The Vigenere Cipher

The key: A sequence of characters.

To make brute-force decryption impractical, the key should have at least 15 or 16 characters. Also, it should not be a "special" sequence. such as an English language word. It may be best if all letters of the key are distinct.

Encryption: Duplicate the key as many times as necessary, so that the length of the (duplicated) key matches the length of the plaintext.

For $i = 0, 1, 2, 3, \ldots$:

"Add" letter i of the key to letter the i of the plaintext, to obtain letter i of the ciphertext.

(In adding letters, we identify them with integers modulo 26: **a** $\rightarrow 0$, **b** $\rightarrow 1, \ldots,$ **z** $\rightarrow 25$.)

Example:

key: **`wonderland`** *(10 characters, not an ideal key)*

plaintext: **`alicewasbeginningtogetverytiredof`**
key (duplicated): **`wonderlandwonderlandwonderlandwon`**
ciphertext: **`WZVFINLSOHCWAQMERTBJAHIHVPEIEHZCS`**

We obtained letter 5 the ciphertext like this:

$$\begin{array}{rcl} \mathbf{w} & \rightarrow & 22 \\ +\,\mathbf{r} & \rightarrow & +\underline{17} \\ \mathbf{N} & \leftarrow & 13 \pmod{26} \end{array}$$

The 463 character plaintext

```
alicewasbeginningtogetverytiredofsitting
byhersisteronthebankandofhavingnothingto
doonceortwiceshehadpeepedintothebookhers
isterwasreadingbutithadnopicturesorconve
rsationsinitandwhatistheuseofabookthough
talicewithoutpicturesorconversationsoshe
wasconsideringinherownmindaswellasshecou
ldforthehotdaymadeherfeelverysleepyandst
upidwhetherthepleasureofmakingadaisychai
nwouldbeworththetroubleofgettingupandpic
kingthedaisieswhensuddenlyawhiterabbitwi
thpinkeyesranclosebyher
```

encrypts using the key **`wonderland`** to

```
WZVFINLSOHCWAQMERTBJAHIHVPEIEHZCSVMKEIAJ
XMUHVJTSGHNCAWLVMAANWBQRJYLVVQCBBWLZYGGR
ZCBQGVZRGZEQRVLVSAQSASCHHZYTBWDSORSBSEEV
EGGHVNLSEHWRVQKSFTVWDOQQSGTCGXNSFRVTZNIH
NGNWMFYSVQEHNQHNSAGLOHUHYJPOSDXCBNXYZUTK
POYLGVHIGKKIGSMTEUEHOCEFSEGEEVWHVRRJZSUH
SOFFSEDIQHNWAJMESEERSBZLRULSJHHZNVWYPCBX
HRSRVKSEURPRNBQROEUHNTRHPMPRLVHSRSCRYDFW
QDVGAYPTUHNHUHTCPAFXNSBIQRVIAJWRNLWPNHNL
JKBXPUMEJRNHUWLVERBXXZRRJXPTGLJUHSEEOPVF
GWAJXYPDNLOWRVAYPNFXZRRQPPLWULPSEDFSTTJL
PVCLRBPYRVNOAFPFDEOBDSE
```

In C, we could encrypt a plaintext using code like this:

```
for ( i = 0 ; i < textLength ; ++i )
   cipherText[i] = (plainText[i] - 'a' +
                    key[i % keyLength] - 'a') % 26 +
                    'A';
```

(This assumes plaintext and key consist entirely of lower case letters.)

Decryption: Like encryption, except we get the plaintext by <u>subtracting</u> letters of the (duplicated) key from letters of the ciphertext

Breaking Vegenere Ciphers: *(ciphertext only)*

A simple frequency analysis isn't useful. For example, with the 463 character ciphertext, we get the following frequencies.

Letter	*Frequency*
S	33
R	32
H	32
V	29
E	29
N	24
P	21
L	20
W	19
G	18
Q	16
B	16
A	16
J	15
U	14
T	14
F	14
Z	14
Y	13
C	13
O	13
X	12
D	10
I	10
M	9
K	7

The frequencies don't differ that much. (With a longer key, or a key with distinct letters, they would differ even less.)

Here is a method that often works, if we have enough ciphertext. It consists of two steps:

i) Find the *length of the key* (the period).

ii) Find the *key* itself.

Finding the key length: We perform a k-position right cyclic shift of a sequence (or vector) by moving each component k positions to the right. However, the last k positions are moved to the beginning.

For example, a 3-position right cyclic shift of

$\mathbf{v}_0\ \mathbf{v}_1\ \mathbf{v}_2\ \mathbf{v}_3\ \mathbf{v}_4\ \mathbf{v}_5\ \mathbf{v}_6\ \mathbf{v}_7\ \mathbf{v}_8$

gives

$\mathbf{v}_6\ \mathbf{v}_7\ \mathbf{v}_8\ \mathbf{v}_0\ \mathbf{v}_1\ \mathbf{v}_2\ \mathbf{v}_3\ \mathbf{v}_4\ \mathbf{v}_5$

If we have a fair amount of ciphertext, the following method often allows us to make a good guess at the key length.

For $k = 1, 2, 3, \ldots,$ (*largest likely key length*), do the following:

Perform a k-position right cyclic shift of the ciphertext.

Compare the cyclic shift with the original ciphertext, and count the number of positions in which they are the same. Call this number c_k.

If among c_1, c_2, c_3, ..., one of the numbers (say c_m) is significantly larger than the rest, then the key length is likely to be m.

If c_m, c_{2m}, c_{3m}, ... are comparable in size, and larger than the other numbers, the key length is likely to be m.

An example: Consider the 1840 character ciphertext below:

```
SNZCQXLWEWIZNZJYF...EUVURSEIPJHDNGA
```

To compute c_2, we compare the ciphertext itself with a 2-position cyclic right shift of the cipertext.

```
SNZCQXLWEWIZNZJYF...EUVURSEIPJHDNGA
GASNZCQXLWEWIZNZJ...CZEUVURSEIPJHDN
         ↑   ↑          ↑
```

We see three positions where they agree, but if we were to examine all 1840 columns, we would find 63 positions of agreement. So $c_2 = 63$.

With a computer, it is easy to compute all the c_m for, say, $m \le 30$.

m	c_m	m	c_m
1	79	16	67
2	63	17	77
3	66	18	113
4	49	19	75
5	70	20	71
6	72	21	70
7	65	22	78
8	59	23	60
likely value for key length → **9**	129	24	60
10	74	25	69
11	68	26	65
12	63	27	141
13	70	28	77
14	62	29	73
15	50	30	72

Why key length method works: Let $\underline{\mathbf{A}} = (0.0821, 0.0150, 0.0230, 0.0479, 0.1237, 0.0225, 0.0208, 0.0645, 0.0676, 0.0018, 0.0087, 0.0393, 0.0254, 0.0705, 0.0767, 0.0163, 0.0009, 0.0550, 0.0617, 0.0921, 0.0291, 0.0087, 0.0254, 0.0013, 0.0195, 0.0006)$

$= (p(\mathbf{a}), p(\mathbf{b}), p(\mathbf{c}), p(\mathbf{d}),\dots, p(\mathbf{y}), p(\mathbf{z}))$ in a typical English text

Let $\underline{\mathbf{A}}(i) =$ $\underline{\mathbf{A}}$ cyclic right shifted i positions.

$\underline{\mathbf{A}}(i)\cdot\underline{\mathbf{A}}(i) = \underline{\mathbf{A}}(0)\cdot\underline{\mathbf{A}}(0) \approx 0.0654$ (calculate sum of squares)

But if $j \ne i \pmod{26}$, then $\mathbf{A}(j)$ is not a multiple $\underline{\mathbf{A}}(i)$, and

$$\underline{\mathbf{A}}(i)\cdot\underline{\mathbf{A}}(j) = \underline{\mathbf{A}}(0)\cdot\underline{\mathbf{A}}(j-i) < \underline{\mathbf{A}}(0)\cdot\underline{\mathbf{A}}(0).$$

In fact, we can compute $\underline{\mathbf{A}}(0)\cdot\underline{\mathbf{A}}(j-i)$ for the various possible values of j with $j \ne i \pmod{26}$.

$j-i$	$\underline{\mathbf{A}}(0)\cdot\underline{\mathbf{A}}(j-i)$	$j-i$	$\underline{\mathbf{A}}(0)\cdot\underline{\mathbf{A}}(j-i)$
1	0.0399	14	0.0390
2	0.0304	15	0.0444
3	0.0346	16	0.0383
4	0.0438	17	0.0334
5	0.0336	18	0.0339
6	0.0361	19	0.0392
7	0.0392	20	0.0361
8	0.0339	21	0.0336
9	0.0334	22	0.0438
10	0.0383	23	0.0346
11	0.0444	24	0.0304
12	0.0390	25	0.0399
13	0.0415		

The values of $\underline{\mathbf{A}}(0) \cdot \underline{\mathbf{A}}(j-i)$ with $j \neq i \pmod{26}$ range between 0.0304 and 0.0444.

They are considerably less than $\underline{\mathbf{A}}(0) \cdot \underline{\mathbf{A}}(0) \approx 0.0654$.

Now let

n = length of plaintext and ciphertext.

$x_0 x_1 x_2 \ldots x_{n-1}$ = the plaintext.

$y_0 y_1 y_2 \ldots y_{n-1}$ = the ciphertext.

L = length of key. (We assume $L \ll n$.)

$k_0 k_1 k_2 \ldots k_{L-1}$ = the key (which will be duplicated). In general, k_i will denote $k_{i \bmod L}$.

If we compare the ciphertext with its m-position cyclic right shift, how many positions of agreement do we expect?

ciphertext:	y_0	y_1	…	y_{m-1}	y_m	y_{m+1}	…	y_{n-2}	y_{n-1}
ciphertext: *(cyclic right shifted m)*	y_{n-m}	y_{n-m+1}	…	y_{n-1}	y_0	y_1	…	y_{n-m-2}	y_{n-m-1}

What is the probability of a match in column i, i.e., $y_i = y_{i-m}$.

$$y_i = y_{i-m} \iff x_i + k_i = x_{i-m} + k_{i-m}$$

$$\iff (x_i + k_i = \alpha) \text{ and } (x_{i-m} + k_{i-m} = \alpha) \text{ for some } \alpha \text{ in } \{A,\ldots,Z\}$$

$$\iff (x_i = \alpha - k_i) \text{ and } (x_{i-m} = \alpha - k_{i-m}) \text{ for some } \alpha \text{ in } \{A,\ldots,Z\},$$

Since x_i and x_{i-m} are somewhat close to independent (except perhaps when $m = 1$), we obtain

$$p(y_i = y_{i-m}) \approx \sum_{\alpha=A}^{Z} p(x_i = \alpha - k_i)\, p(x_{i-m} = \alpha - k_{i-m}).$$

The sum on the right is approximately

$$\underline{\mathbf{A}}(k_i) \cdot \underline{\mathbf{A}}(k_{i-m}),$$

which equals

$$\underline{\mathbf{A}}(k_i - k_{i-m}) \cdot \underline{\mathbf{A}}(0),$$

assuming our plaintext is somewhat typical of an English language text.

Now

i) If m is a multiple of the key length L, then $k_i = k_{i-m}$ for all i, and

$$p(y_i = y_{i-m}) \approx \underline{\mathbf{A}}(0) \cdot \underline{\mathbf{A}}(0) = 0.0654.$$

So we expect to find the number c_m of matching positions to be about $0.0654n$.

ii) If m is not a multiple of the key length L, and if all the characters in the key are distinct, then $k_i - k_{i-m} \neq 0$ for every i, and

$$p(y_i = y_{i-m}) \approx \underline{\mathbf{A}}(k_i - k_{i-m}) \cdot \underline{\mathbf{A}}(0),$$

so $p(y_i = y_{i-m})$ should lie in the range $[0.030, 0.044]$, or at least close to it.

We expect c_m to be in the range $[0.030n, 0.044n]$, or close to it.

iii) If m is not a multiple of the key length L, and if *most* of the characters in the key are distinct, then $k_i - k_{i-m} \neq 0$ for *most* values of i.

c_m is a sum of n terms, most of which lie in the range $[0.030, 0.044]$, so we expect c_m to be fairly close to the range $[0.030n, 0.044n]$, if not within it.

Consider our previous 1840-character text. We computed the number c_m of matches of the ciphertext with an m-position cyclic right shift of the ciphertext, $m = 1, 2, 3, ..., 30$.

Here is the same data with c_m expressed as a fraction of the n.

m	c_m	m	c_m
1	$0.043n$	16	$0.036n$
2	$0.034n$	17	$0.042n$
3	$0.036n$	18	$0.061n$
4	$0.027n$	19	$0.041n$
5	$0.038n$	20	$0.039n$
6	$0.039n$	21	$0.038n$
7	$0.035n$	22	$0.042n$
8	$0.032n$	23	$0.033n$
9	$0.070n$	24	$0.033n$
10	$0.040n$	25	$0.038n$
11	$0.037n$	26	$0.035n$
12	$0.034n$	27	$0.077n$
13	$0.038n$	28	$0.042n$
14	$0.034n$	29	$0.040n$
15	$0.027n$	30	$0.039n$

This data is just what we would expect for key length $L = 9$.

i) For $m = 9, 18, 27$, c_m is $0.070n$, $0.061n$, $0.077n$ — all reasonably close to the expected $0.0654n$.

ii) For other values of m, c_m ranges from $0.027n$ to $0.043n$ — all in or close to the expected range of $[0.030n, 0.044n]$.

Finding the key: We assume we have found the length L of the key $k_0k_1...k_{L-1}$.

We find k_0 first. Each of the characters

$$y_0, y_L, y_{2L}, y_{3L}, y_{4L}, ..., y_{(q-1)L} \quad (q \approx n / L)$$

have been encrypted by adding k_0. (Essentially, if we restrict to these characters, we have a simple shift cipher with shift k_0.)

We count the frequency of each letter (A, B, ..., Z) in these q characters of the ciphertext, and divide these frequencies by q to obtain probabilities.

Let $\underline{\mathbf{W}} = (w_0, w_1, ..., w_{25})$, where $w_0, w_1, ..., w_{25}$ are the probabilities of A, B, ..., Z in positions $0, L, 2L, ..., (q-1)L$ of the ciphertext.

$w_0, w_1, ..., w_{25}$ are the probabilities of A$-k_0$, B$-k_0$, ..., Z$-k_0$ in the plaintext (same positions).

So $\underline{\mathbf{W}} \approx \underline{\mathbf{A}}(k_0)$ assuming these positions of our ciphertext resemble a typical English language text.

We can use the size of $\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(k_0)$ as a measure of how close $\underline{\mathbf{W}}$ is to $\underline{\mathbf{A}}(k_0)$.

We compute $\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(i)$ for $i = 0, 1, ...,25$.

If one value of i makes $\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(i)$ significantly larger than any other, that value of i is our best guess for k_0.

We can attempt to find $k_1, k_2, \ldots, k_{L-1}$ in a similar manner. For example, to find k_1, we would look at ciphertext characters y_1, y_{1+L}, y_{1+2L}, y_{1+3L}, y_{1+4L}, …

Example of finding the key:

For our 1840 character ciphertext, we compute:

$$\underline{\mathbf{W}} = (\ 0.0488, 0.0000, 0.0049, 0.0390, 0.0098, 0.0927, 0.1317, 0.0098, 0.0000, 0.0732, 0.0488, 0.0732, 0.0293, 0.0000, 0.0244, 0.0000, 0.0195, 0.0000, 0.0683, 0.0195, 0.0049, 0.0732, 0.1366, 0.0195, 0.0146, 0.0585)$$

$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(0) = 0.029$	$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(9) = 0.032$	$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(18) = 0.068$
$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(1) = 0.035$	$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(10) = 0.032$	$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(19) = 0.040$
$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(2) = 0.046$	$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(11) = 0.036$	$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(20) = 0.027$
$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(3) = 0.046$	$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(12) = 0.035$	$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(21) = 0.038$
$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(4) = 0.038$	$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(13) = 0.037$	$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(22) = 0.040$
$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(5) = 0.044$	$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(14) = 0.047$	$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(23) = 0.031$
$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(6) = 0.039$	$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(15) = 0.036$	$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(24) = 0.038$
$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(7) = 0.039$	$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(16) = 0.028$	$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(25) = 0.039$
$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(8) = 0.038$	$\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(17) = 0.042$	

Since $\underline{\mathbf{W}} \cdot \underline{\mathbf{A}}(i)$ is significantly larger than any other entry, it is a good guess that $k_0 = 18 =$ **`s`**.

In a similar way, we get

$k_1 = 2 =$ **`c`**	$k_5 = 1 =$ **`b`**
$k_2 = 17 =$ **`r`**	$k_6 = 11 =$ **`l`**
$k_3 = 0 =$ **`a`**	$k_7 = 4 =$ **`e`**
$k_4 = 12 =$ **`m`**	$k_8 = 3 =$ **`d`**

The key is **`scrambled`**.