Mathematical Cryptography

MATH 490

Start

August 26, 2024

Author

Paul Beggs
BeggsPA@Hendrix.edu

Instructor

Prof. Allie Ray, Ph.D.

End

December 2, 2024

Exercise 5.10

Encrypt each of the following Vigenère plaintexts using the given keyword and the Vigenère tableau (Table 5.1 in book).

(a) Keyword: `hamlet`

Plaintext: `To be, or not to be, that is the question.`

Solution.

(a) Hamlet is made up of 6 letters, so we repeat the keyword to match the length of the plaintext:

`tobeor | nottobe | thatis | theque | stion`

Then, find the row that starts with the key-letter. For instance, since `hamlet` starts with `h`, we go down to the 8^{th} row in Table 5.1. Then, we go right until we reach the column of our plaintext. So, for `t`, that would be the 20^{th} column.

See the table below for the full encryption. Note that $\mathcal{P}, \mathcal{K}$, and $\mathcal{C}$ indicate the plaintext, keyword, and ciphertext, respectively:

$\mathcal{P}$	`tobeor`	`nottobe`	`thatis`	`theque`	`stion`
$\mathcal{K}$	`hamlet`	`hamlet`	`hamlet`	`hamlet`	`hamlet`
$\mathcal{C}$	`aonpsk`	`uofesu`	`lttlxb`	`zttpun`	`lsftsg`

Exercise 5.11

Decrypt each of the following Vigenère ciphertexts using the given keyword and the Vigenère tableau (Table 5.1 in book).

(a) Keyword: `condiment`

Ciphertext: `r s g h z b m c x t d v f s q h n i g q x r n b m p d n s q s m b t r k u`

Solution.

(a) Reversing the method from Exercise 5.10, we have:

$\mathcal{C}$	rsghzbmcx	tdvfsqhni	gpxrnbmpd	nsqsmbtrk	u
$\mathcal{K}$	condiment	condiment	condiment	condiment	c
$\mathcal{P}$	peterpipe	rpickedap	eckofpick	ledpepper	s

we find the decrypted ciphertext to be:

peter piper picked a peck of pickled peppers

Exercise 5.13

Let

$s =$ "I am the very model of a modern major general."
$t =$ "I have information vegetable, animal, and mineral."

(a) Make frequency tables for s and t.

(b) Compute IndCo(s) and IndCo(t).

	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
Freq s	4	0	0	2	6	1	1	1	1	1	0	2	4	2	4	0	0	4	0	1	0	1	0	0	1	0
Freq t	8	1	0	1	4	1	1	1	5	0	0	3	3	5	2	0	0	2	0	2	0	2	0	0	0	0

Table 5.1: Frequency Distribution for s and t

Solution.

(a) The frequency table for s and t is shown in Table 5.1.

(b) We find the index of coincidence for s to be:

$$\begin{aligned}\text{IndCo}(s) &= \frac{1}{n(n-1)}\sum_{i=0}^{25} F_i(F_i-1)\\ &= \frac{1}{36(35)}(4\cdot 3 + 2\cdot 1 + 6\cdot 5 + \cdots + 0\cdot 0)\\ &= \frac{84}{1260}\\ &\approx 0.0667.\end{aligned}$$

Then, for t, we have:

$$\begin{aligned}\text{IndCo}(t) &= \frac{1}{n(n-1)}\sum_{i=0}^{25} F_i(F_i-1)\\ &= \frac{1}{41(40)}(8\cdot 7 + 4\cdot 3 + \cdots + 0\cdot 0)\\ &= \frac{128}{1640}\\ &\approx 0.0780.\end{aligned}$$

Exercise 5.15

(a) One of the following two strings was encrypted using a simple substitution cipher, while the other is a random string of letters. the index of coincidence of each string and use the results to guess which is which.

$$s_1 = \texttt{RCZBWBFHSLPSCPILHBGZJTGBIBJGLYIJIBFHCQQFZBYFP},$$
$$s_2 = \texttt{KHQWGIZMGKPOYRKHUITDUXLXCWZOTWPAHFOHMGFEVUEJJ}.$$

	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
Freq s_1	0	7	3	0	0	4	3	3	4	3	0	3	0	0	0	3	2	1	2	1	0	0	1	0	2	3
Freq s_2	1	0	1	1	2	2	3	4	2	2	3	1	2	0	3	2	1	1	0	2	3	1	3	2	1	2

Table 5.2: Frequency Distribution for s_1 and s_2

Solution.

(a) See the table below for the frequency distribution of s_1 and s_2 in Table 5.2. We find the index of coincidence for s_1 to be:

$$\begin{aligned}\text{IndCo}(s_1) &= \frac{114}{45(44)}\\ &\approx 0.0576.\end{aligned}$$

Then, for s_2, we have:

$$\begin{aligned}\text{IndCo}(s_2) &= \frac{60}{45(44)}\\ &\approx 0.0303.\end{aligned}$$

Since the index of coincidence for s_1 is higher than that of s_2, we can guess that s_1 was encrypted using a simple substitution cipher.

Exercise 5.17

We applied a Kasiski test to the Vigenère ciphertext listed below and found that the key length is probably 5. Use Excel to find the plaintext and the key.

```
togmg  gbymk  kcqiv  dmlxk  kbyif  vcuek  cuuis  vvxqs  pwwej  koqgg
phumt  whlsf  yovww  knhhm  rcqfq  vvhkw  psued  ugrsf  ctwij  khvfa
thkef  fwptj  ggviv  cgdra  pgwvm  osqxg  hkdvt  whuev  kcwyj  psgsn
gfwsl  jsfse  ooqhw  tofsh  aciin  gfbif  gabgj  adwsy  topml  ecqzw
asgvs  fwrqs  fsfvq  rhdrs  nmvmk  cbhrv  kblxk  gzi
```

Solution. From the problem, I knew the key size was 5. Thus, I used the IndCo keysize 5 sheet. This is my order of operations:

1. Change the Ciphertext: Capitalize letters and remove spaces.
2. Paste this into A1.
3. Confirm the IndCo value is appropriate (IndCo = 0.064, so it is).
4. Copy the concatenated string from C122 into first the first Excel sheet we did.
5. Find the lowest χ^2 value, so I know what value to subtract 26 by.
6. Go to rot13 and paste the string in, then rotate it by 26 minus the value we found in the previous step.
7. Record this string in the Excel sheet.
8. Repeat step 4-7 for the rest of the concatenated strings.

We end up with 5 decrypted strings. Now, we need to read them from top to bottom. To concatenate the strings, I used the following python code in Listing 5.1. Once I ran the code, I got the following plaintext:

Radio, envisioned by its inventor as a great humanitarian contribution, was seized upon by the generals soon after its birth and impressed as an instrument of war. But radio turned over to the commander a copy of every enemy cryptogram it conveyed. Radio made cryptanalysis an end in itself.[1]

To find the key, I used the following python code in Listing 5.2. The key was found to be `CODES`.

[1] A Google search with the decrypted words led me to the plaintext in proper grammatical form without capitalization and proper spacing.

Listing 5.1: Python Code to Concatenate Strings

```
cipher_text = [
    "REIBITATNINUWIPTNSAIRDEANMFUINEHMRYEEYRCYDDPLAIE",
    "ANOYNOGHIATTAZOHESFTTISSSEWTOEREAAORMPAOEIETYNNL",
    "DVNIVRRUTNRISENEROTSHMSATNARTDTCNCFYYTMNDOCASEIF",
    "IIETEAEMACIOSDBGAOEBAPENRTRAUOOODOEECOIVRMRNINT",
    "OSDSNSAAROBNEUYELNRINRDIUOBDRVTMEPVNRGTEAAYASDS"
]

# Number of rows and columns
num_rows = len(cipher_text)
num_cols = min(len(row) for row in cipher_text)

# Read the text column by column
decoded_text = ""
for col in range(num_cols):
    for row in range(num_rows):
        decoded_text += cipher_text[row][col]

# Print the result
print("Decoded text (column-by-column):")
print(decoded_text)
```

Listing 5.2: Python Code to Find Key

```
# Plaintext and Ciphertext have been omitted because they would not fit in the page.
plaintext = "..."
ciphertext = "..."

# Known key length
key_length = 5

# Helper function to convert letters to alphabetical index (A=0, B=1, ..., Z=25)
def letter_to_index(letter):
    return ord(letter) - ord('A')

# Helper function to convert index back to a letter
def index_to_letter(index):
    return chr(index + ord('A'))

# Calculate the key by determining the shift for each character in the key
key = []
for i in range(key_length):
    # Compute the shift for each position in the key
    shift = (letter_to_index(ciphertext[i]) - letter_to_index(plaintext[i])) % 26
    key.append(index_to_letter(shift))

# Join the key characters into a string
key_word = ''.join(key)
print("Derived key:", key_word)
```