tau (3) = \sigma(3) = 5$$
$$\sigma \circ \tau (4) = \sigma(4) = 4$$
$$\sigma \circ \tau (5) = \sigma(6) = 6$$
$$\sigma \circ \tau (6) = \sigma(2) = 1.$$

So $\sigma \circ \tau = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 2 & 5 & 4 & 6 & 1 \end{pmatrix}$

which is actually $\sigma \circ \tau = (1\ 3\ 5\ 6)$.

We call two cycles $\sigma, \tau : S \longrightarrow S$ "disjoint" if $\sigma = (a_1, \dots a_k)$ and $\tau = (b_1 \cdots b_m)$ with $\{a_1, \dots, a_k\} \cap \{b_1, \dots, b_m\} = \emptyset$. (I.e. they "cycle through" two non-intersecting sets).

E.g. The cycles in the previous example are not disjoint, but

$$\sigma = (1\ 2\ 3) \text{ and } \tau = (5\ 4\ 6)$$

are disjoint.

Theorem: Every permutation $\sigma : \{1, \dots, n\} \longrightarrow \{1, \dots, n\}$ can be written as a composition of disjoint cycles.

Example: Consider

$$f = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 4 & 3 & 5 & 1 & 6 & 2 \end{pmatrix}.$$

Let's see what f does to 1:

$$f(1) = 4$$
$$f(4) = 1$$

aha! So the cycle $(1\ 4)$ is part of what f does to $\{1, 2, 3, 4, 5, 6\}$.

Now start at the smallest number which didn't appear above, i.e. at 2. Then

$$f(2) = 3$$
$$f(3) = 5$$
$$f(5) = 6$$
$$f(6) = 2$$

another cycle! So $(3\ 5\ 6\ 2)$ is part of what f does to $\{1, 2, 3, 4. 5, 6\}$.

This completely describes f, so we have

$$f = (1\ 4)(3\ 5\ 6\ 2)$$

Remark: Note that if σ and τ are disjoint cycles, then $\sigma \circ \tau$ and $\tau \circ \sigma$ are equal

Why?

$$\{\overbrace{a, b, c,}^{\sigma \text{ acts here}} \dots\dots , \overbrace{x, y, z}^{\tau \text{ acts here}}\}$$

Last : If we write a permutation f as a product of disjoint cycles:

$$f = \sigma_1 \sigma_2 \sigma_3 \sigma_4 \quad \text{(could be longer, let's do length 4)}$$

and
$$\sigma_1 = (a_1 \; a_2 \cdots a_j)$$
$$\sigma_2 = (b_1 \; b_2 \cdots b_k)$$
$$\sigma_3 = (c_1 \; c_2 \cdots c_l)$$
$$\sigma_4 = (d_1 \; d_2 \cdots d_m)$$

then the set of numbers $\{j, k, l, m\}$ is called the cycle type of f. I.e list the lengths of cycles you used to write down f.

Example : In our last example, we found

$$f = (1 \; 4)(3 \; 5 \; 6 \; 2)$$

so the cycle type of f is $\{2, 4\}$.

Final new idea before returning to the enigma:

Theorem : Suppose f and g are permutations. If there is a third permutation σ such that

$$\sigma f \sigma^{-1} = g$$

then f and g have the same cycle type.

Proof:

If $f = \tau_1 \tau_2 \tau_3 \cdots \tau_k$, then

$$\sigma f \sigma^{-1} = (\sigma \tau_1 \sigma^{-1})(\sigma \tau_2 \sigma^{-1}) \cdots (\sigma \tau_k \sigma^{-1})$$

and it turns out that each ~~cycle~~ $\sigma \tau_i \sigma^{-1}$ is a cycle of the same length as the original cycle τ_i.

Back to enigma:

We have figured out DA, EB, FC by eaves dropping on messages. So we know

$$DA = P^{-1} S^{-3} R_1^{-1} S^3 R_2^{-1} R_3^{-1} R R_3 R_2 S^{-3} R_1 S^3 \cancel{P P^{-1}} R_1^{-1} R_2^{-1} R_3^{-1} R R_3 R_2 R_1 P$$

i.e. we know that

$$DA = P^{-1} X_{DA} P$$

where P is the plugboard and X_{DA} is a permutation that depends only on the rotor position and reflectors and we assume the reflector is known.

By the previous theorem, this does tell us something about the rotor positioning, even if we don't know anything about the plugboard: We know the cycle type of X_{DA}; it's the same as DA.

Similarly, we know the cycle type of X_{EB} (same as EB) and X_{FC} (same as FC).

So here's the attack:

Make a huge table of initial rotor positions (about 105 000 entries) and for each initial position, write down the corresponding cycle types for DA, EB, FC that result.

Then, having determined the cycle types by eavesdropping, you look up in your table what rotor positions could've given this - it's often just a few, sometimes only one! So you're basically done, since what remains is no harder than solving a handful of permutation cyphers.

Late war & British efforts

The germans eventually realized this error, and stopped sending double message keys, added more rotors, etc. They also invaded Poland, so the Polish codebreaking effort fell apart. They handed their knowledge (and enigma replicas) to the British.

At Bletchly Park, with groundbreaking ideas of Alan Turing, they developed a new method based on "cribs": Likely phrases or repeated words that allow for a "brute force" analysis

E.g.: If you intercept a code from a weather station, you expect the word "wetter" to appear. If you succeed in identifying "wetter" in the message, say:

w e t t e r
e t j w p x } becomes

then you can automatically reject some rotor positionings, because they would never encrypt "wetter" to etjwpx.

By using multiple cribs you can cut down the number of rotor positions to a manageable amount, and "brute force" the rest.

§4.1

Breaking classical ciphers

Caesar cipher.

Obviously you can break the Caeser cipher by correctly guessing the encryption of a single letter.

I.e. If $f(x) = x + b \pmod{26}$ is the cipher, and "b" the key,

then if you guess $f(2) \equiv 12 \pmod{26}$

then you know $2 + b \equiv 12 \pmod{26}$

$\Rightarrow b \equiv 10 \pmod{26}$.

So $b = 10$. This can often be done in practice; e.g if you intercept

T QZFYO ESP MLR

Then we know $T \mapsto A$ or $T \mapsto I$ since "A" and "I" are the only one-letter english words.

So try $f(x) = x + b$ with

$f(0) = 19$

ie $a \longmapsto t$.

Then applying $f^{-1}(x) = x - 19 \equiv x + 7 \bmod 26$ to our ciphertext, we get:

A XGWFV

we stop. It must be that $T \longmapsto I$ upon decrypting.

Try $f(x) = x + b$ with $f(8) = 19$

ie $I \longmapsto t$

So $f(8) = 8 + b = 19 \mod 26$

$$\Rightarrow b = 11.$$

So $f^{-1}(x) = x - 11 \equiv x + 15 \mod 26.$

Try applying this to ciphertext, we get:

I FOUND THE BAG.

So it works.

Even in the absence of single-letter words, we can reduced the number of guesses we need to make from 25 (which is not so bad) to a handful by using frequency analysis.

E.g if we analyze texts in English, and make a table of how often each letter occurs:

Letter	freq of occurence
e	12.58%
t	9.09%
a	8.00%
o	7.59%
i	6.92%
n	6.90%
⋮	⋮
etc.	etc

So in the absence of any other hints, guessing that the most common letter in the ciphertext maps to "e" (or t, a, o, i, n) is a pretty safe bet — and this can crack the code.

§4.3 Cracking Affine ciphers

If you look back, we didn't really cover decryption of the affine cipher. Let's mention that now:

If $f_{a,b}(x) = ax+b \bmod 26$ where $\gcd(a,26)=1$ then $f_{a,b}: \{0,\dots,25\} \longrightarrow \{0,\dots,25\}$ is a 1-1 and onto function.

To decrypt, we need an inverse function g, and we'll go out on a limb and guess that g has the same form as $f_{a,b}$, that is

$$g = g_{c,d}(x) = cx+d \bmod 26 \text{ with } \gcd(c,26)=1.$$

Compute:

$$g_{c,d}(f_{a,b}(x)) = c(ax+b)+d$$

$$= acx + cb+d \bmod 26$$

This will give "x" if $cb+d \equiv 0 \bmod 26$

and $ac \equiv 1 \bmod 26$.

Choosing $c = a^{-1} \bmod 26$ guarantees $ac \equiv 1 \bmod 26$, and then choosing $d \equiv -bc \bmod 26$ guarantees

$$cb + d \equiv 0 \bmod 26.$$

ie. $d \equiv -b\bar{a}^{1} \bmod 26.$

In our $f_{a,b}$ notation,

$$f_{a,b}^{-1} = f_{\bar{a}^{1}, -b\bar{a}^{1}}.$$

As with our "improved Caesar cipher attack", we can use frequency analysis to break this cipher. (In reality, brute force is still possible since there are only 312 keys).

<u>Example</u>: Suppose that we intercept the message

MCCLL IMIPP ISKLN UHCGI ..... etc.

First, it's chunked into 5-letter blocks so no "word size" attack will work. If we count occurences of letters, in the total message (only a piece is shown above) the letters C, I, L occur most frequently.

We have: C – 9 times

I – 7 times

L – 7 times.

So it is reasonable to guess that "E" became "C" upon encryption.

So if the message was encrypted with

$$f_{a,b}(x) = ax+b \pmod{26}$$

Then our guess is that

$$f_{a,b}(\underset{\underset{E}{\uparrow}}{4}) = \underset{\underset{C}{\uparrow}}{2}$$

or ~~4a~~ $4a+b \equiv 2 \bmod 26$.

Next most common letter in english: t. We guess that t was encrypted to either I or L, let's guess "I". Then

$$f_{a,b}(19) = 8$$

or

~~8a+~~ $19a+b \equiv 8 \bmod 26$.

So we try to solve

$$4a+b \equiv 2 \bmod 26$$
$$19a+b \equiv 8 \bmod 26$$

subtract to eliminate b's:

$$+15a \equiv +6 \bmod 26$$

The inverse of 15 mod 26 is 7, so multiply both sides by 7:

$$7 \cdot 15a \equiv 7 \cdot 6 \bmod 26$$

$$a \equiv 16 \bmod 26.$$

But this makes no sense, because then a would be even — and $\gcd(a,26)=1$ for the affine cipher to work.

So we change our guess: E encrypts to C
T encrypts to L

Then we get
$$4a+b \equiv 2 \bmod 26 \text{ ——— } ①$$
$$19a+b \equiv 11 \bmod 26 \text{ ——— } ②$$

Subtract ① from ②, we get
$$15a \equiv 9 \bmod 26$$

Since $15^{-1}=7$, $a \equiv 7\cdot 9 \bmod 26$
$$\equiv 11 \bmod 26.$$

Then $4(11)+b \equiv 2 \bmod 26$
$$\Rightarrow b \equiv 2-44 \equiv 10 \bmod 26.$$

So our guess is that the message was encrypted with $f_{11,10}(x) = 11x+10 \bmod 26$.

Then $f^{-1}_{11,10}(x) = f_{11^{-1},-10\cdot 11^{-1}}$. Since $11^{-1} \bmod 26 = 19$,

$$f^{-1}_{11,10}(x) = f_{19,18}(x)$$

So we try our ciphertext in this function:

ciphertext:	M C C L L I M I P P I S K L N U H C G I ⋯
as numbers:	12 2 2 11 11 8 12 8 15 15 ⋯⋯
apply $f^{-1}(x)$:	12 4 4 19 19 14 12 14 17 17
as letters:	M E E T T O M O R R ⋯

(It's going to say meet tomorrow...)

Moral of story: While trying all 312 keys is certainly possible with modern computers we needed *only two attempts* when we blindly applied the rule "The most common letter in the ciphertext likely corresponds to the most common letter in english".

Remark: We also got lucky that our system of congruences had only one solution. For example, if we had arrived at

$$1 \equiv 8a+b \mod 26$$

$$3 \equiv 0a+b \mod 26$$

subtract: $24 \equiv 8a \mod 26$

But then 8 does *not* have an inverse mod 26, since $\gcd(8,26)=2$.

It turns out that this allows *two* solutions:

$a=3$ and $a=16$ both work.

So then we have to either try both, or in this case, simply observe that $a=16$ is *not* allowed since we assume $\gcd(a,26)=1$.

§4.5 Breaking substitution ciphers with frequency analysis.

As before, we can simply take the ciphertext and guess that the most common letter corresponds to "e" in the plaintext.

Of course, for this to work the ciphertext needs to be long enough for the frequency with which letters occur to roughly match the frequency of the unencrypted letters in the english language.

E.g. "ARTTE"
is not a long enough ciphertext for any sort of frequency analysis to make sense. In fact, it could be multiple words!

PIZZA
LATTE
TALLY
SWEET ... etc, etc, etc.

But what if it IS long enough?