Lab Manual-2: Checkpoint 2

Amit Hassan Joy
Registration Number: 2020831026
Date: November 8, 2025

Introduction

This lab report presents a substitution cipher attack using frequency analysis. Checkpoint-2 focuses on breaking monoalphabetic ciphers by analyzing character frequencies and iteratively building a substitution key. Unlike Caesar ciphers, which use simple shifts, substitution ciphers employ arbitrary one-to-one mappings. Despite this, they are vulnerable to frequency analysis when the ciphertext is long enough to reveal language patterns. The report summarizes the methodology and results for two encrypted texts.

Objective

The primary objectives of this checkpoint are to:

- Understand **monoalphabetic** substitution cipher mechanics
- Implement frequency analysis attacks
- Build substitution keys through linguistic pattern matching
- Successfully decrypt two substitution ciphers using partial key recovery
- Demonstrate the vulnerability of substitution ciphers to cryptanalysis

Background: Substitution Cipher

A monoalphabetic substitution cipher replaces each letter of the plaintext with a fixed corresponding letter in the ciphertext. Unlike the Caesar cipher with only 26 possible keys, a substitution cipher has 26! (factorial) possible permutations, making brute-force enumeration infeasible.

Cipher Characteristics

- Deterministic mapping: Each letter always maps to the same cipher letter
- Monoalphabetic: The same cipher letter represents the same plaintext letter throughout
- Large keyspace: 26! ≈ 4.03 × 10^26 possible substitution keys
- Preserves structure: Word lengths and letter patterns are retained in ciphertext

Vulnerability to Frequency Analysis
Despite the large keyspace, substitution ciphers are vulnerable to frequency analysis because:

- Natural language has predictable letter frequency patterns
- These patterns are preserved in the ciphertext
- Common English letters (E, T, A, O, I, N) have measurable frequencies
- High-frequency cipher letters likely map to high-frequency English letters
- Linguistic knowledge helps disambiguate partial decryptions

Attack Methodology: Frequency Analysis

The attack methodology consists of the following steps:

Step 1: Calculate Character Frequencies

- Count the occurrence of each character in the ciphertext.
- Calculate the percentage of each character based on total alphabetic characters.
- Sort characters by frequency in descending order.

Step 2: Compare with English Frequency Distribution

Compare cipher character frequencies with known English letter frequencies:

Letter	Frequency (%)
E	12.22
T	9.67
A	8.05
O	7.63
I	6.28
…	…

Step 3: Build Initial Key Mapping

- Map high-frequency cipher letters to high-frequency English letters.
- Sort by frequency: highest cipher letter → highest English letter.
- Create an initial substitution key from frequency-sorted lists.

Step 4: Decrypt and Analyze Partial Results

- Apply initial key mapping to decrypt ciphertext.
- Look for recognizable English patterns (words, digraphs, trigraphs).
- Identify plaintext from partial decryption.

Step 5: Iteratively Refine the Key

- Adjust character mappings based on recognized English words and patterns.
- Repeat until complete or nearly complete plaintext is recovered.

Implementation Details

```
import collections

def calculate_char_frequency(text):
    text = text.lower()
    char_counts = collections.Counter(c for c in text if c.isalpha())
    total_alpha_chars = sum(char_counts.values())
    frequencies = [
        (char, count, (count / total_alpha_chars) * 100)
        for char, count in char_counts.items()
    ]
    frequencies.sort(key=lambda x: x[2], reverse=True)
    return frequencies
def decrypt(ciphertext, key):
    plaintext = []
    for char in ciphertext.lower():
        if char.isalpha():
            plaintext.append(key.get(char, '_'))
        else:
            plaintext.append(char)
```

```
    return "".join(plaintext)
```

Key Components

- `calculate_char_frequency(text)`: Analyzes character distribution in ciphertext.
- `decrypt(ciphertext, key)`: Applies substitution key to decrypt text.
- `key_map` dictionary: Stores cipher→plaintext character mappings.
- Iterative refinement: Gradually build complete key through pattern recognition.

Cipher 1: Attack Results

Ciphertext

```
af p xpkcaqvnpk pfg, af ipqe qpri, gauuikifc tpw, ceiri udvk tiki afgarxifrphni cd eao--wvmd popkwn, hiqpvri du ear jvaql vfgikrcpfgafm du cei xkafqaxnir du xrwqedearcdkw pfg du ear aopmafpcasi xkdhafmr afcd fit pkipr. ac tpr qdoudkcafm cd lfdt cepc au pfwceafm epxxifig cd ringdf eaorinu hiudki cei opceiopcaqr du cei uaing qdvng hi qdoxnicinw tdklig dvc--pfg edt rndtnw ac xkdqiigig, pfg edt odvfcpafdvr cei dhrcpqnir--ceiki tdvng pc niprc kiopaf dfi mddg oafg cepc tdvng qdfcafvi cei kiripkqe
```

Frequency Analysis

Cipher 1 character frequencies (top 10):

I: 11.33%

D: 8.87%

C: 8.13%

P: 7.88%

U: 7.88%

F: 7.88%

A: 7.64%

E: 5.42%

R: 5.67%

K: 4.68%

Key Mappings Built

Frequency-based initial mapping:

i → e, d → t, c → a, p → o

Iterative refinement:

g → d, q → c, r → s, k → r, t → w, w → y

Final Key Map for Cipher 1

Cipher	Plaintext
a	i
c	t
d	o
e	h
f	n
g	d
h	b
i	e
j	q
k	r
l	k
m	g
n	l
o	m

Cipher	Plaintext
p	a
q	c
r	s
s	j
t	w
u	f
v	u
w	y
x	p

Decrypted Plaintext

in a particular and, in each case, different way, these four were indispensable to him--yugo amaryl, because of his quick understanding of the principles of psychohistory and of his imaginatije probings into new areas. it was comforting to know that if anything happened to seldon himself before the mathematics of the field could be completely worked out--and how slowly it proceeded, and how mountainous the obstacles--there would at least remain one good mind that would continue the research

Cipher 2: Attack Results

Ciphertext (Excerpt)

"aceah toz puvg vcdl omj puvg yudqecov, omj loj auum klu thmjuv hs klu zlcvu shv zcbkg guovz, upuv zcmdu lcz vuwovroaeu jczoyyuovomdu omj qmubyudkuj vukqvm. klu vcdluz lu loj avhqnlk aodr svhw lcz kvopuez loj mht audhwu o ehdoe eunumj, omj ck toz yhyqeoveg auecupuj, tlokupuv klu hej sher wcnlk zog, klok klu lcee ok aon umj toz sqee hs kqmmuez zkqssuj tckl kvuozqvu. omj cs klok toz mhk umhqnl shv sowu, kluvu toz

oezh lcz yvhehmnuj pcnhqv kh wovpue ok. kcwu thvu hm, aqk ck zuuwuj kh lopu eckkeu ussudk hm wv. aonncmz. ok mcmukg lu toz wqdl klu zowu oz ok scskg. ok mcmukg-mcmu klug aunom kh doee lcw tuee-yvuzuvpuj; aqk qmdlomnuj thqej lopu auum muovuv klu wovr. kluvu tuvu zhwu klok zlhhr klucv luojz omj klhqnlk klcz toz khh wqdl hs o nhhj klcmn; ck zuuwuj qmsocv klokomghmu zlhqej yhzzuzz (oyyovumkeg) yuvyukqoe ghqkl oz tuee oz (vuyqkujeg) cmubloqzkcaeu tuoekl. ck tcee lopu kh au yocj shv, klug zocj. ck czm'k mokqvoe, omj kvhqaeu tcee dhwu hs ck! aqk zh sov kvhqaeu loj mhk dhwu; omj oz wv. aonncmz toz numuvhqz tckl lcz whmug, whzk yuhyeu tuvu tceecmn kh shvncpu lcw lcz hjjckcuz omj lcz nhhj shvkqmu. lu vuwocmuj hm pczckcmn kuvwz tckl lcz vueokcpuz (ubduyk, hs dhqvzu, klu zodrpceeu-aonncmzuz), omj lu loj womg juphkuj ojwcvuvz owhmn klu lhaackz hs yhhv omj qmcwyhvkomk sowcecuz. aqk lu loj mh dehzu svcumjz, qmkce zhwu hs lcz ghqmnuv dhqzcmz aunom kh nvht qy. klu uejuzk hs kluzu, omj aceah'z sophqvcku, toz ghqmn svhjh aonncmz. tlum aceah toz mcmukg-mcmu lu ojhykuj svhjh oz lcz lucv, omj avhqnlk lcw kh ecpu ok aon umj; omj klu lhyuz hs klu zodrpceeu- aonncmzuz tuvu scmoeeg jozluj. aceah omj svhjh loyyumuj kh lopu klu zowu acvkljog, zuykuwauv 22mj. ghq loj aukkuv dhwu omj ecpu luvu, svhjh wg eoj, zocj aceah hmu jog; omj klum tu dom dueuavoku hqv acvkljog-yovkcuz dhwshvkoaeg khnukluv. ok klok kcwu svhjh toz zkcee cm lcz ktuumz, oz klu lhaackz doeeuj klu cvvuzyhmzcaeu ktumkcuz auktuum dlcejlhhj omj dhwcmn hs onu ok klcvkg-klvuu"

Frequency Analysis

Cipher 2 character frequencies (top 10):

U: 12.8%

K: 8.53%

O: 8.47%

H: 7.3%

C: 6.59%

Z: 6.14%

M: 6.14%

L: 5.75%

V: 5.49%

J: 4.78%

Key Mappings Built

Frequency-based initial mapping:

u → e, k → t, l → h, h → a

Iterative refinement through pattern recognition:

o → a, z → s, m → n, j → d, v → r, c → i, d → c

Final Key Map for Cipher 2

Cipher	Plaintext
a	b
b	x
c	i
d	c
e	l
g	y
h	o
i	j
j	d
k	t
l	h
m	n

Cipher	Plaintext
n	g
o	a
p	v
q	u
r	k
s	f
t	w
u	e
v	r
w	m
y	p
z	s

Decrypted Plaintext

Deciphered Text : bilbo was very rich and very peculiar, and had been the wonder of the shire for sixty years, ever since his remarkable disappearance and unexpected return. the riches he had brought back from his travels had now become a local legend, and it was popularly believed, whatever the old folk might say, that the hill at bag end was full of tunnels stuffed with treasure. and if that was not enough for fame, there was also his prolonged vigour to marvel at. time wore on, but it seemed to have little effect on mr. baggins. at ninety he was much the same as at fifty. at ninety-nine they began to call him well-preserved; but unchanged would have been nearer the mark. there were some that shook their heads and thought this was too much of a good thing; it seemed unfair that anyone should possess (apparently) perpetual youth as well as (reputedly) inexhaustible wealth. it will

have to be paid for, they said. it isn't natural, and trouble will come of it! but so far trouble had not come; and as mr. baggins was generous with jis money, most people were willing to forgive him jis oddities and jis good fortune. he remained on visiting terms with jis relatives (except, of course, the sackville- bagginses), and he had many devoted admirers among the hobbits of poor and unimportant families. but he had no close friends, until some of jis younger cousins began to grow up. the eldest of these, and bilbo's favourite, was young frodo baggins. when bilbo was ninety-nine he adopted frodo as jis heir, and brought jim to live at bag end; and the hopes of the sackville- bagginses were finally dashed. bilbo and frodo happened to have the same birthday, september 22nd. you had better come and live here, frodo my lad, said bilbo one day; and then we can celebrate our birthday-parties comfortably together. at that time frodo was still in jis tweens, as the hobbits called the irresponsible twenties between childhood and coming of age at thirty-three

Observations

Cipher 2 is easier to break

- Contains many short, repeated words (omj, klu, toz, ok, ck) → maps cleanly to "the", "and", "to", "of", "be".
- Recognizable word patterns and endings (e.g., czm'k → n't) facilitate quick pattern matching.
- Clear sentence structure, punctuation, and numbers provide context for validation.
- Frequency ranks closely match English letter frequency.

Cipher 1 is harder

- Longer, less-repetitive tokens and hyphenated compounds obscure common function words.
- Fewer short-word anchors → frequency-based guesses less reliable.
- Requires more manual refinement for accurate decryption.

Conclusion

Cipher 2: Simple frequency analysis + pattern matching sufficient to recover plaintext.

Cipher 1: More complex; requires advanced techniques or extended context.

Substitution ciphers are weak: Language redundancy, frequency preservation, and predictable patterns allow successful attacks with 50-100 characters of ciphertext.

Security Implications and Cryptanalysis

Why Substitution Ciphers Are Vulnerable

Although monoalphabetic substitution ciphers theoretically allow 26! possible keys, they are practically broken because of:

- **Language redundancy:** Natural language exhibits predictable patterns and repetitions that persist even after encryption.
- **Frequency preservation:** Letter frequency distributions in plaintext are largely maintained in the ciphertext.
- **Insufficient diffusion:** Changes in the plaintext do not significantly alter the ciphertext, making patterns detectable.
- **Lack of confusion:** Simple one-to-one letter mapping provides no confusion, leaving the cipher vulnerable to analysis.

Minimum Ciphertext Length

Studies indicate that for English text, roughly 50–100 characters of ciphertext are sufficient for frequency analysis attacks to succeed.

- **Cipher 1:** 495 characters
- **Cipher 2:** 1,949 characters

Both are well above this threshold, and longer texts increase the probability of a successful attack by providing more statistical evidence for pattern recognition.

Thought Process for the Attack

The attack strategy followed a systematic approach:

Initial reconnaissance

- Calculate character frequencies for both ciphertexts.
- Compare them against standards English letter frequencies.
- Identify the most frequently occurring cipher characters.

Hypothesis formation

- Map the highest-frequency cipher letters to the most common English letters (E, T, A, O).

- Create an initial substitution key based on frequency correspondence.
- Apply the partial key to decrypt the ciphertext.

Pattern recognition

- Look for recognizable English words, even if partially decrypted.
- Identify word boundaries, digraphs, trigraphs, and common letter combinations.
- Refine the key mapping based on observed patterns.

Iterative refinement

- Validate and strengthen key mappings as more words become readable.
- Cross-check mappings across multiple words for consistency.
- Continue the process until complete or nearly complete plaintext recovery is achieved.

References

- Wikipedia - Substitution Cipher: https://en.wikipedia.org/wiki/Substitution_cipher
- Wikipedia - Frequency Analysis: https://en.wikipedia.org/wiki/Frequency_analysis
- GeeksforGeeks - Cryptanalysis: https://www.geeksforgeeks.org/cryptanalysis/
- "Breaking Short Textbook RSA Encrypted Messages" - Academic paper on cryptanalysis
- NIST Special Publication 800-175B: Guideline for Using Cryptographic Algorithms