SUBSTITUTION CIPHERS AND MARKOV CHAINS

ALAN LEE, EULER CIRCLE FALL 2019: CRYTOPGRAPHY

1. Introduction

Throughout the course of history, many civilizations have used different methods of preventing outsiders and third parties from accessing confidential information. When the issue at hand was secure messaging systems, one effective way was to encrypt messages using a specific protocol, such as replacing each letter with a number. This way, only the involved parties could accurately deduce the information. While many ciphers have become outdated due to the numerous amounts of methods devised to break them, these classic ciphers continue to be involved in challenging and intriguing open questions. In this paper, we will be covering the correlation between Markov Chains and a completely computer-automated solution to substitution ciphers.

2. Definitions

A **bigram** is a string consisting of two letters in any fixed alphabet Σ. This alphabet may consist of letters, numbers, etc. For instance, "AA", "DC", "X1" are all examples of bigrams, while "ABC", and "C" are not. For the sake of clarity, the bigrams in this paper will be concerning letters from the English 26-letter alphabet.

The **plaintext** is the original text that one party wants to send to another party.

The **ciphertext** is the encrypted plaintext that can only be read and understood if decrypted properly.

A **Markov Chain** is a model that indicates the probability of moving to another state given the current state. A very simple Markov Chain is in **Figure 1**, with each percentage representing the likelihood of that even occurring given that the first even has just occurred. For instance, if it rained today, the chance that it will be cloudy tomorrow is 30%.

A **substitution cipher** is a primitive method of encryption. The way a substitution cipher works is by replacing all "A"s of a plaintext by a randomly selected letter from the alphabet. Next, the "B"s are replaced by a letter different from the one already designated for the "A"s. This continues until all "Z"s are replaced by the only remaining alphabet letter. The process in which each letter is converted into a different letter in this process can also be represented by a single permutation of the 26 letters. The nth letter in the 26-letter string would tell us what letter to change each nth letter in the alphabet with in our plaintext message. We define the **encryption key** as the string of 26 letters we have

Date: December 8, 2019.

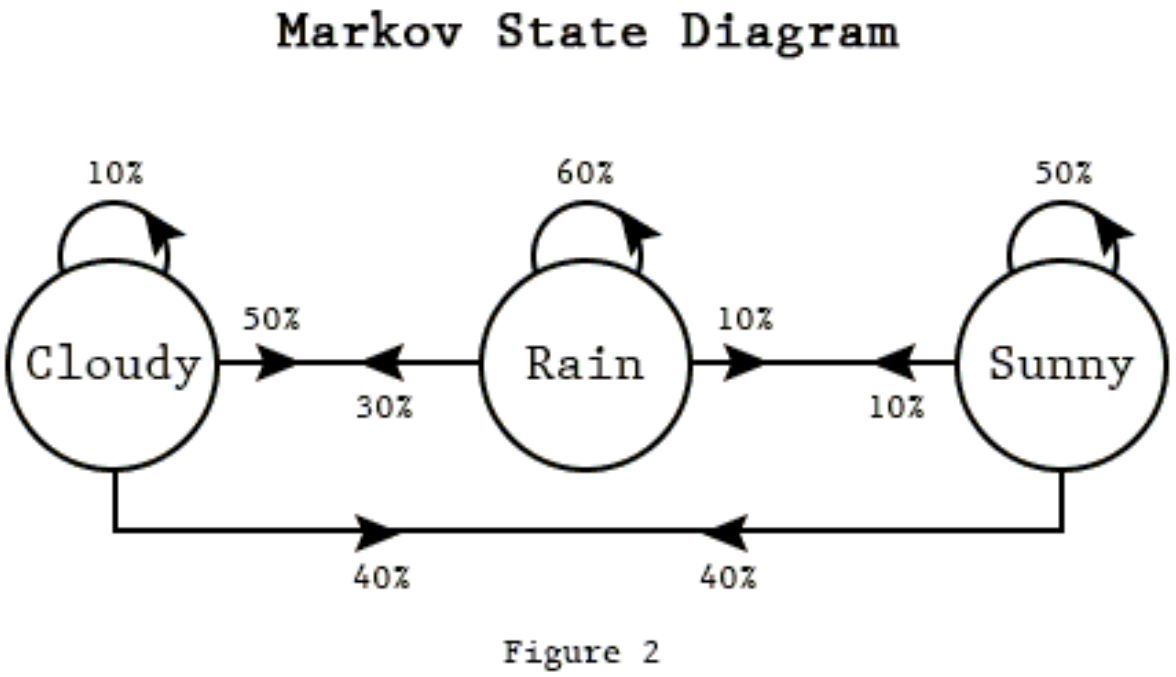

Figure 1. A Simple Markov Chain. [3]

designated for each letter from A through Z.

Example: *If the encryption key is QWERTYUIOPASDFGHJKLZXCVBNM, and we wanted to encrypt the phrase "Have Fun", we would replace "H" with "I"(the 8th letter), "A" with "Q"(the first letter), and so on. Our final result would be* ***IQCTYXF****, with spaces omitted to maximize security.*

3. Problem Statement

Substitution ciphers can be broken through multiple methods, but one purely automatic way is by running a computer program that involves the use of matrices and Markov Chains. Markov Chains, due to their perfect modeling of a transition from one state to another, can also be used to effectively evaluate the likelihood of one letter in the alphabet being followed by another. As frequency models, Markov Chains allow for the calculation of a frequency index, which can in turn analyze how plausible any given encryption key is. In order to increase the efficiency of the decoding process, we will be exploring this Markov Chain method.

4. Solution

As an example of this method in effect throughout this paper, we will use the encryption key "VIQZQKTNGXZQSAOFUQWGXZ" to encrypt a plaintext. Now the objective is to find the key and be able to decrypt the text.

	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
A	8	80	143	248	0.25	52	115	21	184	1	55	250	144	879	2	97	1	458	503	500	48	111	84	3	137	12
B	73	7	1	1	289	1	0.25	2	33	0.25	0.25	71	0.25	1	75	0.25	0.25	51	2	6	67	0.25	2	0.25	30	0.25
C	141	0.25	13	1	153	0.25	0.25	163	30	1	121	36	0.25	1	209	2	0.25	37	1	59	31	0.25	1	0.25	3	0.25
D	229	77	57	78	199	57	56	188	279	8	7	56	92	102	198	34	0.25	67	176	273	41	8	99	0.25	40	0.25
E	513	96	197	639	252	143	97	222	149	7	28	328	193	457	116	138	7	972	551	438	17	90	289	21	132	4
F	103	9	16	2	89	50	6	55	93	1	0.25	19	19	3	187	8	1	88	20	127	63	0.25	6	0.25	13	0.25
G	153	17	12	15	174	16	17	222	86	1	0.25	64	13	13	162	5	1	82	31	66	24	2	20	0.25	11	0.25
H	650	12	11	6	1744	13	5	41	554	1	0.25	10	13	10	245	4	1	49	18	165	32	15	0.25	0.25	20	0.25
I	29	10	173	145	125	101	110	10	2	2	46	258	225	893	96	26	0.25	112	458	497	6	64	10	3	1	10
J	3	0.25	0.25	0.25	7	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	6	0.25	0.25	0.25	0.25	0.25	31	0.25	3	0.25	0.25	0.25
K	39	6	6	4	209	11	2	15	80	0.25	0.25	11	7	45	21	1	0.25	1	27	38	6	0.25	11	0.25	10	0.25
L	239	17	9	215	310	45	7	30	272	4	19	331	24	19	217	14	1	11	50	70	37	6	30	0.25	243	0.25
M	318	95	6	3	294	20	6	37	188	1	2	5	18	3	120	47	2	5	56	54	39	1	18	0.25	26	0.25
N	165	26	94	687	310	27	565	86	127	5	46	43	25	35	260	15	1	9	122	553	24	12	60	9	61	3
O	93	60	37	82	18	261	29	93	59	1	85	170	259	420	197	70	2	372	138	306	630	75	245	0.25	24	4
P	100	4	4	1	128	4	1	0.25	59	0.25	1	63	6	1	89	57	0.25	66	26	39	39	0.25	10	0.25	10	0.25
Q	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	32	0.25	0.25	0.25	0.25	0.25
R	260	34	46	69	579	44	40	113	299	9	27	91	66	56	290	24	4	49	182	185	36	14	0.25	0.25	165	0.25
S	265	57	71	16	338	64	24	409	195	7	28	67	141	53	232	85	6	12	270	444	0.25	9	103	0.25	18	0.25
T	289	55	63	44	317	47	33	1482	305	4	11	121	68	29	624	16	4	149	158	338	69	3	132	0.25	63	0.25
U	26	21	65	38	36	8	92	8	32	2	10	186	31	136	2	0.25	1	174	184	238	0.25	7	13	0.25	2	0.25
V	6	0.25	0.25	0.25	331	0.25	0.25	0.25	40	0.25	0.25	0.25	0.25	0.25	14	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	4	0.25
W	369	3	3	3	161	6	0.25	207	195	0.25	1	8	12	67	141	3	0.25	12	35	22	1	2	9	0.25	9	0.25
X	4	0.25	6	0.25	2	0.25	0.25	0.25	11	0.25	0.25	0.25	0.25	0.25	0.25	15	0.25	0.25	0.25	7	1	0.25	0.25	0.25	0.25	0.25
Y	153	29	22	20	98	30	21	46	67	4	2	26	37	10	204	18	5	9	86	89	6	2	52	0.25	14	0.25
Z	1	0.25	0.25	0.25	20	0.25	0.25	0.25	3	0.25	0.25	1	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	2	3

Figure 2. A Matrix With Bigram Frequencies.

We will look at letter frequencies similar to the method of frequency analysis, but now we will look at 2-letter frequencies; in other words, given a letter, we will look at the possible frequencies of the following letter.

We can find these frequencies by investigating a sufficiently long English book, such as *The Big Blue Soldier* which in brought from Project Gutenberg. For the sake of simplicity in this paper, we will just be observing only the first approximately 50000 bigrams. Taking every bigram and counting its frequency by inputting each entry into a 26 by 26 matrix gives us the desired bigram frequencies. As shown in **Figure 2**, the letters in the rows (left) correspond to the first letter in the bigram, and the letters on the columns (top) correspond to the second letter in the bigram. Common bigrams such as "TH" appear many more times (1482 times) than uncommon bigrams such as "BH" (2 times). Additionally, bigrams that never appeared, such as "JB" were replaced with "0.25" instead of "0" to prevent a product of 0. The reason for this will be clarified in more detail later in the paper.

Now, we will choose a random permutation of the letters from A through Z. Our encryption key will now be this string of 26 letters, so that in the plaintext, A is replaced by the first letter in the permutation, B is replaced by the second, and so on. Since we will be *decrypting* an encrypted ciphertext, we will be reversing the process mentioned above.

Example: *If the encryption key was "ASDFGHJKLQWERTYUIOPZXCVBNM" and the ciphertext was "TDYFIG", we would see that "T" corresponded to the 14th letter in our permutation. Therefore, we wuld replace "T" with the 14th letter in the alphabet, "N". For "D", we would notice it was the third letter in the permutation. Since the third letter in the alphabet is "C", "D" would be replaced by it. We would do the same for "Y", "F", "I" and "G" to obtain the decrypted text "NCODQE".*

Once we decrypt our ciphertext message with this randomly selected permutation, we can take all the bigrams that occur in our decrypted message and find their relative frequencies

in the 26 by 26 matrix. Once we find all of the frequences, we multiply them together to obtain a number n. The value of n is quite significant because it acts an index that shows how accurate (i.e. how close to the original encryption key) a given permutation is. Larger n indicate that there were more common bigrams in the decrypted text, while smaller n indicate that the bigrams in the decrypted text were less common.

Example: *Taking out ciphertext that we wish to decode, "VIQZQKTNGXZQSAOFUQWGXZ", we randomly choose our permutation to be "ABCDEFGHIJKLMNOPQRSTUVWXYZ". By applying our permutation to our ciphertext, we obtain "VIQZQKTNGXZQSAOFUQWGXZ". Now we will multiply the frequncies of all the appearing bigrams in this text together in the following fashion: "VI" has frequency 40, "IQ" has frequency 0.25, "QZ" has frequency 0.25, "ZQ" has freuquency 0.25, "QK" has frequency 0.25, and so on. Now we multiply all of these frequencies to get a frequency index of our given permutation, which we call "n". For the above permutation, our* $n \approx 12936805.96$.

Our next step is to analyze the value we have found and how likely our combination is really the plaintext message. The smaller the value n is, the less likely our combination is. If our value of n does not indicate a plausible combination, we will randomly select two different spaces in our 26-letter permutation and switch them around.

After doing so, we will start again with the process of decrypting using our key. We proceed to calculate a n_{new} value for our new decryption. If this new value is greater than our original n (which we will call n_{old} for the sake of clarity), we will obtain the new permutation and repeat our process.

Example: *we will randomly switch the letters A and X to get "XBCDEFGHIJKLMNOPQRSTUVWAYZ" as our new permutation. Decoding the ciphertext with this key, we get "VIQZQKTNGAZQSXOFUQWGAZ". We now calculate* n_{new} *by starting the process again from the beginning: "VI" has frequency 40, "IQ" has frequency 0.25, "QZ" has frequency 0.25, "ZQ" has frequency 0.25, "QK" has frequency 0.25, and so on. Eventually, multiplying these together, our result is* $n_{new} \approx 1316486866617.6$. *Since* $n_{new} > n_{old}$, *we forget our original permutation and adopt the new one, namely "XBCDEFGHIJKLMNOPQRSTUVWAYZ".*

However, if n_{new} is smaller than n_{old}, we would think of trying to abandon our new possibility. But this should not be the case because this leads to a possibility of an optimization loop, where the correct permutation may never be achieved simply due to the fact that one permutation possesses an abnormally high value of n. In this case, we will select n_{new} with probability $\frac{n_{new}}{n_{old}}$. Therefore, the larger n_{new} is, the higher to probability it is for us to switch (Now we see why replacing the 0 frequencies with "0.25" was a good idea. If we kept all the "0"s, not only would most of our values of n be equal to 0, the probability of the switch happening would be impossible to calculate due to a zero in the denominator!).

Example: *Consider the following switch that replaces "O" with "J" and vice versa. Now we obtain the new decoded ciphertext: "VIQZQKTNGAZQSXJFUQWGAZ". Calculating n using the same method, we see that* $n \approx 1261002745.8$. *Here,* $n_{new} < n_{old}$. *Now, we will forget our old permutation and proceed with the new one with probability* $\frac{n_{new}}{n_{old}}$, *which in this*

```
 100 ER ENOHDLAE OHDLO UOZEOUNORU O UOZEO HD OITO HEOQSET IUROFHE HENO ITORUZAEN
 200 ES ELOHRNDE OHRNO UOVEOULOSU O UOVEO HR OITO HEOQAET IUSOPHE HELO ITOSUVDEL
 300 ES ELOHANDE OHANO UOVEOULOSU O UOVEO HA OITO HEOQRET IUSOFHE HELO ITOSUVDEL
 400 ES ELOHINME OHINO UOVEOULOSU O UOVEO HI OATO HEOQRET AUSOWHE HELO ATOSUVMEL
 500 ES ELOHINME OHINO UODEOULOSU O UODEO HI OATO HEOQRET AUSOWHE HELO ATOSUDMEL
 600 ES ELOHINME OHINO UODEOULOSU O UODEO HI OATO HEOQRET AUSOWHE HELO ATOSUDMEL
 900 ES ELOHANME OHANO UODEOULOSU O UODEO HA OITO HEOQRET IUSOWHE HELO ITOSUDMEL
1000 IS ILOHANMI OHANO RODIORLOSR O RODIO HA OETO HIOQUIT ERSOWHI HILO ETOSRDMIL
1100 ISTILOHANMITOHANOT ODIO LOS TOT ODIOTHATOEROTHIOQUIRTE SOWHITHILOTEROS DMIL
1200 ISTILOHANMITOHANOT ODIO LOS TOT ODIOTHATOEROTHIOQUIRTE SOWHITHILOTEROS DMIL
1300 ISTILOHARMITOHAROT ODIO LOS TOT ODIOTHATOENOTHIOQUINTE SOWHITHILOTENOS DMIL
1400 ISTILOHAMRITOHAMOT OFIO LOS TOT OFIOTHATOENOTHIOQUINTE SOWHITHILOTENOS FRIL
1600 ESTEL HAMRET HAM TO CE OL SOT TO CE THAT IN THE QUENTIOS WHETHEL TIN SOCREL
1700 ESTEL HAMRET HAM TO BE OL SOT TO BE THAT IN THE QUENTIOS WHETHEL TIN SOBREL
1800 ESTER HAMLET HAM TO BE OR SOT TO BE THAT IN THE QUENTIOS WHETHER TIN SOBLER
1900 ENTER HAMLET HAM TO BE OR NOT TO BE THAT IS THE QUESTION WHETHER TIS NOBLER
2000 ENTER HAMLET HAM TO BE OR NOT TO BE THAT IS THE QUESTION WHETHER TIS NOBLER
```

Figure 3. An Iteration of the Markov Chain Method on *Hamlet*

case is $\frac{1}{1044}$. *In this example, we end up not switching and retaining the old one, namely "XBCDEFGHIJKLMNOPQRSTUVWAYZ".*

After switching or retaining the original permutation, we will continue switching the original permutation around, one 2-letter-pair at a time. Each time, we can repeat the process by multiplying all bigram frequencies, then analyzing and comparing n_{old} to n_{new}. Gradually, we will see an increase in the value of n and will also start to see a comprehensible message. Although the program does not physically come to a halt when a sufficiently high value of n is reached, the program eventually reaches a permutation that has such a high n that it is near impossible for the program to switch any pair of letters and obtain a better index n. By having the program print out the current state of its decrypted message every 50 or 100 steps, one can eventually see a near-constant decryption text. By filling in necessary errors such as switching a "Q" with an "X", the full message can be decoded. An example by Persi Diaconis' students Marc Coram and Phil Beineke involving Shakespeare's play *Hamlet* is shown in **Figure 3**, with the iteration number on the left [2]. As displayed, after just 1900 iterations, the message has been fully decrypted and matches with the original text verbatim.

Example: *In our case, the plaintext of our ciphertext that is decoded with the correct key "QWERTYUIOPASDFGHJKLZXCVBNM" is "WHATAREYOUTALKINGABOUT". This text has a very high frequency, namely* $n \approx 2.82905 * 10^{50}$. *Due to this high frequency, any new possibilities would have a very slim chance of replacing of this primary permutation.*

5. Summary

Although this method seems to have many steps and seems very complicated, all of these steps may be very easily processed using a computer system. For instance, the probability of switching from one permutation to another could be replicated by a random integer function.

By running this program on a computer, we can quickly obtain the original message without having to do any 1-letter frequency analysis, which is often done manually with multiple guesses.

However, with the advent of new words and vocabulary such as text message slang, this method of finding and determining adequate n becomes ever-so-challenging. By referencing a classic book that does not include vocabulary such as "ttyl" or "brb", a computer system may easily skip over a very plausible combination due to the fact that it does not recognize more modern forms of communication. On the other hand, by referencing a text-message conversation between two people from the 21st century in order to derive bigram frequencies, it may be hard to determine a document encoded 500 years ago in Old English. Our versatile and volatile English language leaves cryptography a fast-moving and fast-changing subject, open to anyone for new interpretations and solutions.

References

[1] Chen, Jian, and Jeffrey S. Rosental. *Decrypting Classical Cipher Text Using Markov Chain Monte Carlo.* Springer, 1 Apr. 2011, probability.ca/jeff/ftpdir/decipherartpub.pdf.

[2] Diaconis, Persi. "The Markov Chain Monte Carlo Revolution."

[3] "A Markov Chain." *Tech-Effigy*, 8 Jan. 2015, techeffigytutorials.blogspot.com/2015/01/markov-chains-explained.html.

[4] Rubinstein-Salzedo, Simon. "19. Markov Chains." *Cryptography*, by Simon Rubinstein-Salzedo, Springer, 2018, pp.221-230.