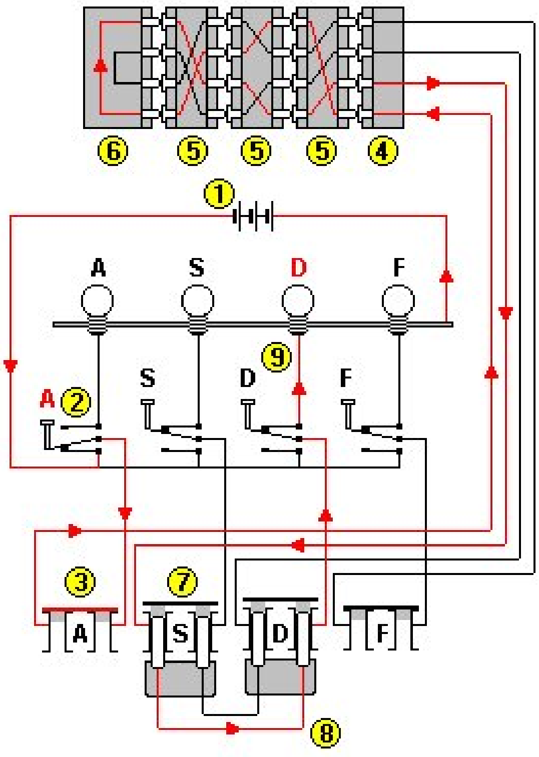
# Hillclimbing the Enigma Machine

To understand hill climbing, we need to look at how the Enigma machine works. There are two processes involved in encrypting letters from the keyboard through to the lamp output letters; a bank of three wired cipher wheels and the *Stecker* board. To decode the message we need to have the wheel bank set up exactly right. Consider the 3-wheel machine first, used by the army and air force. Chosen from a set of five wheels there are 60 possible arrangements for the three wheels inserted into the machine. Furthermore each wheel must be set to the same position as that used to encrypt the message. Since the wheels have 26 positions, there are 26x26x26 = 17567 positions to choose from. There is a further complication that the *Ringstellung* (ring position) of the right hand wheel needs to be set correctly; otherwise the wheels will not always step in the correct sequence. There is also a *Ringstellung* for the middle wheel, but this can mostly be ignored since it does not act very often – only once every 676 steps.
If the machine had no plugboard then decrypting the message would be easy. There are only:

26x26x26x26x60 = 27,418,560

different positions to try. A desktop PC would do this in a matter of minutes, perhaps 15 minutes.



The schematic on the left shows the arrangement of the wheels and plugboard of the Enigma. For clarity only four of the 26 circuits are shown.
A single cable is connected to the *Stecker* board between sockets D and S.
The key A [2] is pressed and a signal passes unchanged through the self *Stecker* A [3] and enters the wheel bank commutator [4] at pin A. The wheels [5] and reflector [6] substitute S for this letter and the signal arrives at the S pin on the commutator. From here it arrives at the S *Stecker* [7]. S is *Stecker*–paired to D; the signal is therefore swapped with D and leaves the D *Secker* to light the lamp D [9].

Enigma schematic © 2006 Dirk Rijmenants.

Now let us look at the *Stecker* board. This is where we need hill climbing. The task of the *Stecker* board is to increase the number of machine settings by swapping some of the letters as they pass through it. You will notice in the diagram above that some letters may be swapped up to twice in passing through the *Stecker* board (for example letters that enter and leave the wheel commutator at S and D). This means that if we have a wrong connection, we lose many letters when decoding. The *Stecker* board has a socket pair for each letter. A cable can be inserted into two letter positions and thereby swap these two letters. Normally for the *Wehrmacht* Enigma ten cables were used. This means that 20 letters undergo swapping and the other 6 pass through unchanged.

Here is a typical *Stecker*-pairing for ten cables; this was used on 17th August 1941 during Operation Barbarossa, the attack on Russia:

```
A/E C/F G/L H/I K/P M/S N/R O/U Q/Y T/W
```

Another way of looking at these connections is in the form of a look-up table:

```
EBFDACLIHJPGSRUKYNMWOVTXQZ
```

The number of different ways we can insert 10 cables into the 26 socket panel is:

150,738,274,900,000

Here is the problem. This number is over 5 million times larger than the number of wheel settings. So if we try all these *Stecker* possibilities, at all wheel settings, the computing time will be:

15 minutes x 150,738,274,900,000

That comes out at a few thousand million of years! – for one message! A distributed attack using an army of PCs is even useless.

## The M4 Naval Enigma

In the case of the 4-wheel Naval M4 machine, there is an added complications of a set of eight, not five, removable wheels from which to choose the three stepping wheels (336 possible arrangements rather than 60) and an additional fourth wheel with its reflector, of which there are two fourth wheels and two reflectors to choose from. The possible wheel settings thus become:

4x336x26x26x26x26 = 15,968,569,344

This is almost 600 times greater than the wheel settings for the 3-wheel machine......We need to look at hillclimbing.

## Wheels and Steckers

There is one big difference between the machine's wheel settings and the *Stecker* connections. If we get the wheels wrong, even slightly so, then we cannot decrypt a single letter correctly. However, the *Stecker* board will pass some letters through correctly at the correct wheel settings even if some of the plug connections are incorrect. We will get fragments of correct decrypt. What hill climbing attempts to do is to guess one connection, then examine the output to see if we can detect any increase in correct letters. If we succeed we keep that connection and try another and so on. We get a little help at the start since we know that six of the letters are already correct – but we do not know which six. Initially with no plugs connected, on average, six out of 26 letters will pass through the plugboard correctly. We have already mentioned the problem due to the letters passing through twice, so the number of correct letters will be:

6/26 x 5/25

This is about one in every twenty-one letters.

Here is a real message decrypted at the correct wheel setting with none of the ten cables connected:

```
EJWNZ YHJWH MAZHJ JNDSK ESQMJ IRJFO HJMNZ UQLVY AHRSN OTRCX
```

And here is the decrypt when all ten cables are connected. Can you spot any of this plaintext in the above?

```
        FUERD IVXKD RXEIN EVORD EREFE DERFU ERKFZ XKFZX EINSF UENFX
```

## The Index of Coincidence

So when we make a trial cable connection, how do we tell if it is correct or not, since it looks almost impossible to see any improvement? The Enigma message should consist of random letters, that is if we have a message 200 letters long we would expect each of the 26 letters of the alphabet to appear, on average, about 7 or 8 times. We encounter a problem here since a typical Enigma message, of maximum length 250 letters, is a very short sample with which to measure the distribution of 26 items. For example here is the letter distribution of a cipher message of 202 letters:

| K | F | X | L | I | A | T | Q | M | J | U | P | G | E | S | R | O | D | B | Z | W | V | H | Y | C | N |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 15 | 11 | 10 | 10 | 10 | 10 | 9 | 9 | 9 | 9 | 8 | 8 | 8 | 8 | 7 | 7 | 7 | 7 | 7 | 6 | 6 | 6 | 6 | 4 | 3 | 2 |

(Note. The property of the Enigma to never encipher a letter to itself has the effect of reducing the frequency of some letters, especially E in the cipher message, since E occurs about 13% in the plain text of Enigma messages. But often the message is too short to see this effect)

In simple terms, the Index of Coincidence (IC) is a measure of the ‘roughness’ of the distribution. A more mathematical explanation can be found here:
http://en.wikipedia.org/wiki/Index_of_coincidence

In practice it requires the sum of all letter frequencies in the decrypt to be multiplied together thus:

$$IC = \sum (fn)(fn-1)$$

(To be correct, the IC score is actually this product divided by (N)(N-1), where N is the message length, but this not necessary for the hill climb and it saves a bit of processing time)

Plain text has an uneven distribution, for example in Enigma messages E is 13%, N 8.4%, X 6.9%, R 6.8%, S 6.2%....and J is way down at 0.4%. If we make a cable connection that is wrong and gives a number of incorrect letters, we would expect these letters to be random. However, if the connection is correct, then some, but probably not all of letters in this circuit may be correct and the letter distribution will move towards that of plain text. IC is a measure of this distribution. Random text, that is where all letters are present with nearly the same frequency, will have an IC ‘score’ of 1/26 or 0.03846. If we measure the IC score of plain Enigma text it can be up to around 0.05 to 0.07, depending on the actual frequencies of the letters in the plain message.

## Hillclimbing details

Now we come to the hill climb details. With no cables connected to the plugboard, each letter passes through unchanged. We can represent this with the look-up table:

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ.
```

We now systematically connect each cable, starting with the first and second position letters (here A-B), then the firsts and third ( A-C) and so on up to A-Z. With no cables connected, the first few groups of the decode looks like this with an IC score of 0.0385

```
EJWNZ YHJWH MAZHJ JNDSK ESQMJ IRJFO HJMNZ UQLVY AHRSN OTRCX
```

Connecting the cable A-B, we now have a new *Stecker* look-up table:

```
BACDEFGHIJKLMNOPQRSTUVWXYZ
```

and the decode, with a score of 0.0385, using these connections is:

```
EJWNZ YHJWH MBZHJ JNMSK LRQWJ IRJFO HJMNZ UQLVY BHRSN OTRCX
```

We remove this connection since there is no improvement in the score and move on to the next connection A-C giving a score of 0.0380

```
CBADEFGHIJKLMNOPQRSTUVWXYZ
```

```
EJWNZ YUJWH MCZHZ JNNSK OQQEJ IRJFO FRMNZ UQLVY CHRSN OTRAX
```

Again there is no improvement.

Eventually we find an increase in the score for the connection A-E and this is the highest score we can obtain, 0.0407, for all connections to the first letter.

```
EBCDAFGHIJKLMNOPQRSTUVWXYZ
```

```
AJWNZ YHJWH MEZHJ JNUSK ENQCJ IRJFO HJMNZ UQCVY  EHRSN OTRCX
```

The decrypted letters that are correctly decoded are highlighted here in red. This is just for illustration purpose; we don’t know at this stage which letters are correct.

Moving on to the second letter position, we make the new connections B-C in our look-up table

```
ECBDAFGHIJKLMNOPQRSTUVWXYZ
```

We then continue with B-D, B-A, B-F and so on in turn, testing for an improvement in the score at each connection. But note that if we reach the connection BA, there is a slight complication in that A is already connected to E. We therefore need to test the two possible arrangements of B-A with E-E and B-E with A-A and choose any arrangement which gives an increase in the score.
Each letter is thereby swapped with all following letters until we reach the end of the alphabet. At the end of this process, which will be after around 600 trial connections, we hopefully have several cable connections that are correct. A decode now shows a few more correct letters and the IC score is up to 0.0502

```
AOWND IVXKI NEZIR ENUNP ENEFE HRJCO ENKNZ UQFZX EIRSN OERFX
```

## Trigram scores

We can now see fragments of words appearing in the decrypt. We next attempt to find further correct connections by using a score of frequency of trigrams that appear in the decode, in place of the IC score. These frequencies are obtained from known plain text. For example, some frequencies of common trigrams from a sample of 20,000 letters of Enigma telegrams are:

| EIN | INS | FUE | ZWO | ULL | IER | NUL | UNG | ENF | VIE | XEI | REI | UEN | DER |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 194 | 137 | 108 | 90 | 90 | 89 | 87 | 83 | 80 | 80 | 78 | 76 | 71 | 67 |

Wherever a trigram appears in the decode we sum its score to arrive at a total score for the whole decrypt. For example XEI highlighted in red in group 8 and 9 would contribute 78 to the total score.

After the first 600 or so trial decrypts, the look up table is:

EBFDACLIHJPGMNOKQRSTUVWXYZ

## Hillclimb second pass

We now repeat the entire hill climb process starting with the first letter again, but now base the score on the sum of the trigrams scores found in the decrypt, rather than IC. Trigrams are a much more powerful detector of plain text but could not be used for the first pass since the likelihood of connected word fragment appearing in that stage is extremely low.

We first try connections E-B, E-F, E-D and so on as before, hopefully we get a correct or near correct decrypt at the end of the second pass:

```
FUERD IVXKD RXEIN EVORD EREFE DERFU ERKFZ XKFZX EINSF UENFX
```

Easy messages may well break with the above procedure. However, in most situations it is necessary to run over with several passes using IC scoring followed by bigram scores (based on the frequencies of pairs of letters) and finally trigram scores, and also use more cable swapping tests. A successful hillclimb may therefore be possible with only a few thousand trial decrypts, a considerably smaller number than the 150,738,274,900,000 plugboard settings.
**But please note that this hillclimb needs to be repeated for each of the 27,418,560 wheel setting since we don't know which wheel setting is correct!**
There is no guarantee with hill climbing, it may fail and in the Enigma situation it depends a lot on the letter distribution and word content of the message, the *Stecker*-pairs and the accuracy of the bigram and trigram tables.

## Distributed computing - The Naval M4 Enigma

Breaking 3-wheel Enigma messages on a single PC is feasible. The processing time is around 10 – 60 hours depending on the message length and the PC speed. However, for the M4 Naval Enigma, the number of possible wheel settings is several hundred times greater, therefore unless a super-computer is available a distributed computing attack is very attractive. A few hundred or thousand machines running a background task could break into M4 within the same timescale as the single PC attacking a 3-wheel message. An added complication of the naval machine is that three of the eight wheels have double stepping notches. It makes it important therefore to consider the *Ringstellung* of the middle wheel if a double notch wheel is in the middle or fast position.