Automated Ciphertext-Only Cryptanalysis of the Bifid Cipher

António Machiavelo   Rogério Reis

Technical Report Series: DCC-2006-1

Departamento de Ciência de Computadores – Faculdade de Ciências

&

Laboratório de Inteligência Artificial e Ciência de Computadores

Universidade do Porto

Rua do Campo Alegre, 823 4150 Porto, Portugal

Tel: +351+2+6078830 – Fax: +351+2+6003654

`http://www.ncc.up.pt/fcup/DCC/Pubs/treports.html`

Automated Ciphertext-Only Cryptanalysis of the Bifid Cipher

António Machiavelo
ajmachia@fc.up.pt
Centro de Matemática da Universidade do Porto

Rogério Reis
rvr@ncc.up.pt
DCC&LIACC Universidade do Porto

February, 2006

Abstract

In this paper we describe a fully automated ciphertext-only cryptanalysis attack on the Bifid cipher, for which the original text language is known. We have implemented this attack using Python. We use an easily computable statistical function to find the period of the cipher, and then the key-table is generated in a fairly efficient way. The process is directed in such a way that strongly narrows the search space of possible solutions. This results in a feasible attack to a Bifid cryptogram, provided that its length is enough for accurate statistical analysis.

1 Introduction

The Bifid cipher [Ame05, Kah67] was invented by Félix-Marie Delastelle (1840-1902) and although was never used in any "serious" application, it became one of the most popular ciphers among "amateur" cryptologists.

The key consists of a square table, henceforth called key-table, composed by the characters of the alphabet, normally a 5×5 square with characters `i` and `j` identified, also called a Polyabus key, and a small integer ℓ, the block size or period, normally greater than 6. Take for instance $\ell = 7$ and the following key-table

	0	1	2	3	4
0	u	d	v	g	r
1	q	t	p	z	a
2	h	b	w	f	e
3	x	c	i	k	o
4	n	l	s	y	m

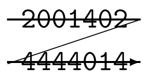
The text is divided in blocks of size ℓ, padded if necessary with some *nulls* at the end, and the coordinates of each letter are then written underneath it. Taking, for example, the text

"Err and err and err again but less and less and less" ,

one would obtain

```
errande rrander ragainb utlessa ndlessa ndlessx
2001402 0014020 0101342 0142441 4142441 4042443
4444014 4440144 4434201 0114224 0114224 0114220.
```

The ciphertext is now obtained recoding each block, using the same table, by reading pairs of coordinates horizontally, from left to the right as the following scheme suggests:



In our example the resulting cryptogram would be:

```
hdnemna uavrmdm ddoeysd dsmqtse lsmqtse nsmxtlh.
```

The parity of the period affects some aspects of the cipher, and in fact most authors [Ame05, Gai39] tend to recognize in the ciphers with an odd period an additional cryptographic strength, although some others present arguments against that [Bow60].

In general, let Σ be the alphabet used, with $|\Sigma| = n^2$, for some $n \in \mathbb{N}$. Let the key of a specific Bifid cipher be:

	0	1	$\cdots$	$n-1$
0	$\sigma_{0,0}$	$\sigma_{0,1}$	$\cdots$	$\sigma_{0,n-1}$
1	$\sigma_{1,0}$	$\sigma_{1,1}$	$\cdots$	$\sigma_{1,n-1}$
$\vdots$	$\vdots$	$\vdots$	$\cdots$	$\vdots$
$n-1$	$\sigma_{n-1,0}$	$\sigma_{n-1,1}$	$\cdots$	$\sigma_{n-1,n-1}$

When the block size is odd, encryption is done according to the following scheme,

$$\begin{array}{c|c|c|c|c|c|c}
\sigma_0 & \sigma_1 & \sigma_2 & \cdots & \sigma_{\ell-3} & \sigma_{\ell-2} & \sigma_{\ell-1} \\ \hline
x_0 & x_1 & x_2 & \cdots & x_{\ell-3} & x_{\ell-2} & x_{\ell-1} \\
y_0 & y_1 & y_2 & \cdots & y_{\ell-3} & y_{\ell-2} & y_{\ell-1}
\end{array}
\longmapsto
\begin{array}{c|c|c|c||c||c|c|c}
\tau_0 & \tau_1 & \cdots & \tau_{\frac{\ell-3}{2}} & \tau_{\frac{\ell-1}{2}} & \tau_{\frac{\ell+1}{2}} & \cdots & \tau_{\ell-1} \\ \hline
x_0 & x_2 & \cdots & x_{\ell-3} & x_{\ell-1} & y_1 & \cdots & y_{\ell-2} \\
x_1 & x_3 & \cdots & x_{\ell-2} & y_0 & y_2 & \cdots & y_{\ell-1}
\end{array}$$

while for a even block size,

$$\begin{array}{c|c|c|c|c|c|c}
\sigma_0 & \sigma_1 & \sigma_2 & \cdots & \sigma_{\ell-3} & \sigma_{\ell-2} & \sigma_{\ell-1} \\ \hline
x_0 & x_1 & x_2 & \cdots & x_{\ell-3} & x_{\ell-2} & x_{\ell-1} \\
y_0 & y_1 & y_2 & \cdots & y_{\ell-3} & y_{\ell-2} & y_{\ell-1}
\end{array}
\longmapsto
\begin{array}{c|c|c|c||c|c|c|c}
\tau_0 & \tau_1 & \cdots & \tau_{\frac{\ell-2}{2}} & \tau_{\frac{\ell}{2}} & \tau_{\frac{\ell+2}{2}} & \cdots & \tau_{\ell-1} \\ \hline
x_0 & x_2 & \cdots & x_{\ell-2} & y_0 & y_2 & \cdots & y_{\ell-2} \\
x_1 & x_3 & \cdots & x_{\ell-1} & y_1 & y_3 & \cdots & y_{\ell-1}
\end{array}$$

For a fixed period, the number of different Bifid ciphers is actually smaller than the number of distinct key-tables, because if one applies the very same permutation two both rows and columns, the resulting key-table will yield the same cipher. That this comprehends all possible Bifid ciphers, follows from the fact that in order to get the same cipher using a table in which two rows were swapped, an additional swap of columns with identical indexes must also be performed, because all the diagonal elements must be preserved. This is made obvious by the observation that for a block made by repeating a single character, its image in the cryptogram will result in itself if and only if that character is in the diagonal of the key-table. This shows that, although the number of different tables is $n^2!$, the number of corresponding Bifid ciphers is only $\frac{n^2!}{n!}$.

2 How to get the period

In this section we will see that the block size, ℓ, can easily be determined by computing the frequency of pairs of equal letters at a given distance d, *i.e.* the number of occurrences of the pattern $\alpha\Sigma^{d-1}\alpha$, for all α, and graphing the results as a function of d. The shape obtained will approximately be a sinusoid, with the "right" period. That this is so, will now be shown in detail for the case in which the period is odd. A similar argument applies to the even case.

2.1 Distribution of frequencies of homogeneous non-connected digraphs

For $\alpha \in \Sigma$, let $row(\alpha)$ and $col(\alpha)$ denote the row and column, respectively, of the key-table to which α belongs to. Let $p(\alpha)$ represent the probability of an occurrence of α in the text, and $p(\alpha\beta)$ the corresponding probability for the digraph $\alpha\beta$. By α^t, the transposed of α, we denote the key-table entry that satisfies: $row(\alpha^t) = col(\alpha)$ and $col(\alpha^t) = row(\alpha)$. The following quantities play an important role in what follows:

$$
\begin{aligned}
\rho_\alpha &= \sum_{\beta \in row(\alpha)} p(\beta), && (1)\\
\kappa_\alpha &= \sum_{\beta \in col(\alpha)} p(\beta), && (2)\\
B_\alpha &= \sum_{\substack{\beta \in row(\alpha)\\ \gamma \in row(\alpha^t)}} p(\beta\gamma), && (3)\\
C_\alpha &= \rho_\alpha\, \kappa_\alpha, && (4)\\
A_\alpha &= \sum_{\substack{\beta \in col(\alpha)\\ \gamma \in col(\alpha^t)}} p(\beta\gamma). && (5)
\end{aligned}
$$

We start by determining the probability, $\mathcal{P}(\tau_i = \alpha)$, that the i-th letter of the ciphertext be equal to the particular letter $\alpha \in \Sigma$. This probability depends on whether the letter τ_i occurs in the section before the central position of the block, in that central position, or in the section after it. By the way, the symbols B, C and A given above stand for "before", "central" and "after". Let us consider the three cases, as displayed in the following figure:

$$
\begin{array}{c|c|c|c|c|c|c}
\multicolumn{1}{c}{} & \multicolumn{1}{c}{\overbrace{\qquad\qquad}^{B}} & \multicolumn{1}{c}{} & \multicolumn{1}{c}{\overbrace{\qquad}^{C}} & \multicolumn{1}{c}{} & \multicolumn{1}{c}{\overbrace{\qquad\qquad}^{A}} & \multicolumn{1}{c}{} \\
\cdots & \tau_i & \cdots & \tau_{\frac{\ell-1}{2}} & \cdots & \tau_{\frac{\ell+1}{2}+j} & \cdots \\
\hline
\cdots & x_{2i} & \cdots & x_{\ell-1} & \cdots & y_{2j-1} & \cdots \\
\cdots & x_{2i+1} & \cdots & y_0 & \cdots & y_{2j} & \cdots
\end{array}
\qquad \left(0 \le i,j \le \tfrac{\ell-3}{2}\right)
$$

In the first case:

$$\mathcal{P}(\tau_i = \alpha) = \mathcal{P}(\sigma_{2i-1} \in row(\alpha) \wedge \sigma_{2i} \in row(\alpha^t)) = B_\alpha. \qquad (6)$$

In the second case:

$$\mathcal{P}(\tau_{\frac{\ell-1}{2}} = \alpha) = \mathcal{P}(\sigma_\ell \in row(\alpha) \wedge \sigma_1 \in col(\alpha)) \simeq \rho_\alpha\, \kappa_\alpha = C_\alpha. \qquad (7)$$

Finally, in the last case:

$$\mathcal{P}(\tau_{\frac{\ell+1}{2}+j} = \alpha) = \mathcal{P}(\sigma_{2j} \in col(\alpha^t) \wedge \sigma_{2j+1} \in col(\alpha)) = A_\alpha. \tag{8}$$

The probability, $\mathcal{P}_d$, that two letters at distance d are the same can now be expressed in terms of the quantities $B_\alpha, C_\alpha, A_\alpha$ as:

$$\mathcal{P}_d = \sum_\alpha (B_\alpha^2 + A_\alpha^2) \cdot \mathcal{P}_d^{(HH)} + \sum_\alpha C_\alpha^2 \cdot \mathcal{P}_d^{(MM)} + \sum_\alpha C_\alpha (B_\alpha + A_\alpha) \cdot \mathcal{P}_d^{(HM)} + \sum_\alpha B_\alpha A_\alpha \cdot \mathcal{P}_d^{(HH')},$$

where:

$$\begin{aligned}
\mathcal{P}_d^{(HH)} &= \mathcal{P}(\text{“letters are both in the same “half” } (B \text{ or } A) \text{ of the blocks they belong to”}),\\
\mathcal{P}_d^{(MM)} &= \mathcal{P}(\text{“both letters are in middle position of their block”}),\\
\mathcal{P}_d^{(HM)} &= \mathcal{P}(\text{“one letter is in the middle position of its block while the other is not”}),\\
\mathcal{P}_d^{(HH')} &= \mathcal{P}(\text{“one of the letters is in the } B \text{ half and the other in the } A \text{ half”}).
\end{aligned}$$

It is not too hard to see that these probabilities depend only on d modulo ℓ, and that one has, for $1 \le d \le \ell$:

$$\begin{aligned}
\mathcal{P}_d^{(HH)} &= \left|1 - \frac{2d}{\ell}\right| - \frac{1}{\ell}\\
\mathcal{P}_d^{(MM)} &= \begin{cases} 0 & ,d < \ell \\ \frac{1}{\ell} & ,d = \ell \end{cases}\\
\mathcal{P}_d^{(HM)} &= \begin{cases} \frac{2}{\ell} & ,d < \ell \\ 0 & ,d = \ell \end{cases}\\
\mathcal{P}_d^{(HH')} &= \begin{cases} 1 - \frac{1}{\ell} - \left|1 - \frac{2d}{\ell}\right| & ,d < \ell \\ 0 & ,d = \ell. \end{cases}
\end{aligned}$$

For example, the first equality may be established as follows. For an homogeneous non-connected digraph of distance d, let j be the position of its leftmost character in the ciphertext. We have to consider two separate cases: $d \le \frac{\ell-1}{2}$ and $d > \frac{\ell-1}{2}$. In the first case, illustrated on the top of Figure 1, for the digraph to be entirely contained in a B or A “half” of a block, one must have

$$\left(0 \le j < \frac{\ell-1}{2} \;\wedge\; j + d < \frac{\ell-1}{2}\right) \;\vee\; \left(\frac{\ell-1}{2} < j < \ell \;\wedge\; j + d < \ell\right).$$

The number of j that satisfies the above conditions is

$$\ell - 2d - 1.$$

In the second case, illustrated on the bottom of Figure 1, one is lead to the conditions:

$$\left(0 \le j < \frac{\ell-1}{2} \;\wedge\; l \le j + d < \ell + \frac{\ell-1}{2}\right) \;\vee\; \left(\frac{\ell-1}{2} < j < \ell \;\wedge\; \ell + \frac{\ell-1}{2} < d + j < 2\ell\right),$$

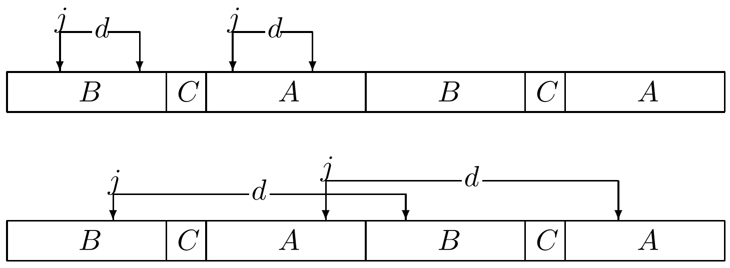


Figure 1: Illustration for (HH) case

and the number of j in this case is

$$-\ell + 2d - 1.$$

Hence the result above for $\mathcal{P}_d^{(HH)}$.

Considering,

$$\begin{aligned} r &= \sum_\alpha (B_\alpha^2 + A_\alpha^2) \\ s &= \sum_\alpha C_\alpha^2 \\ u &= \sum_\alpha C_\alpha (B_\alpha + A_\alpha) \\ v &= \sum_\alpha B_\alpha A_\alpha, \end{aligned}$$

we get from the above that,

$$\mathcal{P}_d = \begin{cases} r + \frac{1}{\ell}(2u - v - r) - \frac{2d}{\ell}(r - v) & , d \le \frac{\ell-1}{2} \\ 2v - r + \frac{1}{\ell}(2u - v - r) + \frac{2d}{\ell}(r - v) & , \frac{\ell-1}{2} < d < \ell \\ r + \frac{1}{\ell}(s - r) & , d = \ell. \end{cases} \tag{9}$$

Notice that $r - v = \sum_\alpha \left(B_\alpha^2 + A_\alpha^2 - B_\alpha A_\alpha\right) = \sum_\alpha \left((B_\alpha - \frac{1}{2}A_\alpha)^2 + \frac{3}{4}A_\alpha^2\right) \ge 0$, and that $\mathcal{P}_i = \mathcal{P}_{\ell-i}$ for $i=1,2,\cdots,\frac{\ell-1}{2}$. So one concludes that:

$$\mathcal{P}_1 \ge \mathcal{P}_2 \ge \cdots \ge \mathcal{P}_{\frac{\ell-1}{2}} = \mathcal{P}_{\frac{\ell+1}{2}} \le \cdots \le \mathcal{P}_{\ell-2} \le \mathcal{P}_{\ell-1} = \mathcal{P}_1. \tag{10}$$

On the other hand,

$$\begin{aligned} \ell(\mathcal{P}_\ell - \mathcal{P}_1) &= (\cancel{\ell r} + (s - \cancel{r})) - (\cancel{\ell r} + (2u - v - \cancel{r}) - 2(r - v)) \\ &= 2r + s - 2u - v \\ &= \sum_\alpha \left(2B_\alpha^2 + 2A_\alpha^2 + C_\alpha^2 - 2C_\alpha B_\alpha - 2C_\alpha A_\alpha - B_\alpha A_\alpha\right). \end{aligned}$$

The quadratic form $2x^2 + 2y^2 + z^2 - 2zx - 2zy - yx$ is not positive definite and thus can take on negative values. In fact, for some cryptograms we have observed a negative sum for the right hand side of the last equality. However looking at $\mathcal{P}_\ell - \mathcal{P}_2$ one obtains a sum of quadratic expressions that are positive definite. Therefore $\mathcal{P}_\ell > \mathcal{P}_2$.

All this proves the following:

Theorem 1 *The function*

$$f : d \longmapsto \mathcal{P}_d$$

evaluated over a cryptogram obtained from a Bifid cipher of period ℓ is approximately a periodic function with the same period.

This is well illustrated by the example in Figure 2.

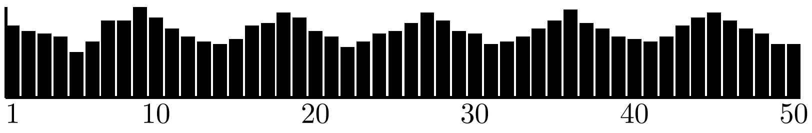


Figure 2: Analysis for a cypher of period 9.

2.2 Distribution of the standard deviation for non-connected digraphs

For some keys the graph of the distribution function $f : d \mapsto \mathcal{P}_d$ is specially flat making the above method very hard to apply (see Figure 3). However, if we take the function $f' : d \mapsto std_d$, where std_d is the standard deviation of the frequencies of the non connected digraphs of distance d, its graph will reveal the half period as its maximum value (see Figures 4 and 5). Because pairs of letters, in the cryptogram, that are not at distances equal or around half of the period (respectively for the even and odd period cases, see figure 6) come from non contiguous letters in the text, this tends to flatten the original language statistical signature. But for those special distances, the digraph statistical peculiarities of the original language causes a slight increase of the standard deviation.

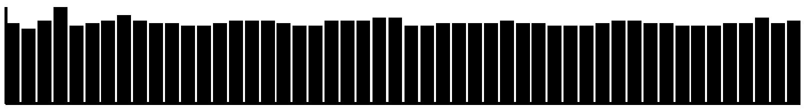


Figure 3: Example of a cryptogram that defeats method of Section 2.1 (period 8).

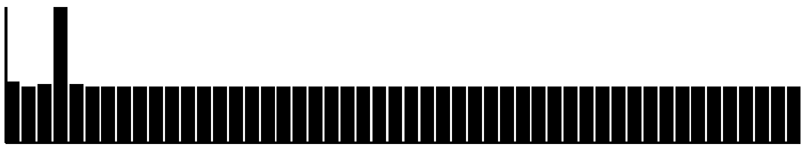


Figure 4: Example presented in Figure 3 defeated by method of Section 2.2.

3 How to find the key

All the previous known attacks on the Bifid cipher rely on the knowledge of some cribs (pieces of known plaintext). Because we intend to design a ciphertext-only attack, and although we use some results of Bowers [Bow60], we will essentially rely on the knowledge of statistical properties of the original language.

In the cryptanalysis of the Bifid cipher we used three different kinds of clues that are described below.

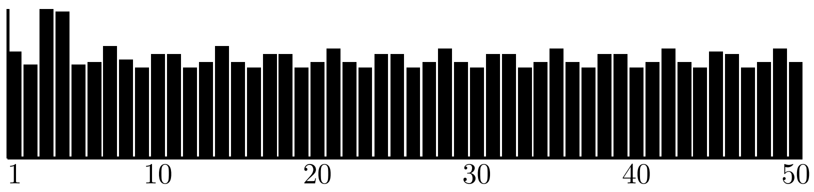


Figure 5: Standard deviation test for a cryptogram of period 7

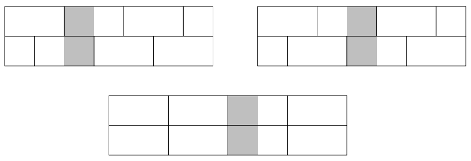


Figure 6: Interference of coordinates for odd and even periods.

3.1 Row and column differential probabilities

Observe that:

$$\begin{aligned}\sum_{\beta\in row(\alpha)} B_\beta &= \sum_{\beta\in row(\alpha)} \sum_{\substack{\gamma\in row(\beta)\\ \delta\in row(\beta^t)}} p(\gamma\delta) = \sum_{\beta\in row(\alpha)} \sum_{\substack{\gamma\in row(\alpha)\\ \delta\in row(\beta^t)}} p(\gamma\delta)\\ &= \sum_{\gamma\in row(\alpha)} \sum_{\beta\in row(\alpha)} \sum_{\delta\in row(\beta^t)} p(\gamma\delta) = \sum_{\substack{\gamma\in row(\alpha)\\ \delta\in\Sigma}} p(\gamma\delta) = \sum_{\beta\in row(\alpha)} p(\beta).\end{aligned}$$

This, and entirely analogous observation for $\sum_{\beta\in col(\alpha)} A_\beta$, leads to:

Theorem 2

$$\sum_{\beta\in row(\alpha)} (B_\beta - p(\beta)) = 0, \tag{11}$$

and

$$\sum_{\beta\in col(\alpha)} (A_\beta - p(\beta)) = 0. \tag{12}$$

Using the first of these equalities, that the sum of probability deviation of the entries of each row must be approximately zero, we can generate all the acceptable key-tables, although no information is available on the order of the entries in each row. To speed up this process, and to avoid the generation of each key table more than once, we direct the choice of the entries for each row by means of a sort of generalized Dirichlet principle: the value of a summand must be at least the value of the sum divided by the number of summands. To choose the first element of each row, we are free to choose one with maximal absolute deviation, from the set of the unused letters (after all, it has to belong to some row). The search for the other letters is narrowed by considering only the ones selected by the principle just stated. The equality (12) is used only at a latter stage of the algorithm.

3.2 Clues on line and column membership

The following scheme, based on an observation that goes back at least to Bowers [Bow60],

$\cdots$	σ_{2i-1}	σ_{2i}	σ_{2i+1}	σ_{2i+2}	σ_{2i+3}	$\cdots$
$\cdots$	x_{2i-1}	x_{2i}	x_{2i+1}	x_{2i+2}	x_{2i+3}	$\cdots$
$\cdots$	y_{2i-1}	y_{2i}	y_{2i+1}	y_{2i+2}	y_{2i+3}	$\cdots$

$$\downarrow$$

$\cdots$	τ_i	τ_{i+1}	τ_{i+2}	$\cdots$	$\tau_{\frac{\ell+1}{2}-i-1}$	$\tau_{\frac{\ell-1}{2}+i}$	$\tau_{\frac{\ell-1}{2}+i+1}$	$\cdots$
$\cdots$	x_{2i}	x_{2i+2}	x_{2i+4}	$\cdots$	$*$	y_{2i+1}	y_{2i+3}	$\cdots$
$\cdots$	x_{2i+1}	x_{2i+3}	$*$	$\cdots$	y_{2i}	y_{2i+2}	y_{2i+4}	$\cdots$

makes it clear that, for an odd period greater than 5, if a five-letter group appears repeated in the plaintext, in an odd position relative to the period, this will produce the appearance of the following repetition in the ciphertext: `ABW?XCD` and `ABY?ZCD`, where `W` and `Y` belong to the same line, while `X` and `Z` belong to the same column of the key table. This can be used to shorten the row generation phase described above, as well to validate its output. Information about row membership is used directly in the generation, while information on column membership can be used as a negative filter.

3.3 Clues on the diagonal and transposed pairs

In a ciphertext generated from a random text one should expect

$$B_\alpha \simeq \rho_\alpha \rho_{\alpha^t}, \quad \forall \alpha \in \Sigma. \tag{13}$$

Therefore, $\max\{B_\alpha : \alpha \in \Sigma\}$ should be attained at the letter in the diagonal on the line with maximum ρ value. Moreover, the values B_α $(\alpha \in \Sigma)$ should be distributed in pairs, since $B_\alpha \simeq \rho_\alpha \rho_{\alpha^t} \simeq B_{\alpha^t}$. Actual experiments show that in a ciphertext generated from a real text, among the top five values of the sequence B_α $(\alpha \in \Sigma)$ one tends to observe a letter of the diagonal, and a pair of transposed letters. Entirely similar remarks hold for the sequence A_α $(\alpha \in \Sigma)$.

In order to crack the Bifid, besides the composition of all rows and columns, one needs to know the diagonal composition. This determines, in conjunction with the period, the cipher. To obtain the diagonal member of a given row, the following observation can be used. From the ciphertext one readily obtains the value B_α for all the row members. From the original language statistics, one computes the right hand member of (3) using the given row for both variables. This value identifies the letter α in the row that has the right B_α value and thus belongs to diagonal. After obtaining in this way a candidate for the key table, one can then use (3) once more to further verify its soundness. In the end, one tries to decrypt the cryptogram using the candidate key, and validates the resulting text running it through a Friedman test [Bau97, MvOV96].

4 The program

As referred before, this method was successfully implemented as a computer program. The program has approximately $60K$ characters of well commented Python [pyt], and can obtain a solution for a cryptogram in less than 20 seconds for the majority of the cases, although it can take a couple of hours for some others. Because most of the tests and properties in which this method relies upon are of statistical nature, some limits for error tolerance must be tuned and provided as parameters for the algorithm. The explanation of the structure of the program can be better understood with the help of scheme in Figure 7. Given a cryptogram

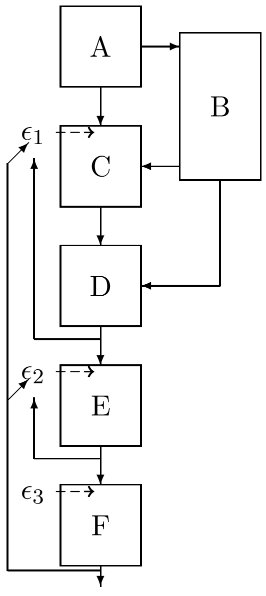


Figure 7: Scheme of the program structure

and the respective period, a first block, **A**, computes statistics for the frequencies of each character in positions in the cipher blocks corresponding to B, C and A (see Figure 1. Because this only depends on the value of the period, and on the cryptogram itself, it is computed only once. Statistics for the original language of the text must be also present, both for monographs and digraphs occurrences.

The method described in Section 3.2 depends only on the data already computed in block **A**. A new block of program, **B**, uses this information to infer the largest possible number of pairs that must belong to the same row (or to the same column). But because some pattern repetitions can be caused by "noise" introduced by the cipher and not by pattern repetitions in the text, some of the pairs can be erroneous and the whole set of restrictions become unsound. The transitive closure of the constraint pairs is evaluated, and the largest sound set of restrictions is collected, disregarding all with less occurrences than the first that originates a contradiction. This method does not ensure that all the restrictions kept are valid, although no example of the contrary was found in the numerous test runs of the program. This module contributes heavily for the reduction of the search space in block **C**, but alone is not enough, not even close, to permit the rest of the search as a brute-force approach. It is easy to see that although 16 well chosen pairs determine the row composition, it is possible to have 41 pairs and still be impossible to infer such composition. Even with texts with more than $100K$ characters, for some keys, the number of pairs inferred do not exceed 5 (!!). Once again this shows that the method described in Section 3.2 is not capable of solving the problem by itself.

The next block of program, **C**, is responsible for the generation of the row composition, *i.e.* what elements are inscribed in each row regardless the order of the entries inside each row and the order of the rows. Using restrictions produced by the **B** block, the program tries to fill the rest of the table taking in consideration what was described in Section 3.1. The equality (11) must be considered as an approximation, because it is going to be evaluated using two different sets of data: the one pertaining to the plaintext (that we ignore) is substituted by the language statistics; the other can directly obtained from the ciphertext. To evaluate the validity of the approximation, an error tolerance value ϵ_1 is thus needed. It is not possible to establish the "right" value for ϵ_1 for all cryptograms. The same value can be too loose for one problem, thus giving place to a search space larger than what is feasible to visit, and too tight for another, leaving the search space empty, or even worst desert of valid solutions. The solution for this problem was to assume a rather conservative value for ϵ_1, that normally originates a valid solution, but does not take too much time to compute, and in the eventuality of a set of solutions without any valid one, the process ins repeated with a larger value. Experience has shown that successive increments of 10% give good behaviour results. Each candidate solution is generated by a recursive call of the main method in this program block, and then goes through all the other stages of generation and validation of the other blocks. If in any stage a validation concludes that the key table (or some other preliminary structure) generated is not acceptable, the process stops for that instance, and backtracks to the last point where another choice is available. This use of recursion and depth-first search permits that the amount of memory wasted is relatively modest and do not constitute the bottleneck for the all process.

After row composition is established (in program block **C**), it is time for finding the order inside each row, *i.e.* the columns composition. This is accomplished in block **D**, using exactly the same method used in the previous block, and with the same value for error tolerance ϵ_1. If no solutions are find, program backtracks to block **C** with a higher value for ϵ_1.

As referred above, the key table is only completely determined after the composition of all rows and columns have been established, if it is possible to identify the elements of the diagonal. This is the role of block **E**. Using the observations made in Section 3.3 these letters are identified in each row, as being the ones that satisfy equation (13) for $\alpha = \alpha^t$. For the same reasons explained before, an error bound ϵ_2 is needed to evaluate acceptance. If no solution is provided by this block, the whole program returns to block **C**, relaxing ϵ_2.

Finally, block **F** evaluates each key-table. Using the key table in question, a try is made to decrypt the cryptogram, and the Friedman test is used on the resulting text. All tests whose coefficient differs from the known Friedman coefficient for the original language less than an error ϵ_3 are printed in the output. The program does not try to adjust the value of this limit ϵ_3. If some adjustment is necessary, it must be done "manually", and the whole process restarted. If no solutions are produced at the end of this block, the program returns to block **C**, alternating some relaxation on the limits ϵ_1 and ϵ_2. The solutions that pass this final test, as well as some segment of the tentative decryption are printed in the output. At this moment, human text recognition is the ultimate test. Amazingly, it is not uncommon to get "wrong" translations of the cryptogram, that pass Friedman test with better results than the original text! If the human does not find a suitable key in the set proposed by the program, what we discovered that is quite uncommon, the whole process should be restarted with all error limits (ϵ_1, ϵ_2 and ϵ_3) increased.

5 Concluding remarks

The method here presented, and its implementation as a computer program, permits to crack in an acceptable time any sufficiently large cryptogram, provided that the language of origin is know. It is not, because it essentially uses heavy statistical analysis, a system to solve small cryptographic puzzles such as those for which the Bifid cipher is currently used: for recreational purposes. "Typographical" ciphers such as this simply do not have good security properties when used for large messages. If modified to better resist this kind of attacks, they become too cumbersome to be used "by hand".

To what this attack is concerned, there is no difference in cryptographic resistance between ciphers of even and odd period. Some relations apply to the occurrence statistics in part C of ciphertext block, that only exists in odd period ciphers, but because of its less significance, no advantage on it could be obtained.

Although some cryptograms can be harder to break than others, because of the "noise" induced by the cipher, that result in a necessary relaxation of the error bounds in the program, no key-tables should be considered weaker than the others. It is to the coupling message-cipher that those characteristics belong, and not only to the cipher. By other hand, key-tables that show a good distribution of the maximum differential probabilities in both rows and columns, seem to be stronger in respect to Bowers' attack. From this results a much smaller set of *a priori* information possible to deduce on the composition of rows and columns. This can slow down the process of cracking, but not in a impeditive manner.

References

[Ame05] American Cryptogram Association. *The ACA and You — A handbook for the members of the American Cryptogram Association*, 2005.

[Bau97] F. L. Bauer. *Decrypted Secrets. Methods and Maxims of Cryptology.* Springer, 1997.

[Bow60] William Maxwell Bowers. *Practical Cryptanalisys (Volume II).* The American Cryptogram Association, 1960.

[Gai39] Helen Fouché Gaines. *Cryptanalysis. A study of ciphers and their resolution.* Dover Publications, 1939.

[Kah67] David Kahn. *The Codebreakers. The Story of Secret Writing.* Scribner, 1967.

[MvOV96] A. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography.* CRC Press, 1996.

[pyt] Python "official" web page. http://www.python.org.