Breaking the Hutton 2 challenge

Thomas Kaeding
summer, 2023

In 2018, Eric Bond Hutton posed a challenge online involving a classical cipher of his creation [1]. It was broken nearly two years later by brute-forcing the keywords [2], and a new challenge that involves a modified cipher was posted [3]. This is an explanation of how we broke the second challenge.

Before we begin, we need to see how the ciphers work, and what is the main weakness of each. Both can be considered “classical” ciphers, not because they were used before the invention of computers, but because they can easily be performed with pen and paper, or Scrabble® tiles. They act on a symbol set comprised of the letters of the alphabet, rather than bits 0 and 1. Whether they are easily broken with computers like their predecessors remains to be seen. We can say that because the results of encryption or decryption change drastically when the key is only changed slightly, hill-climbing attacks do not work. However, each version of the cipher has a weakness related to the encryption of a letter as itself, but how each weakness helps us is different for the two versions.

Hutton cipher 1

The Hutton cipher was invented by Eric Bond Hutton as an amusement and later as a means to challenge strangers on the internet [4][5][6]. It uses a pair of keywords to encrypt or decrypt a text. One keyword is used to set a 26-letter register that holds a permutation of the alphabet. The initial state of this register is determined in a way familiar to connoisseurs of classical ciphers. The alphabet is appended to the keyword, and then all repeated letters other than the first occurrence of each are deleted. For example, a keyword `SECRETKEY` would give us this register:

SECR~~E~~TK~~E~~YAB~~C~~D~~E~~FGHIJ~~K~~LMNOPQ~~RST~~UVWX~~Y~~Z

SECRTKYABDFGHIJLMNOPQUVWXZ

The other keyword serves to define a cyclically used set of shifts, much like that used in the Vigenère cipher [7]. The letters of the shift key are assigned numerical values `A`=1, `B`=2, ..., `Z`=26. When each letter of a text is encrypted, the ciphertext letter is found by looking for the letter that appears in the register to the right of it by the number of steps given by the shift, with roll-over. After each letter is encrypted, the plaintext letter and ciphertext letter swap positions in the register.

Let’s work through a quick example. Take the text

WE LAND AT MIDNIGHT

and the shift key ALIENS and alphabet keyword `FLYINGSAUCER`. The initial state of the register is

FLYINGSAUCERBDHJKMOPQTVWXZ

The shifts will be applied cyclically and are 1, 12, 9, 5, 14, 19, 1, 12, 9, 5, 14, 19, ...
The first plaintext letter is W, and the first shift is A=1, so the first ciphertext letter is X:

```
FLYINGSAUCERBDHJKMOPQTVWXZ
                       1→
```

Then the two letters are swapped in the register:

```
FLYINGSAUCERBDHJKMOPQTVXWZ
```

The next plaintext letter is E, and the next shift is L=12, so the second ciphertext letter is V:

```
FLYINGSAUCERBDHJKMOPQTVXWZ
          12→
```

We swap the two letters:

```
FLYINGSAUCVRBDHJKMOPQTEXWZ
```

This continues until we have the full ciphertext:

```
XV VBOS SB FUQRCFXS
```

Before moving on, we would like to mention that the swapping of letters in the register is the innovative feature that distinguishes H1 from the Quagmire 3 cipher (Q3) [8][9]. Without it, the two ciphers are identical, except for one detail. Because Hutton assigned 26 (= 0) to Z, rather than A, each letter of the shift key must be adjusted one place to the right in the alphabet (and Z goes to A) when going from H1 to Q3.

Weakness in Hutton cipher 1

The H1 cipher has a striking weakness: if the letter Z appears in the shift key, then its shift is 26, which is the same as 0 when roll-over is taken into account (mathematicians should be thinking of numbers modulo 26). Whenever the shift is 0, a plaintext letter is encrypted as itself. This leaks too much information to an attacker. To deal with this weakness, a cryptographer can decide never to use the letter Z in the shift key. However, this does not remove the weakness: if no plaintext letter can be encrypted as itself, some information is still leaked to the attacker. This weakness is shared by the Enigma machine used by Germans in WWII [10]. It allows an attacker to match a ciphertext (or part of a ciphertext) to a plaintext taken from a pool of texts.

This weakness is important to us because a variation of it will arise again in the Hutton 2 cipher.

The Hutton 1 challenge

The first Hutton challenge [1] is short enough to fit on a page. Here is the ciphertext:

```
WQQOZAYKTCUJACPCSZZJGRMFJRAALRVMYJACGYOZUDXYUPNIKVIVBMZKFHBCVOKD
CGBCXJJAVVQYUQTWMRYJECPFWTFLQDNTSKJKCKEQMYGLKWLCCUCGLFWDLKOATUNQ
GDGGYLUPZRWBSUTMTOUIISWYXHYWEJJIWCXZGWAKXOFFRQSGUFDVBBJHLRNEIEJD
EBFXNJWRNSRZRPBDXVWNPMMIHEFAXGCZVJYPRXRKZHJIXJOWKCWHUWQDTQMZVTSC
UUABXBIFUEDEBNXGBHHHUXCDBJUZNVRCAASWFFESDZYORKHUWUNVBKXVUJMDMXMY
CCMAZTOMPMINSORYYODDGOOYYYXNBJWJVFGYKXKYEMRCLXLZZRZUNIBKJTOCSNEA
GBVTXJHQGXDLWQBTEJTGKBKOD
```

The ciphertext was broken [2] by using brute force to find the shift key `MINOX` and alphabet keyword `QUARK`. The plaintext is from *The Napoleon of Notting Hill* by Gilbert Keith Chesterton:

```
FORHUMANBEINGSBEINGCHILDRENHAVETHECHILDISHWILFULNESSANDTHECHILDI
SHSECRECYANDTHEYNEVERHAVEFROMTHEBEGINNINGOFTHEWORLDDONEWHATTHEWI
SEMENHAVESEENTOBEINEVITABLETHEYSTONEDTHEFALSEPROPHETSITISSAIDBUT
THEYCOULDHAVESTONEDTRUEPROPHETSWITHAGREATERANDJUSTERENJOYMENTIND
IVIDUALLYMENMAYPRESENTAMOREORLESSRATIONALAPPEARANCEEATINGSLEEPIN
GANDSCHEMINGBUTHUMANITYASAWHOLEISCHANGEFULMYSTICALFICKLEDELIGHTF
ULMENAREMENBUTMANISAWOMAN
```

In a brute-force attack, every possible key is tried, the text is decrypted, and the resulting plaintext is scored on how well it appears to be English text. If the score is high enough, we believe we have found the correct key and stop searching. One way to score a plaintext is to find the frequencies of all three-letter sequences (trigrams) in typical English texts; for each trigram in the plaintext, its English frequency is added to the score.

Because we will use an analogous process for H2, we now describe a different way to break the first challenge which exploits the weakness in the cipher. Since an H1 cipher without a `Z` in the shift key never encrypts a letter as itself, we can determine that a plaintext matches if no letter in it is the same as the ciphertext letter at the same position. If we search all English texts on Project Gutenberg, we find a match at position 3238 (counting from 0) to ebook #20058 [11]. Due to the length of the ciphertext, the probability of a false positive result is very small.

Recovering the keywords from the correctly matched plaintext and ciphertext is another task altogether, and requires some knowledge of permutations. We go through the plaintext and ciphertext and every time letters in the register are swapped we keep track of them in a substitution table. In this way, whenever we see a plaintext-ciphertext letter pair, we can undo the substitution and see what the letters would have been when the register was in its initial state. We organize the unsubstituted pairs into bins, according to the length of the shift key and the position in the text. So, for example, for the challenge text above, with key length 5, the first bin contains

```
FW MA IZ CS QJ SC ZI DT LU KO EV RN
BP PB WF VE HY AM NR UL XG JQ TD GX
```

In order for consistency, no two distinct pairs in a bin can share the same first letter. Nor can they share the same second letter. This feature allows us to eliminate incorrect key lengths and to find the correct one. For the challenge text, consistency is obtained for key lengths of 5, 10, 15, ..., so we can take the correct value to be 5 with high confidence.

Once we have the correct shift key length and have populated our bins with letter pairs, we put them together into cycles. A cycle is a permutation that goes around in a loop. This will become clearer as we forge onward. Note that the rest of this section uses the same method that Gaines uses to find the keywords of a Q3 cipher [8]. In the first bin for key length 5, we listed the pairs above. Notice that `F`→`W` and `W`→`F` are both in the list, so we have a 2-cycle `F`→`W`→`F`, which we denote as (`FW`). On closer inspection, we see that the first bin is nothing but the product of thirteen 2-cycles:

(AM)(BP)(CS)(DT)(EV)(FW)(GX)(HY)(IZ)(JQ)(KO)(LU)(NR)

Two pairs are missing, but we don't worry about that, because from our experience with Quagmire ciphers [12], we know that Q3 permutations can only contain thirteen 2-cycles, two 13-cycles, one 26-cycle, or twenty-six 1-cycles, and earlier we noted the connection between H1 and Q3. By unsubstituting the letters, we have removed the effect of the swapping, which was the fundamental difference between H1 and Q3. A 2-cycle corresponds to a shift of 13, so we now know the first letter of the shift key: `M`. Let us now look at the second bin:

```
OQ MY KJ ZE EO UG BL AH HT JW CM FP GS
YD PU DN SA NZ LX WB QF XC TR RI VK
```

Here we can see the chain V→K→J→W→B→L→X→C→M→Y→D→N→Z→E→O→Q→F→P→U→G→S→A→H→T→R→I, which we denote

(AHTRIVKJWBLXCMYDNZEOQFPUGS)

The pair `I`→`V` is missing in the bin, but we are not concerned, since we are sure that this is a 26-cycle. A 26-cycle can only correspond to an odd shift that is not 13. So the second shift in the key is one of 1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25, or letter `A`, `C`, `E`, `G`, `I`, `K`, `O`, `Q`, `S`, `U`, `W`, `Y`. In the third bin we have

```
RO NK GY JU LA UM FX ZJ KP CT QL TE YI
OB XH IQ VF EW PC SD DV WG HZ BS AN
```

which can be collected into a product of two 13-cycles:

(ANKPCTEWGYIQL)(BSDVFXHZJUMRO)

A product of two 13-cycles indicates an even non-zero shift, so it is one of 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, and its letter is one of `B`, `D`, `F`, `H`, `J`, `L`, `N`, `P`, `R`, `T`, `V`, `X`. The fourth bin contains the pairs

```
HQ BT SE YJ DW JA WH UN NB MK ZL TF FY
VG GZ XI KS LR IU CV AO OC RP QM PD
```

which form one 26-cycle:

(AOCVGZLRPDWHQMKSEXIUNBTFYJ)

So the fourth key letter is again one of A, C, E, G, I, K, O, Q, S, U, W, Y. The fifth bin contains

```
UZ EC TP CK VS KA QY AQ BR ZX GE XV PN
LI SO IG OM MJ YW DB WT NL FD HF JH
```

They form two 13-cycles:

(AQYWTPNLIGECK)(BRUZXVSOMJHFD)

Here we get lucky and see YW and ZXV; since we might expect VWXYZ to appear at the end of the initial register, the fifth shift appears to be 24, so the last letter of the key is X. What we do next is take one of the 26-cycles, perhaps the one from the second bin, and take every n^{th} letter, where n is an odd number not equal to 13 (for the mathematicians, n must be invertible modulo 26), with roll-over back to the beginning whenever we reach the end:

n	
1	AHTRIVKJWBLXCMYDNZEOQFPUGS
3	ARKBCDEFGHIJLMNOPSTVWXYZQU
5	AVLDQSIBYOGRWMEUTJCZPHKXNF
7	AJYFTBNUIXESKMQHWDPRLZGVCO
9	ABEHLOTXQRCFIMPVYUKDGJNSWZ
11	AXPJERYSLFKZTMGBQVNHCUWOID
15	ADIOWUCHNVQBGMTZKFLSYREJPX
17	AZWSNJGDKUYVPMIFCRQXTOLHEB
19	AOCVGZLRPDWHQMKSEXIUNBTFYJ
21	AFNXKHPZCJTUEMWRGOYBISQDLV
23	AUQZYXWVTSPONMLJIHGFEDCBKR
25	ASGUPFQOEZNDYMCXLBWJKVIRTH

The most orderly choice is when n=3, where we see most of the letters in alphabetical order. Rotating it so that Z is at the end, we find the initial register state to be

QUARKBCDEFGHIJLMNOPSTVWXYZ

and the alphabet keyword is QUARK. We can now find the remaining letters of the shift key by grabbing one pair from each bin and finding the corresponding shift relative to this register state.

key position	pair	shift	letter
0	AM	13	M
1	OQ	9	I
2	RO	14	N
3	HQ	15	O
4	UZ	24	X

We now have the shift key: MINOX.

If we ever find ourselves in a position without a 26-cycle, we can make one from a collection of 2-cycles and 13-cycles. For example, from the first and third bins above, we had

```
FW MA IZ CS QJ SC ZI DT LU KO EV RN
BP PB WF VE HY AM NR UL XG JQ TD GX
```

and

```
RO NK GY JU LA UM FX ZJ KP CT QL TE YI
OB XH IQ VF EW PC SD DV WG HZ BS AN
```

We can combine `F`→`W` from the first set with `W`→`G` from the other set to get `F`→`G`. Doing this for all of the pairs gives us

(`ARKBCDEFGHIJLMNOPSTVWXYZQU`)

How lucky was that? Had we obtained nonsense, we would try taking every n^{th} letter, as we did earlier. If we only have 13-cycles to work with, things are a little harder. We need to interleave and take every n^{th} letter at the same time, while looking for a well-ordered list of letters. From the third bin we had

(`ANKPCTEWGYIQL`)(`BSDVFXHZJUMRO`)

One good interleaving is

$$\mathtt{A_{F}N_{X}K_{H}P_{Z}C_{J}T_{U}E_{M}W_{R}G_{O}Y_{B}I_{S}Q_{D}L_{V}}$$

Taking every fifteenth letter then gives a familiar result:

`ARKBCDEFGHIJLMNOPSTVWXYZQU`

Hutton cipher 2

The second Hutton cipher (H2) [13] is identical to the first, with one change introduced by Girkov Arpa: the shift is calculated as the sum of the numerical value of the appropriate letter in the shift key plus the numerical value of the first letter in the register. If the sum is more than 25, we can subtract 26 if we wish, to make the encryption process easier (for the mathematicians, we are working modulo 26). The purpose of this change is to allow a plaintext letter to be encrypted as itself, when the sum of the two values is 26 (the same as 0).

We won’t work through an example, but if you would like to check that you are using the H2 cipher correctly, here is a test vector:

plaintext:	`WE LAND AT MIDNIGHT`
shift key:	`ALIENS`
alphabet key:	`FLYINGSAUCER`
ciphertext:	`NY KOMB ZB DHNVBVZX`

The Hutton 2 challenge

Soon after the first challenge, the second Hutton challenge was posted online [3]. It has a ciphertext with 169,081 letters. We are not going to make you read 150 pages of it, but only give you the first few hundred letters and some of the last letters:

```
NJLNRNDKRFYAKISUNNBSMMDXCXBRDYFOSPCMSSVUGWFJPRENGHAQBETECKM
NTOWGRQPFYBKDVNHYBRGNLFAASMOJONVMMAIFQYHQLGMBDUHOJXBYHVLDFB
ETSMKZCWAMIJDZDUEWOUQQHQYOECPGACVXXSBJOUOAMAXSPHVDJNPTBMDCA
ZYRGCPXTQSNEZBPKBBNYJFBLNVYQORDKSKWDLKKGBULFYZZXPMNRPTRDZIF
XAHZTOHBEHJTEQWLTTGOSPQXPSIFNJOAVDUPVACBFLUIKUBYKQTTDTQXDJU
DUUWGBEQFUYEMSJTJODCCVZOSLYFJZCRTPHGTFMMOWUPUYJPPXTQUKKNGRL
EOHEHKEVTBKTURPWEFDMHRMNBRDAELMUXMYBCYMCXKVQBWBJUDOPLDBIBPS
MFECIDNWLYVYSLMBLBOPEVEWZATXOCRYSCOTHALHFLGWKVPWMUNCHBDWZXN
MQBSLVPCXYMXFNRTOCJOOZZSLMTO...RBEIVKDTOVGZCMLYDPNKXQDBIDHU
LDYFGMJMXKYHGILQPYUWSTUDCKCVDHHWLLYNWAMFNTYO
```

This will be enough so that you can verify for yourself that the keywords we find do indeed decrypt the text. Now let’s see if we can get a handle on it.

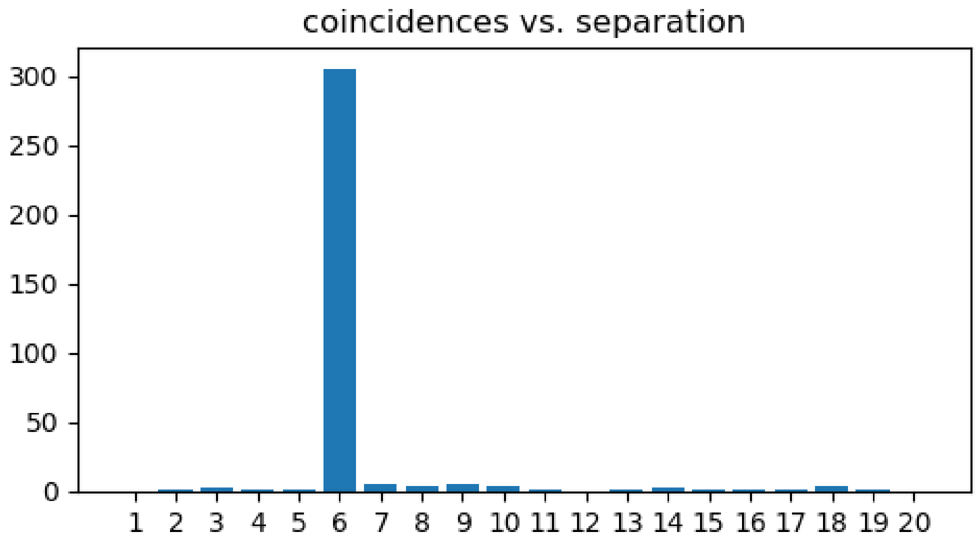
Weakness in the Hutton 2 cipher

In this cipher, a letter is encrypted as itself when the numerical values of the shift-key letter and the first letter of the register add to 26 (=0). Since the first letter of the register has a naively calculated 1/13 chance of changing with each position of the text, it is often the case that it remains in place for a full period of the shift key, which we can call L, the length of the shift key. Then another plaintext letter will be encrypted as itself, and the two will be L steps apart in the text. Occasionally we will even see long chains of plaintext letters peeking through at regular intervals. When the values of two letters add to 26, we will call them “complements.”

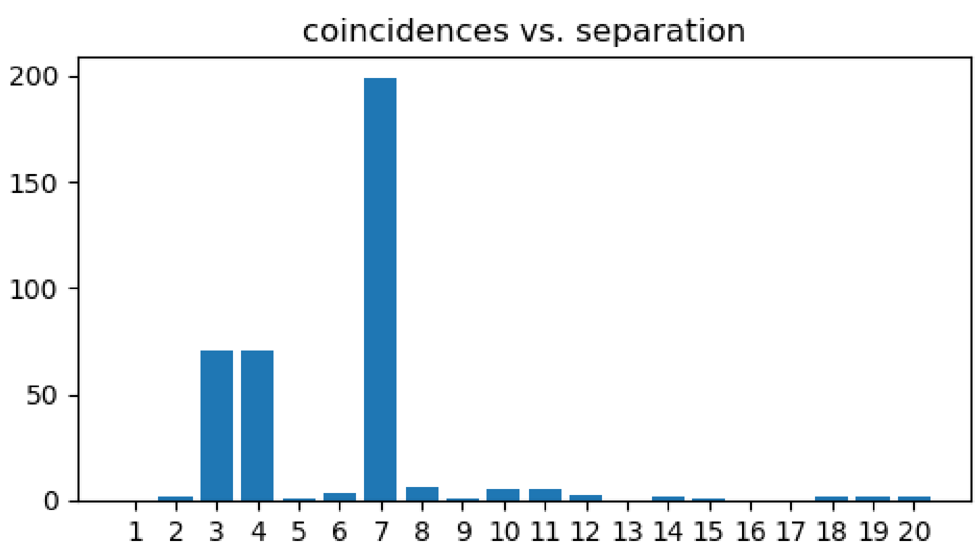
As an example, we can compare the first chapter (about 10,000 letters) of *The War of the Worlds* [14] with its encryption using the shift key `ALIENS`. Good examples of this phenomenon can be seen in positions 1850-1950 (counting from 0):

```
position:   11111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111
            88888888888888888888888888888888888888888888888888999999999999999999999999999999999999999999999999999
            55555555556666666666777777777788888888889999999999000000000011111111112222222222333333333344444444445
            01234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890

shift key:  23450123450123450123450123450123450123450123450123450123450123450123450123450123450123450123450123450

plaintext:  OTERFROMTHESUNITNECESSARILYFOLLOWSTHATITISNOTONLYMOREDISTANTFROMTIMESBEGINNINGBUTNEARERITSENDTHESECUL
ciphertext: NCCHRZWVXFLBYZDKSEXFZKPRFAUUWLKBZPQHPHGHOSORSFSIIIKOZPYPCPWOAYMLGQXWXBUBSJVOJIYAQEESDDKBTAKMBYWMHFSGG
```

With a magnifying glass we can see a chain of peek-throughs for the second letter of the shift key (denoted 1 because we count from 0), and later a single peek-through for the last letter of the key, and then a pair of peek-throughs for the first letter. If we scan the entire chapter, we accumulate this data:



There is a clear peak for a separation of 6. The small bits for other values of the separation are noise from the occasional change of the register's first letter from one that complements one letter of the shift key to one that complements another. If we encrypt with a shift key that has a repeating letter, like `MARTIAN`, we see secondary peaks:



These secondary peaks are due to a heartbeat pattern that occurs when the complement of the shift key's repeated letter occupies the first position of the register. Here we see an example of that pattern with repeating separations 4, 3, 4, 3, ...

position:	`3333333333333333333333333333333333333333344444444444444444444`
	`6666666666777777777788888888889999999999000000000001111111111 2`
	`0123456789012345678901234567890123456789012345678901234567890`

shift key:	`3456012345601234560123456012345601234560123456012345601234560`
plaintext:	`OMPLACENCYMENWENTTOANDFROOVERTHISGLOBEABOUTTHEIRLITTLEAFFAIRS`
ciphertext:	`DQNFMRNGTXZBNGAWTWWASKERTXVOLOHDCGMHOEPZOTALHNMRIJHTWYPXYOQYN`

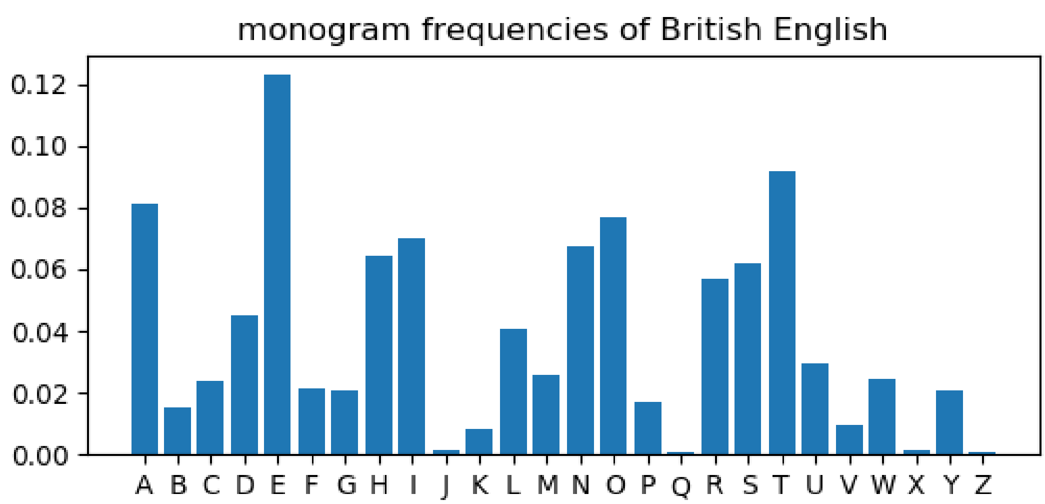
We still haven't explained why this feature of the cipher is a weakness. Like the fact that the H1 never encrypts a letter as itself, the patterns of peek-throughs for the H2 will allow us to match a plaintext to a ciphertext with high confidence. We will also see later that examining the letters near a peek-through can allow us to reconstruct the shift key, and that having the shift key and plaintext allows us to determine the contents of the cipher's register. In the next section we will even try to find the key length without knowing the plaintext.

An interesting thing happens when a lazy cryptographer uses the H2 cipher with a shift key of only one letter. Eventually its complement comes into the first position of the register, and from that point forward every letter is encrypted as itself. The first letter of the register never changes because at every subsequent position in the text a letter in the register is swapped with itself, i.e., not swapped at all. For example, when we encrypt the first chapter of *The War of the Worlds* with shift key `X` and alphabet key `FLYINGSAUCER`, eventually `B` moves into the first position of the register, and from that point onward the plaintext shows through:

```
UVFTWEDSIYRSJSCAYHRZXCNIJIJSJMYTPBZCBJYJZUDCRXMXBLYSVASBXII
PVYCOYRLEDWRLQUWDMTHBVNHCVVNAYKKEETEHTPRDNPWOVLLATNDRPLROQQ
CYPLJQNCOTVYWVJBIJXWXMSJGBEWKMQDDYLFFGLNWIDKTFZOYFPCMZDZVDM
CJVITFODMAOWMEPNNWFMIIVAJOZMBEYWERESCRUTINISEDANDSTUDIEDPER
HAPSALMOSTASNARROWLYASAMANWITHAMICROSCOPE...
```

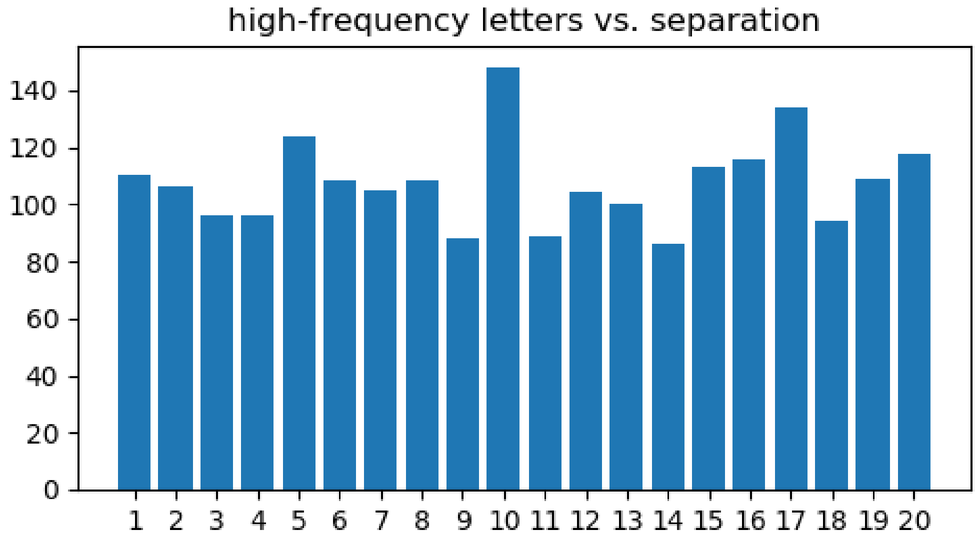
Finding the key length, part 2 (the previous section is part 1)

It is clear from the above discussion that when we have both the ciphertext and the plaintext we can find the length of the shift key. But what about the case when we only have the ciphertext? In this case, we do not know if a letter has been encrypted as itself or not. Or, at least we do not know with certainty. If we can guess correctly more often than random guessing, then we may be able to find it. A feature of natural language is that the frequency distribution of the letters is far from flat. If we focus on the more frequent letters in English, we can accomplish our goal of guessing better than random. We start by finding the frequencies of the twenty-six letters in a collection of British novels. For your consideration, here is what we find:

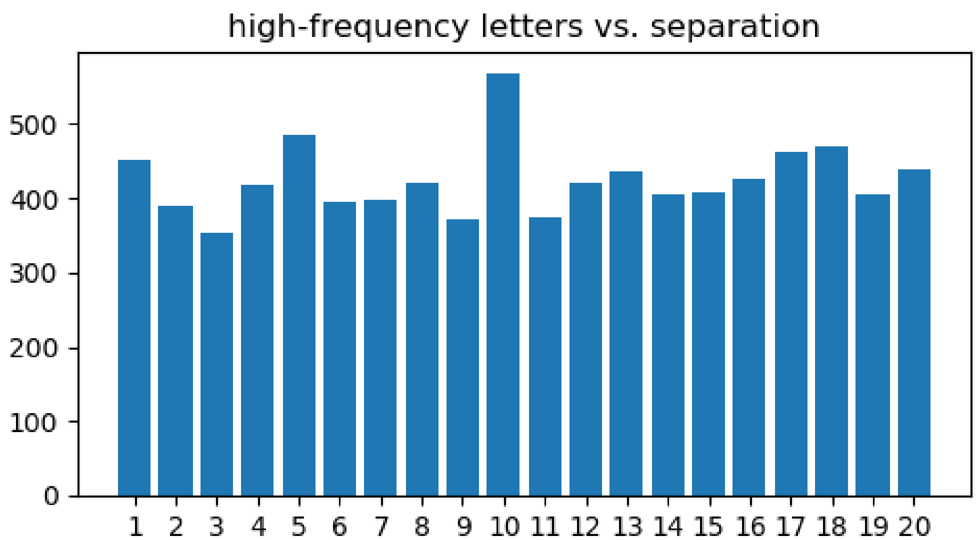


We can also improve our results by looking for chains of suspected peek-throughs, rather than singles.

When we scan the ciphertext for letters whose frequency in English is higher than 0.5, and we insist on the next six letters at a given separation form a chain of high-frequency letters, we obtain these results:



If we lower the frequency threshold to 0.3, this is what we get:



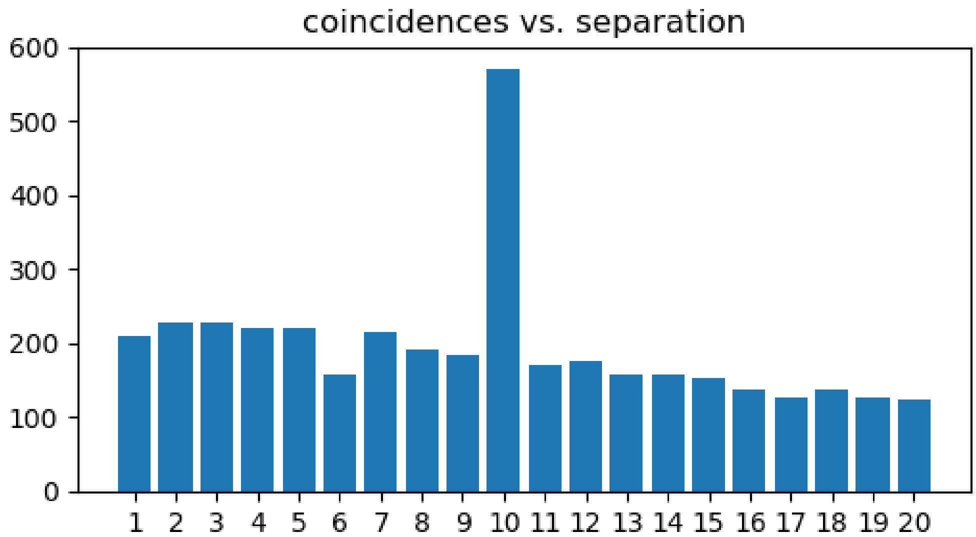
We can see that there is a definite peak above the noise at period 10, and possibly a secondary peak at 5. In retrospect we know that there are no repeating letters in the shift key, and we cannot explain the apparent secondary peak. Other choices of threshold and chain length give similar or noisier results. Nevertheless, we can conclude with caution that the length of the shift key is 10.

Finding a matching plaintext, first method

Unfortunately, a tentative value for the length of the shift key is all we are able to obtain from the ciphertext alone. Although it is quite long by most standards, it is too short for us to get any meaningful results for the values of the shifts, and without those, we are unable to make any progress on finding the contents of the register. So we have to take another approach. Earlier we mentioned that the periodic peek-throughs of plaintext letters constitutes a weakness that we could exploit to match the ciphertext with a plaintext. To accomplish this we downloaded all English books from Project Gutenberg and scanned them for a match. This took several days, and the result was a match at position 359470 (counting from 0) in *Shorter Novels, Eighteenth Century* [15]. The scanner reported that the best period (length of shift key) to be 10, and that there were 642 probable peek-throughs with that period. When we open that text to that position, we see

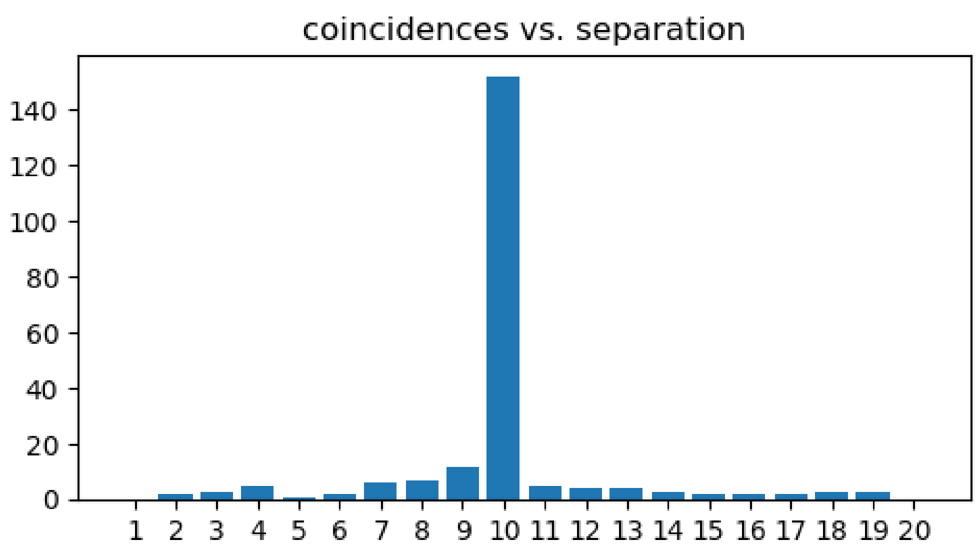
```
...o Bath where he created a miniature Fonthill. He died there on
2 May, 1844, his face showing scarcely a trace of age.

Bibliography: _Memoirs of Extraordinary Painters_, 1780. _Dreams, Waking
Thoughts, and Incidents, in a series of letters from various parts of
Europe_, 1783. _Vathek_ (Henley’s translation), 1784. _Vathek_ (in
Beckford’s French), 1787. _Letters from Italy, with Sketches of Spain
and Portugal_, 1835. See also memoirs by Cyrus Redding and Lockhart’s
review of Beckford’s letters in Vol. II of the _Quarterly Review_.

VATHEK

Vathek, ninth caliph of the race of the Abassides, was the son of
Motassem, and the grandson of Haroun al Raschid. From an early accession
to the throne, and the talents he possessed to adorn it, his subjects
were induced to expect that his reign would be long and happy. His
figure was pleasing and majestic: but when he was angry, one of his eyes
became so terrible, that no person could bear to behold it; and the
wretch upon whom it was fixed instantly fell backward, and sometimes
expired. For fear, however, of depopulating his dominions and making his
palace desolate, he but rarely gave way to his anger...
```

That position is not at an obvious break in the text, nor even is it at a word boundary; however, there is a major break shortly thereafter at the start of Beckford’s novel *Vathek*. This is our first clue that something else is going on, especially at the beginning of the text. Another clue comes when we look at the identical letters in this text with the challenge ciphertext as a function of separation:



We have a good strong signal-to-noise ratio, but significantly more noise than we saw earlier when we were working with examples for which the plaintext was known exactly. We do, however, have confirmation that the period is 10. The noise in this graph indicates that there may be some letters missing or added in going from the true plaintext to the candidate text that introduces a misalignment, as well as the discrepancy that we are sure to find at the beginning.

With these problems in mind, we look for a segment of the text that is well aligned to the ciphertext. For positions 2000 to 9000 we appear to have good alignment, as seen from the graph below, which was generated on only that region. Our analysis will continue in this region.



Finding a matching plaintext, second method

Alternatively, we could use the fact that the underlying cipher is a Quagmire 3 to match a plaintext to the ciphertext. For this we require about one hundred letters of each. There are twenty-six possibilities for the first letter of the register, and for each of them we can unswap the letters of both texts by keeping track with a substitution table. With the resulting texts we can extract some knowledge of the Q3 keys. Naively, we would think that we need $26 \cdot L$ letters (where L is the length of the shift key), since for each shift-key letter and each initial letter of the register there is a permutation of the alphabet in the tableau of the Q3. However, once we can establish a relative shift between two permutations corresponding to different shift-key letters, the other twenty-five pairs can also be equated. With a few letters in each of the permutations, techniques involving symmetry [15] or group theory [12] can fill in much of the missing letters. If there is a conflict of any kind, such as a permutation with two of the same letter, or an attempt to fill one space with two different letters, then we know that the texts do not match. We have decided not to go into further detail, since the previous method involves less computation and is faster.

Recovering the shift key, first method

We have two methods for recovering the shift key and register when the plaintext is know. In this section we describe a method using the letters near a plaintext peek-through; in the next section we recover the register. First, we look for a section containing a chain of peek-throughs. Consider this piece that starts at position 1520:

```
position:     1111111111111111111111111111111111111111111111111111111
              5555555555555555555555555555555555555555555555555555555
              2222222222333333333344444444445555555555666666666677777
              0123456789012345678901234567890123456789012345678901234

shift key:    0123456789012345678901234567890123456789012345678901234

plaintext:    SWHICHWERESUPPLIEDBOTHBYNIGHTANDBYDAYACCORDINGTOTHEIRCO
ciphertext:   YFIFGRXBYYLDRYLUQMNPMOJHNWHIRHXIFODXZOSAXHKZOZWADBCLGFP
```

For shift-key letter 4 (the fifth one, since we count from 0) we have a short chain of three peek-throughs. We know that the letter complementary to shift-key letter 4 was in the first place in the register when those three plaintext letters were encrypted. Because the ciphertext and plaintext letters do not match at position 1524, but do match at 1534, we know that the complementary letter was moved into first place during one of the ten steps leading up to the chain, and it therefore must appear in one of the two texts in one of those ten steps. So the complementary letter must be in the set

$$A = \{\mathrm{B, C, D, E, G, H, L, P, R, S, U, W, X, Y}\}$$

In the lead-out from the end of the chain the first letter of the register must be swapped out before encryption reaches position 1564, so the relevant letter is also in the set

$$B = \{\mathrm{A, C, D, H, I, K, N, O, R, S, T, X, Y, Z}\}$$

Now, between two peek-throughs in the chain, the relevant letter cannot appear exactly once, for otherwise it could have been moved out of the register's first place. So the relevant letter is not in the set

$$C = \{\mathtt{A}, \mathtt{B}, \mathtt{D}, \mathtt{E}, \mathtt{F}, \mathtt{G}, \mathtt{I}, \mathtt{J}, \mathtt{N}, \mathtt{O}, \mathtt{P}, \mathtt{Q}, \mathtt{R}, \mathtt{T}, \mathtt{U}, \mathtt{W}, \mathtt{X}, \mathtt{Y}\}$$

The resulting possibilities are

$$(A \cap B) \setminus C = \{\mathtt{C}, \mathtt{H}, \mathtt{S}\}$$

Now let's look at this block starting at position 5053:

```
position:     55555555555555555555555555555555555555555555555555555555555555555555555555555
              00000000000000000000000000000000000000000000000111111111111111111111111111111
              55555556666666666777777777788888888889999999999000000000011111111112222222222
              34567890123456789012345678901234567890123456789012345678901234567890123456789

shift key:    34567890123456789012345678901234567890123456789012345678901234567890123456789

plaintext:    UNWELCOMEPERCEPTIONOFHISLITTLENESSWITHTHETHOUGHTOFBEINGGREATINTHEEYESOFOTHERS
ciphertext:   RIYKCQLKRIRRKRITNWCIXDWJHIPYEPPAVJXIIYSDYZUHBGFYWSOADSNGWUFHAIPJMEUSCFBYKFHIY
```

We have a single peek-through for shift-key letter 4 and a chain of them for letter 8. The complement to letter 4 must be in

$$D = \{\mathtt{C}, \mathtt{E}, \mathtt{I}, \mathtt{K}, \mathtt{L}, \mathtt{M}, \mathtt{N}, \mathtt{O}, \mathtt{P}, \mathtt{Q}, \mathtt{R}, \mathtt{W}, \mathtt{Y}\}$$

It must move out of the first place in the register before position 5068 where the other chain begins. So it must also be in

$$E = \{\mathtt{C}, \mathtt{E}, \mathtt{I}, \mathtt{K}, \mathtt{P}, \mathtt{R}\}$$

So the complementary letter is

$$(A \cap B \cap D \cap E) \setminus C = \{\mathtt{C}\}$$

and the fifth letter of the shift key is

$$26 - \mathtt{C} = 26 - 3 = 23 = \mathtt{W}$$

While we are here, we can find the ninth letter of the shift key from the chain. From the transition from position 5064 to 5068, its complementary letter must be in the set E above. From the lead-out in positions 5119-5127, we know that the complementary letter must be in the set

$$F = \{\mathtt{B}, \mathtt{C}, \mathtt{E}, \mathtt{F}, \mathtt{H}, \mathtt{K}, \mathtt{O}, \mathtt{S}, \mathtt{T}, \mathtt{U}, \mathtt{Y}\}$$

Within the chain we see that it cannot be a member of this set:

$$G = \{\mathtt{A}, \mathtt{B}, \mathtt{C}, \mathtt{D}, \mathtt{E}, \mathtt{F}, \mathtt{G}, \mathtt{H}, \mathtt{I}, \mathtt{J}, \mathtt{L}, \mathtt{M}, \mathtt{N}, \mathtt{O}, \mathtt{P}, \mathtt{R}, \mathtt{S}, \mathtt{T}, \mathtt{U}, \mathtt{V}, \mathtt{W}, \mathtt{X}, \mathtt{Y}, \mathtt{Z}\}$$

What we have left is

$$(E \cap F) \setminus G \;=\; \{\mathtt{K}\}$$

so that the ninth shift-key letter is

$$26 - \mathtt{K} \;=\; 26 - 11 \;=\; 15 \;=\; \mathtt{O}$$

Finding chains for the complementary letters to the other shift-key letters allows us to reconstruct the shift key completely, and usually works if we have more than 5,000 letters in each of the two texts. The final result is

```
KUDYWSMHOC
```

Recovering the register at position 2000, first method

Now that we have the shift key, we need to recover the contents of the register at a point in the safe region where we have confidence in the plaintext. We are going to find the contents just before the letter at position 2000 is encrypted. For this it turns out that we only need to look at the first 136 positions in the texts:

```
position:    222222222222222222222222222222222222222222222222222222222222222222222
             000000000000000000000000000000000000000000000000000000000000000000000
             000000000011111111112222222222333333333344444444445555555555666666666
             012345678901234567890123456789012345678901234567890123456789012345678

shift key:   012345678901234567890123456789012345678901234567890123456789012345678

plaintext:   EDINTHEMOSTDELIGHTFULSUCCESSIONTHEPALACENAMEDTHEDELIGHTOFTHEEYESORTHE
ciphertext:  IESUTDKJTEYKIHICSFZTHHZRHEHVNAKIPQNAKVASKKNOITVPIRRTWUASNUYEJQNTYDOMO


position:    2222222222222222222222222222222222222222222222222222222222222222222
             0000000000000000000000000000000111111111111111111111111111111111111
             6777777777788888888889999999999000000000011111111112222222222333333
             9012345678901234567890123456789012345678901234567890123456789012345

shift key:   9012345678901234567890123456789012345678901234567890123456789012345

plaintext:   SUPPORTOFMEMORYWASONEENTIREENCHANTMENTRARITIESCOLLECTEDFROMEVERYCOR
ciphertext:  SIFUTNMXCEEBGEOLVWXGAPLWWESTVWMLSTIKDMNJPXMIOIWUWRXWYEEMWZUNBTPBURT
```

As we work, we need to keep track of the swapping of letters in the register; for this purpose we will maintain a translation table. Just before position 2000 in the text, it has this state:

```
original:    ABCDEFGHIJKLMNOPQRSTUVWXYZ
current:     ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

The first step is to find the first position at which the plaintext and ciphertext letters are the same; this occurs at position 2004. There, the fifth shift-key letter is in effect, but the total shift is 26=0, so the first letter in the register between positions 2003 and 2004 is the complement of the shift-key letter W, which is C. We see that C never is swapped in positions 2000-2004, so we know that it occupies the first place in the register just before 2000. We mark it in the translation table with an underline.

At position 2000 we can calculate the shift. It is the sum of the values of the first shift-key letter and the first letter in the register:

$$K + C = 11 + 3 = 14$$

The two letters that exchange places in the register at position 2000 are E and I. So we have a fragment of the register's contents in which E and I are 14 steps apart:

```
E_____________I___________
```

We swap E and I in our translation table:

```
original:   AB<u>C</u>DEFGHIJKLMNOPQRSTUVWXYZ
current:    ABCDIFGHEJKLMNOPQRSTUVWXYZ
```

At position 2001 the shift is the sum of the second key letter and C:

$$U + C = 21 + 3 = 24$$

The letters that swap at that position are D and E, but from the translation table we see that E used to be an I. So we have a fragment in which D and I are 24 steps apart:

```
D_______________________I_
```

Swap D and E in the translation table:

```
original:   ABCDEFGHIJKLMNOPQRSTUVWXYZ
current:    ABCEIFGHDJKLMNOPQRSTUVWXYZ
```

We can combine the two fragments that we have into a single fragment:

```
D_________E_____________I_
```

The letter C leaves the first place in the register at position 2015. After that point, our translation table is in this state:

```
original:   ABCDEFGHIJKLMNOPQRSTUVWXYZ
current:    ABGDIFCKLMSHJUYPQREONVWXTZ
```

and we have collected these fragments:

```
D_H_______E_____KS____L_I_
N_U_______________________
```

```
J______________M__________
O_____________Y___T_______
C___G_____________________
```

At the next step, we calculate the shift with the new first letter in the register (`G`) and the seventh letter of the key:

$$\texttt{G} + \texttt{M} = 7 + 13 = 20$$

The letters `H` and `S` swap positions in the register, but these correspond to `L` and `K` in the register from before position 2000, as seen in the translation table. So we have this fragment:

```
L___________________K_____
```

The information in this fragment is already contained in an earlier one; we just wanted to be sure that the method is understood.

Eventually (136 steps), we have enough to fill in the register completely before position 2000. Remembering that `C` must take first place, we write

```
CPEOGAFBKSRWMZLQIYDXHTNJUV
```

To verify the result, we can decrypt the ciphertext starting from position 2000 with the shift key and using `CPEOGAFBKSRWMZLQIYDXHTNJUV` as the alphabet keyword.

Recovering the shift key and register at position 2000, second method

Earlier we saw that the keys of a Hutton 1 cipher can be found from the cipher- and plaintext by reducing the problem to a familiar one concerning the Quagmire 3. The Q3 problem can then be solved with symmetry or group theory [8] [12] [15]. If we know the first letter of the register at some starting position, we can reduce the problem in H2 to an equivalent problem in H1, and then to a problem in Q3. We do not know the first letter in the register, but by trying all twenty-six letters and insisting on consistency between the cipher- and plaintext, we can narrow the choices down to one. We then find a 26-cycle, as we did earlier, and look at twelve ways of taking every n^{th} of its letters. For each, a shift key can be found from the first ten letters of the text (we already know the shift-key length to be ten; if we did not, then we would soon find out that ten is the smallest that gives consistent results). We must decrypt the text with each of the twelve pairs of keys to find the one that gives the correct plaintext from position 2000 onward.

Look again at the text after position 2000:

```
position:    222222222222222222222222222222222222222222222222222222222222222222222
             000000000000000000000000000000000000000000000000000000000000000000000
             000000000011111111112222222222333333333344444444445555555555666666666
             012345678901234567890123456789012345678901234567890123456789012345678

shift key:   012345678901234567890123456789012345678901234567890123456789012345678

plaintext:   EDINTHEMOSTDELIGHTFULSUCCESSIONTHEPALACENAMEDTHEDELIGHTOFTHEEYESORTHE
ciphertext:  IESUTDKJTEYKIHICSFZTHHZRHEHVNAKIPQNAKVASKKNOITVPIRRTWUASNUYEJQNTYDOMO
```

Earlier, when we looked at the H1 cipher, we kept track of the swaps of letters with a substitution table. We do that here as well. But whereas earlier we kept pairs of cipher- and plaintext letters in separate bins for each letter of the shift key, here we have the complication of adding the first letter of the register to each shift, and we do not know the shifts yet, so we need to keep twenty-six bins for each key letter. Suppose we begin by assuming that A is the first letter in the register. It will remain there until position 2029 of the text. If we look at the bin for shift letter 6, the first pair we have is from position 2006, where we see E and K. But E must be replaced by D because of the swap at 2001. So the pair is DK. The second pair for this bin is at 2016, where we see H and S. But H must be replaced with L because of the swap at 2013, and S must be replaced by K because of the swaps at 2006 and 2009. So this pair is LK. This is inconsistent with DK, since two letters cannot be encrypted to the same letter when the shift is the same. Therefore we reject A as the first letter in the register. An inconsistency occurs within the first 150 positions for every choice except C; we therefore conclude that the first letter of the register at 2000 is C.

Suppose we traverse one thousand positions in the text and bin the pairs. At the end of that time, the largest number in any one bin is eleven. But we know from studying Quagmires that there are only twenty-six alphabetic permutations for any one key. So we must merge bins together when possible and use other techniques when not. There are 260 bins. We can immediately discard any empty bins. That eliminates 0/K, 1/K, 2/K, 2/U, 3/B, 3/K, 3/O, 3/U, 4/B, 4/K, 4/O, 4/U, 5/B, 5/K, 5/O, 5/U, 6/B, 6/K, 6/U, 7/B, 7/K, 7/U, 8/B, 8/K, 8/U, 9/B, 9/K, and 9/U. Bins 0/O, 1/E, 2/V, 3/A, 4/C, 5/G, 6/M, 7/R, and 9/W all involve a shift of zero:

key letter #	register letter	pairs
0	O	MM LL
1	E	QQ TT CC NN
2	V	HH QQ KK LL
3	A	EE NN BB LL ZZ VV OO YY
4	C	TT EE AA XX WW LL GG
5	G	SS TT UU VV YY RR
6	M	YY MM NN WW DD TT AA CC PP HH
7	R	ZZ
9	W	GG TT ZZ

We can merge them all into one bin and fill in the missing pairs, if we wish. From these nine bins we know nine of the shift-key letters: the register letter is the complement of the shift-key letter. So key letter 0 is

$$29 - \text{‘O’} = 26 - 15 = 11 = \text{K}$$

The other letters that we can find in this way give us

KUDYWSMH_C

We can merge bins if they share at least one pair. For example, 0/A and 3/M can be merged:

key letter #	register letter	pairs
0	A	WJ GI LC YO PZ KH
3	M	BX AY OQ DG NK FD MU WJ YO

One of them can be deleted, and the other will contain the thirteen unique pairs above. Another way to combine bins is to notice that one may contain pairs that are inverses of the other's. For example, the new contents of 0/A finds a reciprocal in 6/A:

key letter #	register letter	pairs
0	A	WJ GI LC YO PZ KH BX AY OQ DG NK FD MU
6	A	TS JW GD PQ EI ZP

In such a case, we can add the inverses of each pair from one bin to the other bin:

key letter #	register letter	pairs
0	A	WJ GI LC YO PZ KH BX AY OQ DG NK FD MU ST QP IE
6	A	TS JW GD PQ EI ZP IG CL OY HK XB YA QO KN DF UM

A third way to combine bins is to form one new one from two old ones. In 0/A we have WJ, and in 2/G we have JK. So the composition of them is WK. Here is something that may be surprising: In forming the new bin, we can go in both directions, so CW in 2/G and WJ in 0/A gives us CJ (mathematicians: the Quagmire 3 cipher uses a set of permutations that is isomorphic to $\mathbb{Z}_{26}$, so everything commutes).

key letter #	register letter	pairs
0	A	WJ GI LC YO PZ KH BX AY OQ
2	G	FY IP JK CW AI VR SH NB
-	-	WK LW FO IZ JH CJ NX

We have described all of the possible operations on bins that we can perform, with the except of a house-cleaning operation that allows us to add the missing pair to a bin that has twenty-five pairs. We look for the letter that does not appear as the first letter of any of the pairs, and pair it with the letter that does not appear as the second letter in any of the pairs.

We combine bins and generate new bins to combine with old bins, until we reach the point that we have one bin with twenty-six pairs in it. For the one thousand positions in the text that we used, there are four such bins. One of them contains these pairs:

```
AH BN CQ DB EY FT GX HS IA JM KJ LO MP
NW OD PI QG RV SU TR UZ VL WC XK YF ZE
```

From them we can build a 26-cycle (remember that C belongs in first place):

(CQGXKJMPIAHSUZEYFTRVLODBNW)

This is not necessarily the contents of the register. It is one of twelve options, each of which is found by taking every n^{th} letter of this cycle, where n is 1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, or 25 (for the mathematicians: these correspond to the automorphisms of $\mathbb{Z}_{26}$). For each resulting candidate, we can find a shift key by calculating the shifts needed to encrypt the first ten letters (starting at position 2000). With the candidate register and its corresponding shift key, we decrypt part of the text starting at that position, and compare it to the known plaintext. The one candidate that gives a true decryption is the correct register.

n	candidate register	shift key	decryption
1	CQGXKJMPIAHSUZEYFTRVLODBNW	QITKWUEVSM	EDINTHEMOSTDELIG**RZZYDJYNCAYHNOR...**
3	CXMAUYRONQKPHZFVDWGJISETLB	UAVSWEQNMK	EDINTHEMOSTDELIG**KMFFIMDRKQSLSPH...**
5	CJHYLWKAEVNXIZRBGPUTDQMSFO	AOLEWGIBQU	EDINTHEMOSTDELIG**CBFMJGQTZSLARTN...**
7	**CPEOGAFBKSRWMZLQIYDXHTNJUV**	**KUDYWSMHOC**	EDINTHEMOSTDELIG**HTFULSUCCESSION...**
9	CARQHVGSLXUOKZDJEBMYNPFWIT	EGNMWQUTKS	EDINTHEMOSTDELIG**NPFTPPGMYIEEATO...**
11	CSDPRXEWHOMTGZNALJFQUBIVKY	MQRCWKSDYO	EDINTHEMOSTDELIG**SWFPLSKNRYHCRQF...**
15	CYKVIBUQFJLANZGTMOHWEXRPDS	GCBQWIAPUE	EDINTHEMOSTDELIG**IAFORUIHNMDDAHL...**
17	CTIWFPNYMBEJDZKOUXLSGVHQRA	OMFGWCYZIA	EDINTHEMOSTDELIG**JQFAKLIZZBKJGHJ...**
19	CVUJNTHXDYIQLZMWRSKBFAGOEP	IYPUWAGLEQ	EDINTHEMOSTDELIG**UVFQCRSVCQIMCYN...**
21	COFSMQDTUPGBRZIXNVEAKWLYHJ	SEHOWMKRCY	EDINTHEMOSTDELIG**FWEVHCLUDAZKVOK...**
23	CBLTESIJGWDVFZHPKQNORYUAMX	YSXAWOCFGI	EDINTHEMOSTDELIG**LXFENHRQUQCJKMU...**
25	CWNBDOLVRTFYEZUSHAIPMJKXGQ	CKZIWYOXAG	EDINTHEMOSTDELIG**GPFTPPSQOKJJFPI...**

We find that the register contents are

CPEOGAFBKSRWMZLQIYDXHTNJUV

and that the shift key is

KUDYWSMHOC

which is consistent with the nine letters that we found from the bins with zero shift.

Working back to the beginning

Running the cipher backwards is easy so long as the plaintext is true. All we must do is swap the letters in the register at each step. But when the putative plaintext that we have from Gutenberg no longer matches the original, still hidden, plaintext, there are problems. The first issue we encounter is at position 1041.

position:	1111111111111111111111111111111111111111
	0000000000000000000000000000000000000000
	3333333333444444444455555555556666666666
	0123456789012345678901234567890123456789
shift key:	0123456789012345678901234567890123456789
ciphertext:	HAVKTENWIYONIGBSWSLSMKKDYQLDUZBCDTFGETYD
gutenberg:	HISINDULGENCESUNRESTRAINEDFORHEDIDNOTTHI
plaintext:	????????????ESUNRESTRAINEDFORHEDIDNOTTHI

The state of the register between positions 1041 and 1042 is

```
BURIHONPQJWXKLDTMVCSGFAEYZ
```

The shift at position 1041 is

$$\mathtt{U} + \mathtt{B} \;=\; 21 + 2 \;=\; 23$$

The letter 23 steps after the ciphertext letter `N` in the register is `I` (trust us, we have everything in the right direction). So the plaintext letter there should be an `I`. Once that letter is inserted into the Gutenberg text, things run smoothly again until position 390, which is the beginning of *Vathek*.

From the beginning of the novel to the beginning of the plaintext, we proceeded much as we did above at position 1041. However, there were stumbling blocks. At each position there are three possibilities, and there is no a priori way to know which is correct:

1. The first letter of the register does not change, and we can use the same technique as at position 1041.

2. The plaintext letter is swapped into the first place in the register; in this case it is simple to take the first letter of the register and tentatively use it as the plaintext letter at that position. If we continue and see nonsense in the decryption, then we know that we have made the wrong choice.
3. The ciphertext letter is swapped with the first letter of the register; in this case we are reduced to trial and error in finding the plaintext letter.

There were times when we had to make educated guesses at the next letter, or try random letters, and check to see that the decryption was consistent. In the end, or rather at the beginning, we have the correct first two thousand letters of the plaintext and the initial contents of the register, which are

```
NCVPEBGKLZADFHIJMOQRSTUWXY
```

so that the alphabet keyword is

```
NCVPEBGKLZ
```

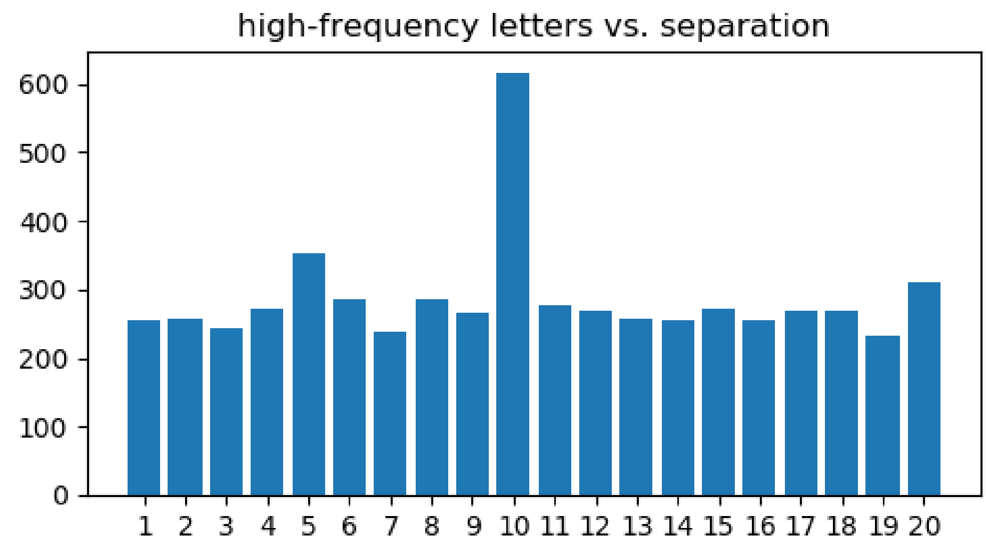
And here is the (partial) decryption of the ciphertext, including Eric Bond Hutton's introduction and the last few words:

```
CONGRATULATIONSYOUHAVESUCCESSFULLYDECRYPTEDACIPHERTEXTOFONE
HUNDREDANDSIXTYNINETHOUSANDANDEIGHTYONELETTERSENCRYPTEDWITH
THEMODIFIEDVERSIONOFHUTTONCIPHERAPRIZEOFTENTHOUSANDPOUNDSST
ERLINGAWAITSTHEFIRSTPERSONPOSTINGACORRECTDECRYPTIONOFTHISCI
PHERTEXTTOREDDITSCODESBOARDORALINKONTHATBOARDTOSUCHADECRYPT
IONWHATFOLLOWSISTHETEXTOFVATHEKBYWILLIAMBECKFORDTHANKYOUFOR
YOURINTERESTINMYCIPHERERICBONDHUTTONVATHEKNINTHCALIPHOFTHER
ACEOFTHEABASSIDESWASTHESONOFMOTASSEMANDTHEGRANDSONOFHAROUNA
LRASCHIDFROMANEARLYACCESSION...PASSEDWHOLEAGESINUNDISTURBED
TRANQUILLITYANDINTHEPUREHAPPINESSOFCHILDHOOD
```

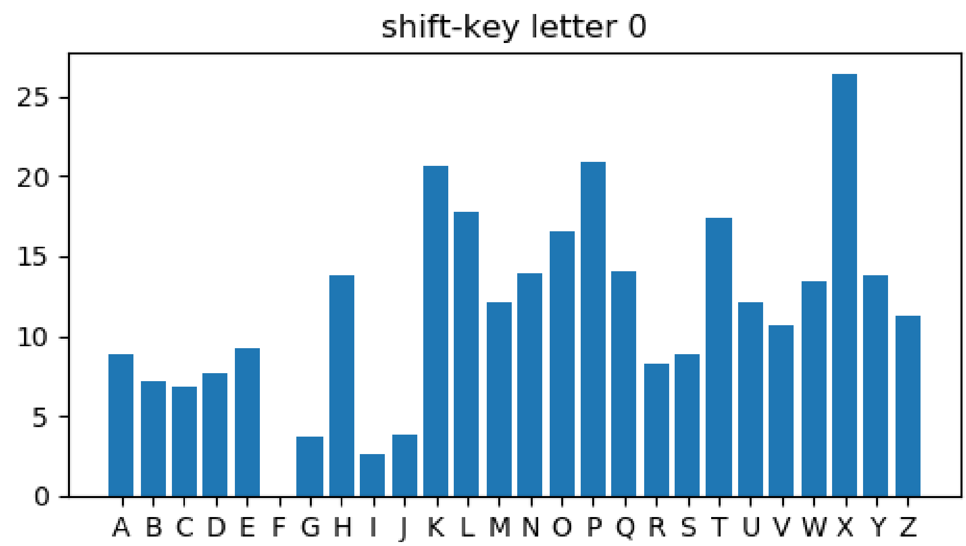
Concluding remarks

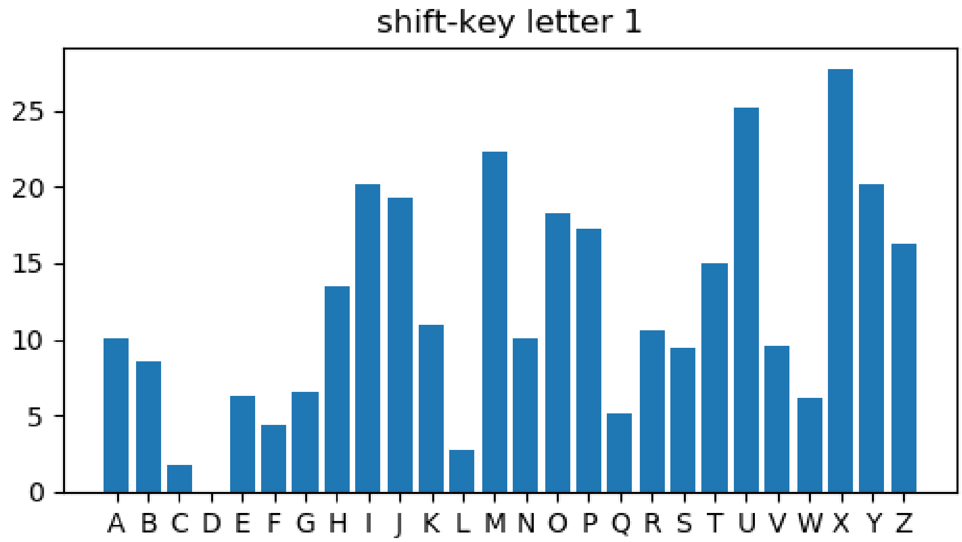
We would like to thank Eric Bond Hutton for an interesting challenge. We also thank him for reading an early draft of this paper and giving his feedback.

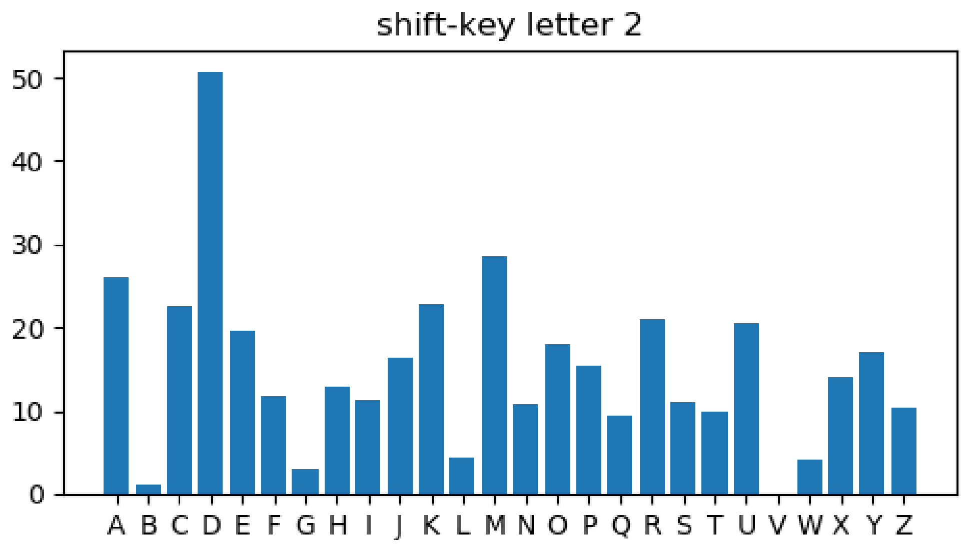
It may be possible to refine the statistical analysis that earlier gave us the length of the shift key. It may further be possible to use it to find letters of that key by looking at probable chains of leaked plaintext letters, which we choose as high-frequency occurring at regular intervals in the ciphertext. Whether any definite result could come of this method for the challenge ciphertext is work in progress. As a means of demonstration of what we hope to accomplish, we made a ciphertext of ten million letters encrypted with the challenge keys. Requiring chains of length at least ten of letters whose English frequency is above 0.05 gave us this graph for the key length:

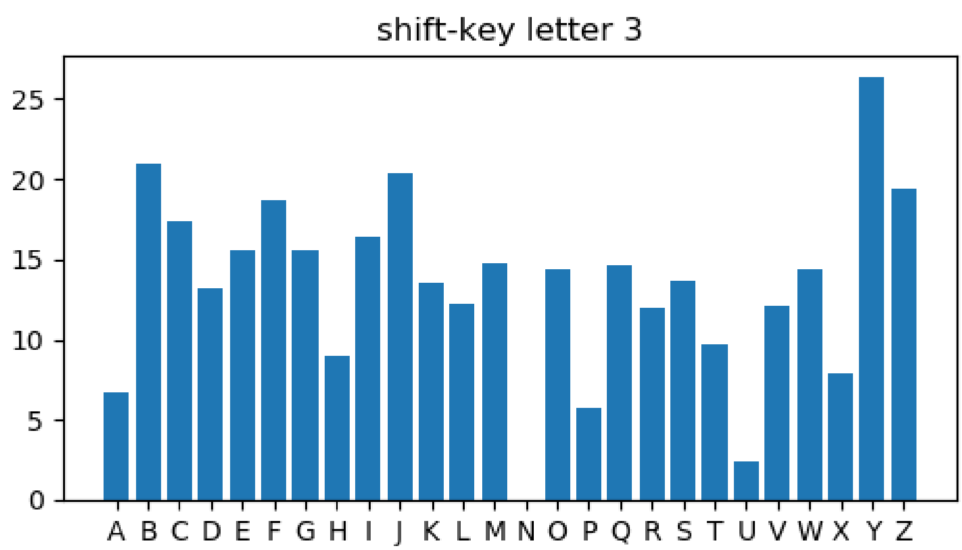


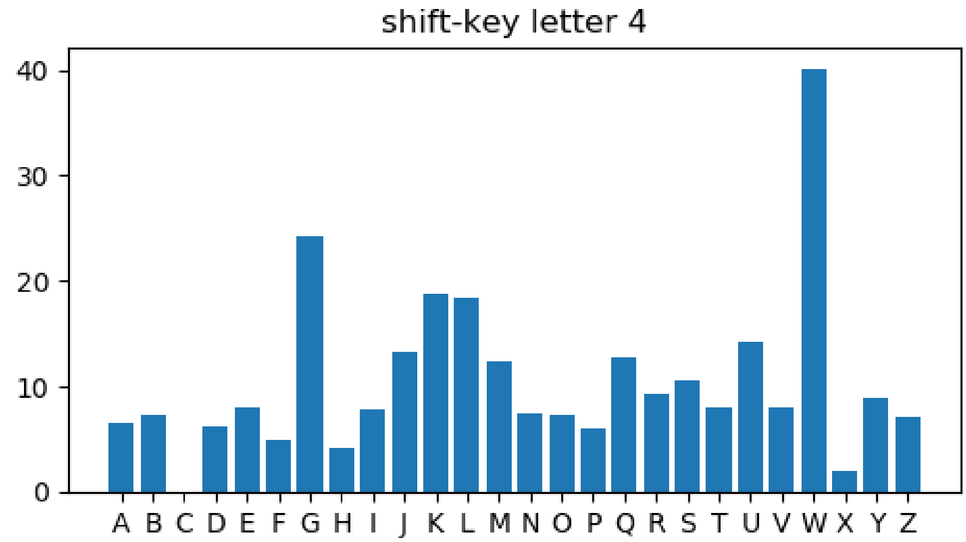
For the same ciphertext, we were able to recover eight of the ten letters of the shift key, from examining only the lead-outs from chains of high-frequency letters. The ten relevant graphs follow.

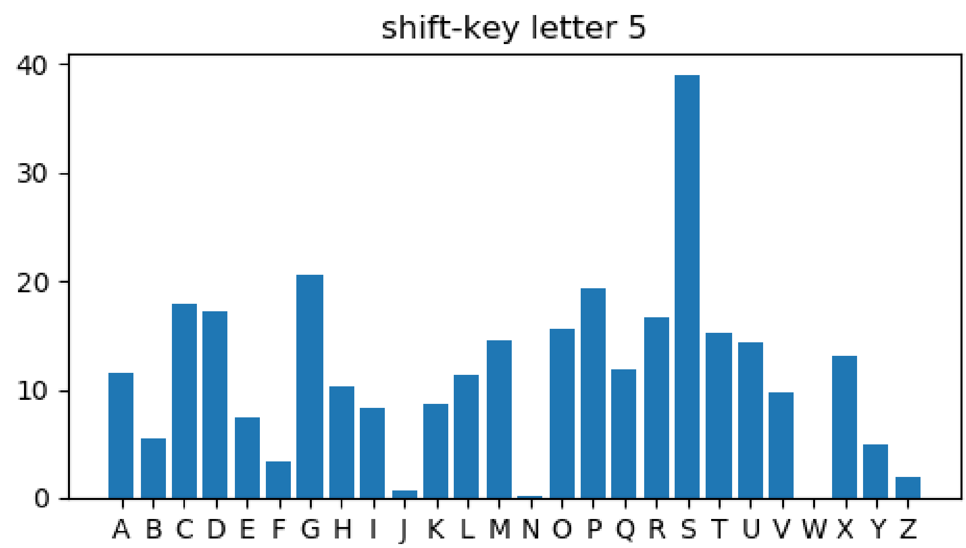


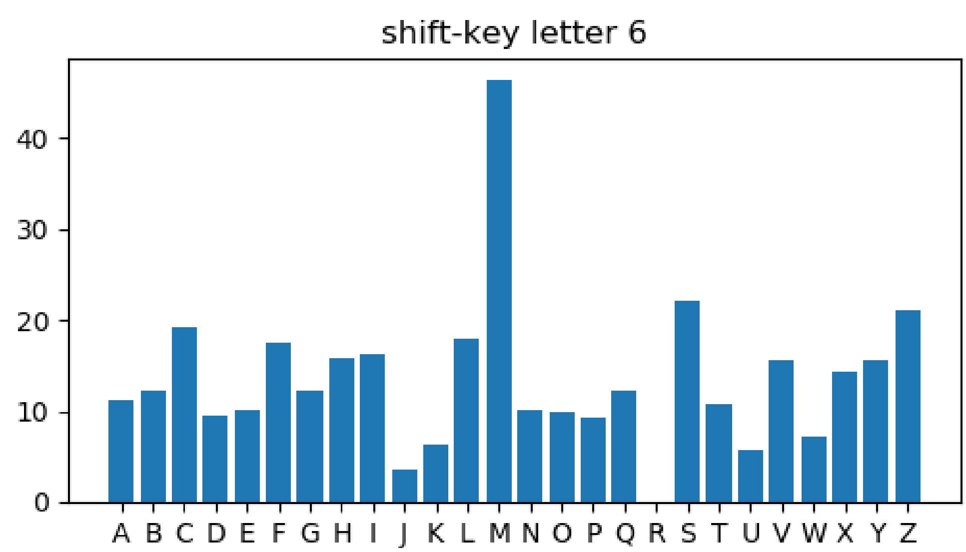

shift-key letter 1
0
5
10
15
20
25
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

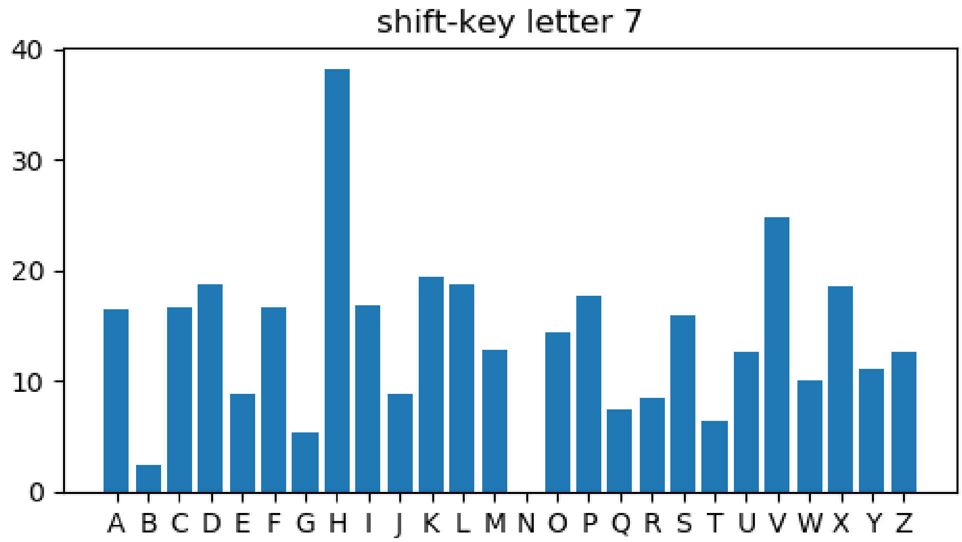

shift-key letter 2
0
10
20
30
40
50
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

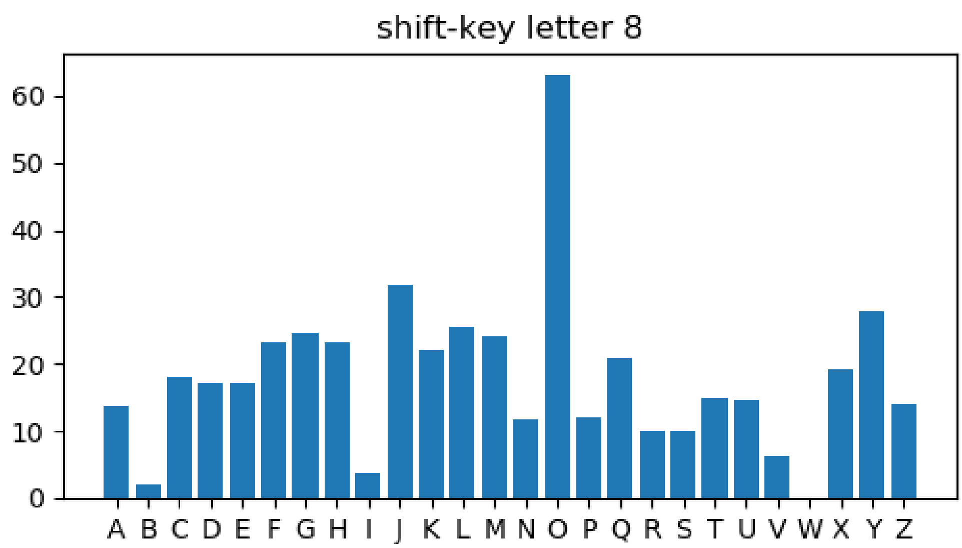

shift-key letter 3
0
5
10
15
20
25
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

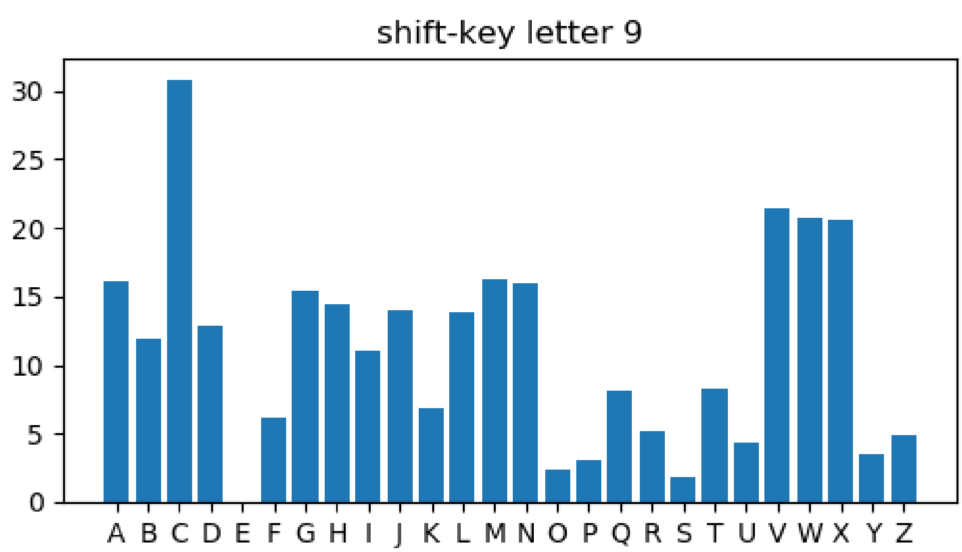

shift-key letter 4
0
10
20
30
40
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z


shift-key letter 5
0
10
20
30
40
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z


shift-key letter 6
0
10
20
30
40
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z


shift-key letter 7
0
10
20
30
40
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z


shift-key letter 8
0
10
20
30
40
50
60
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z


shift-key letter 9
0
5
10
15
20
25
30
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

References

[1] The first Hutton challenge: www.reddit.com/r/codes/comments/8mp1h2, www.reddit.com/r/cryptography/comments/8mozl5, www.reddit.com/r/codes/comments/93ygp2

[2] Solution to the first challenge: www.reddit.com/r/codes/comments/ffgpef

[3] The second challenge: www.reddit.com/r/codes/comments/ar1lbd

[4] “Hutton Cipher v1,” huttoncipher.netlify.com

[5] Girkov Arpa, “How It Works,” hutton-cipher.netlify.app/howto.html

[6] Eric Bond Hutton, “Hutton Cipher,” en.wikipedia.org/w/index.php?title=User:Eric_Bond_Hutton&oldid=840686562

[7] Blaise de Vigenère, *Traicté des chiffres ou secrètes manières d’escrire*, Paris: Abel l’Angelier, 1586, HDL: 2027/ien.35552000251008, gallica.bnf.fr/ark:/12148/bpt6k1040608n, gallica.bnf.fr/ark:/12148/bpt6k94009991

[8] Helen Fouché Gaines, *Cryptanalysis: a study of ciphers and their solution*, New York: Dover, 1956; previously titled *Elementary Cryptanalysis* and published by American Photographic in 1939; archive.org/details/cryptanalysis00gain; chapter XVIII

[9] American Cryptogram Association, “The ACA and You,” www.cryptogram.org/cdb/aca.info/aca.and.you/aca.and.you.pdf; 2005 version: web.archive.org/web/*/http://www.cryptogram.org/cdb/aca.info/aca.and.you/aca.and.you.pdf; 2016 version: web.archive.org/web/*/http://cryptogram.org/docs/acayou16.pdf; the relevant page can also be found at www.cryptogram.org/downloads/aca.info/ciphers/QuagmireIII.pdf

[10] David Kahn, *The Codebreakers: The Story of Secret Writing*, New York: Simon & Schuster, 1967, revised and updated 1996. I have forgotten which of the 978 pages mentions this weakness in the Enigma.

[11] Gilbert Keith Chesterton, *The Napoleon of Notting Hill*, Project Gutenberg ebook #20058, www.gutenberg.org/ebooks/20058

[12] Thomas Kaeding, “Classical substitution ciphers and group theory,” Cryptology ePrint Archive, report 2023/669, eprint.iacr.org/2023/669

[13] “Hutton Cipher v2,” huttoncipher2.netlify.com

[14] H. G. Wells, *The War of the Worlds*, Project Gutenberg ebook #36, www.gutenberg.org/ebooks/36

[15] William F. Friedman, The Principles of Indirect Symmetry of Position in Secondary Alphabets and Their Application in the Solution of Polyalphabetic Substitution Ciphers, Washington D.C.: U.S. Government Printing Office, 1935, www.nsa.gov/Portals/70/documents/news-features/declassified-documents/friedman-documents/publications/FOLDER_226/41760669079979.pdf

[16] William Beckford, *Vathek, an Arabian Tale*, in *Shorter Novels, Eighteenth Century*, Project Gutenberg ebook #34766, www.gutenberg.org/ebooks/34766