# Code Theorie Project
# University Antwerp

Cédric Leclercq, Elias Dams, Robbe Nooyens

December 2022

## Contents

## 1 vigenerePlus

### 1.1 About

ViginairePlus is a chipher in which first viginaire and then single column transposition is applied. Two different keywords are used in this process.

I found it very difficult to find a suitable approach. Vigenère's weakness is Kasiski's test and coincidence index. However, if you put column transposition on top of Vigenère, that weakness is gone. The text is now shuffled and you can't search for digraphs/trigraphs because they give the wrong key length pattern. Nor can you solve for Columnar Transposition. Because the resulting text is still encrypted with vigenere.

If we want to brute force (assuming we both have keys with a maximum length of 10). we have to go through all the permutations. That makes a total of 16304741395569 possibilities. This is just not possible

$\sum_{i=1}^{10} i! \cdot \sum_{j=1}^{10} j! = 16304741395569$

### 1.2 Single-column transposition

We must first solve the column transposition. We do this by going over all possible permutations of maximum length 10 (otherwise it takes a long time) and making a selection. So we will have to identify how hard a transposed cyphertext resembles a vigenere chipher. As mentioned earlier, a coincidence index will not say much. Because only the order of the letters changes.

What we can do however is run a kasiski test. Kasiski suggested looking for repeated fragments in the ciphertext and making a list of the distances separating the repetitions. Then the length of the keyword is likely to divide many of these distances. With a non vigenere text this test will not be of much use and all key lengths are equally possible. However, with a vigenere text there will often be an outlier of a possible key length. If there is such an outlier I write it to a file. So after a run of the first part of my algorithm we have a file consisting of all the vigenere ciphertexts with a possible vigenere key length. So we have reduced the number of possibilities considerably.

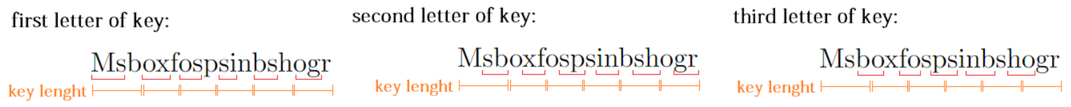
(if the outlier has length smaller than 4 then I manually remove it, because this is an unrealistically small key.)

## 1.3 Vigenère

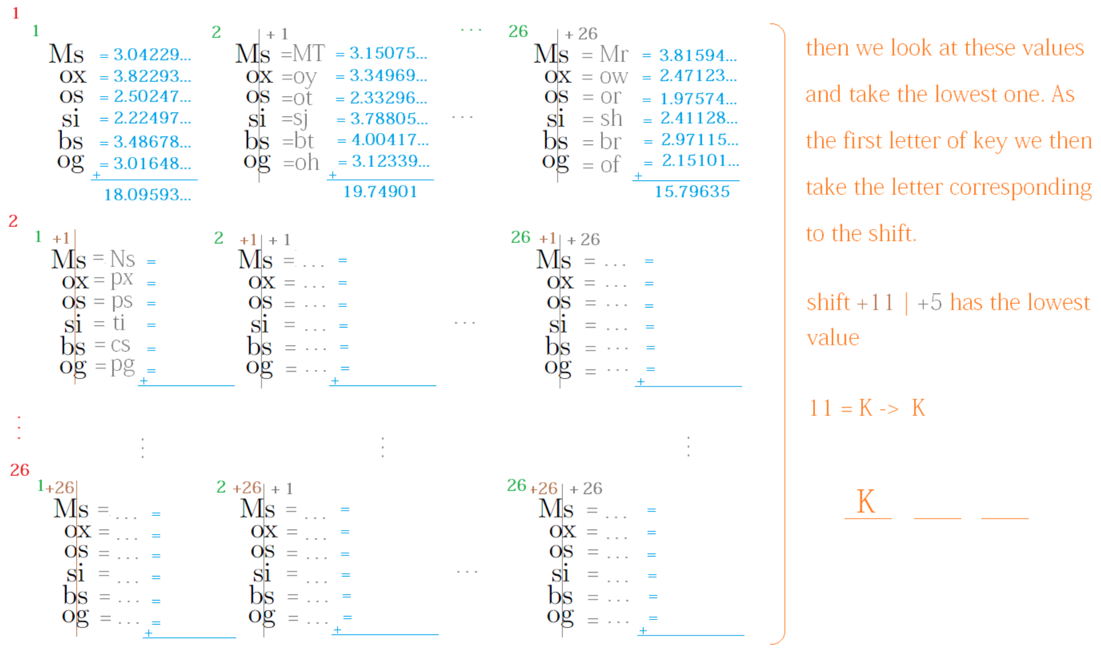
Now we only have a list of vigenere encrypted messages and their possible key lenght(s). To guess the key we are going to use the bigrams, and their appearance in the english language. I will explain the algorithm using an example.

Plaintext: Codetheorieproject
Ciphertext: Msboxfospsinbshogr
Key: KEY

We loop over the text and take the doubles that are 3 positions apart (keylenght). The doubles we are going to use to determine the first letter of the key are Ms, ox, os, si, bs and og.



Then we will calculate the entropy value ( from the table) of these doubles and all their possible shifts. At the end we look at the lowest entrpy value. We look at the shifts. Since our key is "KEY", +11 in the example will have the lowest entropy. The 11th letter is K. This is how we determined the first letter of the key. We repeat this for each letter of the key.



Then we take the final key and use it to find the plaintext. Then calculate the entropy of the resulting text. If it looks like English, we write it to a file.

(I noted that this doesn't always work for small texes, but the texes we got were large enough to find a correct solution each time).

## 2 playfair

## 3 adfgvx

### 3.1 About

This cipher is called ADFGVX because only these 6 letters are used in the ciphertext. These letters were chosen because their Morse code is so vastly different. The ADFGVXcode uses a fixed agreed substitution table, where each letter or digit is replaced by a digram. After this substitution is done another single-column transposition is done with a keyword.

### 3.2 Single-column transposition

So we will first have to reverse the column tranposition. As mentioned earlier with Vigenereplus, there are a lot of possibilities. So we will have to find a way to make a selection. We do this by going through all possible permutations of up to 10 lengths (otherwise it takes a long time) and selecting the text closest to English.

we calculate the index or coinxidece of each transposed text. We then consider the bigrams as 1 letter. If this index of coinxidece is similar to english (0.0667), we write the result to a file. I also write a list of the percentage of occurrence for every bigram to a file. This will make the next part much easier.

### 3.3 Manuel solution

Now we are going to look in the file and select the text with the smallest error (and therefore most similar to english). Fortunately, we found a text with an error (0.000454) that was much smaller than the other possibilities. So this one was almost certainly the correct transposed text.

Now comes the hardest part. We have to match the bigrams to their corresponding letter. We do this manually as we saw in the first lesson. We have created a spreadsheet that will help us do that. This spreadsheet shows the occurences of the letters in english and the occurences of the bigrams in text. after a bit of trail and error, we obtained the following substitution table:

| | **A** | **D** | **F** | **G** | **V** | **X** |
|---|---|---|---|---|---|---|
| **A** | ... | ... | ... | h | b | i |
| **D** | d | ... | p | l | ... | g |
| **F** | q | x | y | n | f | j |
| **G** | u | k | ... | c | r | m |
| **V** | s | ... | ... | ... | ... | v |
| **X** | a | o | t | z | w | e |

(the empty squares can be filled with the numbers 0-9.)

## 4 enigma

## 5 Plaintext Solutions

### 5.1 vigenerePlus

ColumnTransposeKey: `264573810`
VigenereKey: `charlie`
plaintext: `01-SOLUTION-vigenerePlus.txt`

### 5.2 playfair

### 5.3 adfgvx

ColumnTransposeKey: 032145
substitution table:

| | **A** | **D** | **F** | **G** | **V** | **X** |
|---|---|---|---|---|---|---|
| **A** | ... | ... | ... | h | b | i |
| **D** | d | ... | p | l | ... | g |
| **F** | q | x | y | n | f | j |
| **G** | u | k | ... | c | r | m |
| **V** | s | ... | ... | ... | ... | v |
| **X** | a | o | t | z | w | e |

plaintext: 03-SOLUTION-adfgvx.txt

### 5.4 enigma