Lab 03 - Vigenere Cipher

Intro

You will solve the polyalphabetic Vigenere cipher this lab. This cipher was labeled as indecipherable for over three hundred years until Kasiski published a method for decrypting the cipher. You will likely use the Kasiski method in your lab as it is easier than calculating the index of coincidence.

Assignment

Decrypt the following message that was encrypted using a Vigenere cipher. Your answer should be the key and the plaintext along with information on how you solved the problem.

a. You will write a **Python** program to help you with this. You will **annotate the code** explaining the steps, printing out the iterations you went through in the annotations. You will **upload the code** as part of your lab submission.

b. You **CANNOT** use some web program and just submit the plaintext and the key. No points will be given for that solution.

WKTZWSGJGRNZIDVNKXNHAHFXDUYFOKEXFDVNKXFQOCMZGGKUJEEMBDQEJBIGNK
XMLGNUNWOHFRRNGOYWHKZHXSZCZZOTBIGUYFYIOXUCHPXPBUEYTJIARMKHOVMZI
OXTKDCKTXLETDZOIQFIXCRFVUSIJZQCKBIGARMOHCNOJOOMZCDSTPXRNYDDHNIFJII
ZTJZNCIZWHKSDWWOMGEEIPHHALPMFELPMJOUEJUIRMYHPKOYVOTNVQATEJQLEJA
WHKVILTKENWAZFNRCIVKLEYBKRSOUDRNUGKUEKNDQETDZFATXZKERQYHCOEZZHK
UCHRZIDVNKXJFEGORLLRCZDSKBJIPKBXHOXBIHWZFMUILZDQGZIZDTKSJIWGSDGOT
POVAEUCDTCFNKOAMYRRCJGOGUVISRUUZFTKEVJAOONWTNFCRSZJGHMOTPVEUG
NSAIFVQYSPMHTNBIZEMPPQPXPOHCZFYDGGJIVTZIZKOYUDOEATZRFRBIGOXTZDBAU
DGOYBTWHGUNSAIFXDNHFZAPRPMHDGOYPAYUZUEJXDWHUVOIEKEDQGZIZIIXFNRFC
BMZIZIJXTXFKHAZJIJTNFHLSZBFHSZIVWMGOCDSSBYHITFSWETEDQGNJNZROUVUOA
OYWHOTBOOHFJIOASNWHKSZLSTPNWROGZQOVSZMUJJXHNUOVWIUOVOCUOAOIIUD
QOAUZUSVBXHAYZZWIZTCDZGSYVAXFCRSZJGHTUVNDLRJOVCUOLXEYUYHSKSQHSZ
IZEEYUJIARMHDNQJIGATEDWSUQKRRZVILTEGJUPKBXHFAMXROVFMDTOPIPAEOZYEX
DJPEGHVLNHVOZHETJPEYBTWHKNJRNCITFHUPNHTNJNDSUVMJOGMVQDZIZBMGZRH
LRBNNWNZXOISCOKENJBKEYUHRUTUVLNCITWHOSOBFOWZBEGSNDGUGGBTNFVWL
GOOLCCITGOKTMLCKQGDYZFSDSCFXKOUTZWOMPORTNFHROTXZFHUPNHTUHJWOZI
ZPOUORHCNPJVEZPBRTUUCHMUPILNZIDVDKDVGEGOYGOZIZRTNFMWHOOBVNUUWH
CGVNHTNFTDRKFVVYHVOEEIBPVEZIZBAXFCDRJCZFAATZWHGUBRARXDOLYFMYEZPJ
UGGODCEGOYPEGTPUEZIZEEYUJIOASZQEXHDHSGOYVKOMGVBKDVXSKUCDTIIVOLK
OBHIYPIHTNBOZEGSZZIRMDQGZPVFCKQORNKXZDRKVIZIRMDQGZPKRSZQJQEGOYR
NKXZLNZFIGTUXDQATEOKEUUCHRYUJR

Remember, the Kasiski method is the easiest way to do this. Find **trigrams (**or if you get lucky repeats of more than 3 characters) that are repeated and then record the spacing between them. You start counting the letter after the first letter in the repeat and stop counting when you come to the first letter in the next repeat. You include that first repeated letter in your count. Then, see what the factors are to make those spacings (for example, 15, 5, 25 spacing between the repeated character sets might mean a key word size of 5). Then, look at the distribution for each letter at that indexed position to see if it matches the patterns found in the English language. Solve for the first character, the second, and so on until the words start to make sense. (See the lecture slides for more help).

For an idea of what to produce, your program might operate something like the following. **Your program does NOT need to work exactly like this**, but the process should be similar. Assume we have the ciphertext: (Note, this is **NOT** the ciphertext that you must decode for the lab. This is merely an example)

zhmnjkhmqjlmaruyoqaykovvyytjhljhmuwrrabfcyeuajpmekffabevqcvwkcwanvranlzovfzvsk beznovfkwmynvtpvjkynyaxhbgzvmwbfcibjaegaewwlmplkhmflrragzrtohaueurlfwievjsiynrtq bfzsbbhgelnffllzseatbfxtprorypbhznoggwivqkfmmbdufweyftbrfnozqkfriauzevgevlwqavsprll rvrvkoursjinggjagumirgogpibforibvfxtprjvfweqfuqgkxovaskasrscobgguritevaenqwrwzqfub uwieaagkhqaykhigsyuvqjvdurffrubjvcwhduedrjuoqodvsagzvrivfjdwjfznisjzcitgenigsbeabev tqzwkolblyebuaegajweedrjyalgzvwqyvuoofuiywhlznbuweioulrsbuwpgzboieagdvsaygegqa ywozfgdeabdztieqtoucseyqxffwbuskiuhkkdwjzrtaeaxhbnkjuzrsjkqyadavwsiozvkvstvcvotle guantfvmgzvsmewegmgazsmrckokhjvwpnljdmrhznavvvfzvyytmawuongzzsbuaegbuskidrt vcwzwztatgenigsbeiygktwqjrgursnagsjfmgbmkhmewjnwgzznogzrtiumedzrvdevbjdozrufut qwmezqgzbtrkjtprjrivfvfwvvfrfzvurgwafrtixwjourlzmmgguobuwkhqayjwmawmezusuhcejpb wlkyeajsztqaykhmewwozlglibfyfnvnlrkmndftbbviaozwrwilxioulgltprjvsvblyivtlyabnzlnlewu mmagimwewtocyvvvmevfijywjsbuwiaqakuoeaaeaneataqodvsagzvrivfjdwjfznisjzcivtceafly eznaeslboeivnxiiknaslmfkkhmesznaqgnnqaswrqpszbtrkjtprjrivfvfwvvfrfzvurgwafrtixwjourlz mmgguobuwkhqayjwmawmezusu

1.) Trigram analysis on the ciphertext

The five most common trigrams are: ['qay', 'buw', 'khm', 'gzv', 'tpr']

2.) Identify the starting indexes of the most common trigrams

The starting indexes of ‘qay’ are: [17, 332, 482, 777, 807, 1057]
The starting indexes of ‘buw’ are: [322, 452, 462, 772, 907, 1052]
The starting indexes of ‘khm’ are: [5, 125, 680, 810, 985]
The starting indexes of ‘gzv’ are: [103, 373, 433, 573, 933]
The starting indexes of ‘tpr’ are: [181, 276, 731, 856, 1011]

3.) Calculate the difference between each starting index

The differences between each index of ‘qay’ are: [315, 150, 295, 30, 250]

The differences between each index of ‘buw’ are: [130, 10, 310, 135, 145]
The differences between each index of ‘khm’ are: [120, 555, 130, 175]
The differences between each index of ‘gzv’ are: [270, 60, 140, 360]
The differences between each index of ‘tpr’ are: [95, 455, 125, 155]

4.) Determine the key length based on shared common factors of the differences

The key length is most likely five because five is a common factor from the differences

5.) Break the ciphertext into X (in this example X is 5) shift-by-n ciphers where X is the length of the key found in step 4. Assuming we had a key length of 5, we would have five shift-by-n ciphers. The first shift-by-n cipher (Cipher0) would contain characters Ciphertext[0, 5, 10, 15, …], second shift-by-n cipher (Cipher1) would contain characters Ciphertext[1, 6, 11, 17, …], so on and so forth.

///////// Cipher 0   /////////
zklykyjrcjfvkvzvzkvkxvcewkrrufjrzgfexrzwfufnfzvvlkjjiprxvfxkcuvwfikkyvfvuuvvjzzebvkyeey
vuizerpivewdztefkkrxjjdivvgfvezkvjzvyuzekvzebkrnfkjzreddfmzjrfrrrjzukjmupyzkwlfrfirilvyyl
uitvfjiuetvvjzzzcyeeiskznwzjrfrrrjzukjmu

///////// Cipher 1   /////////
hhmoothrypfqcrosnwtyhmiglhrtewstselatynimftorelsroiariitfuoaoraruehhudrceosrdncnetoe
geawoynisgesgoetoywidthukaosouvsgsowdnftosgictnetgamhnntdeouebtiwfgtomohwehb
ethoinktawotsianmmovisaoaasrdnceesiilhnnrbtiwfgtomohwe

///////// Cipher 2   /////////
mmaqvjmaemacwavkompnbwbammaouiiqblltppovmwbzivwpvunggbbpwqvsbiewbaqivuu
wdqaiwiiiaqlbadlqowbobzaaqzaiuqbuwabzqvzttammmmkpmazmnbbdwaiiwuggmwoizvzt
ztpvvzwiumbqmzcwaqmzbvmboiupvvblmwcmjbqenqaiwiiazlvkmmaqqtpvvzwiumbqmz

///////// Cipher 3   /////////
nqravhubuebvanfbvyvygbjepfghreybbnzbrbgqberqagqrrrguofvregargtnzuaagqrbhrogvjst
gbzbujrgyfhuuubgyafbecxuhjenrywvvlngegrhnrvvaguurztgyqrsbeggurbrqqrrfvvaxrguaaue
ljaelfnnbzllrbtneaeyeyuaaeogvjsvfnbnnfeqaprrfvvaxrguaau

///////// Cipher 4   /////////
jjuyylwfakewnlzefnjazfawllzalvnfhfsfohgkdyfkuealvsgmgofjqkssgeqqwgysjfjdjdzffjgsewla
wjzvulwlwodgygdqsfskzaksaskcetzwacjlhvywzastwgsgjsjmwzzmvjuwgkjvfufwlgwywsjksy
wgyldvwxgjllzwgwvvwwkaadzffjtlaoxaksgsskjvfufwlgwyws

6.) Perform X monoalphabetic frequency analysis where X is the key length. So, in this example, there are 5 alphabets used.

(The index of these values correspond to the index in the alphabet.)

///////// Frequency Analysis on Cipher 0 /////////

[0, 2, 4, 4, 15, 18, 2, 0, 10, 18, 21, 5, 3, 3, 0, 3, 0, 20, 1, 3, 13, 26, 6, 5, 10, 23]

['v', 'z', 'k'] (Most common ciphertext characters)

///////// Frequency Analysis on Cipher 1 /////////

[12, 3, 5, 6, 18, 6, 9, 14, 15, 0, 2, 4, 8, 14, 25, 1, 1, 12, 15, 21, 6, 2, 10, 0, 5, 0]

['o', 't', 'e'] (Most common ciphertext characters)

///////// Frequency Analysis on Cipher 2 /////////

[21, 21, 3, 3, 3, 0, 4, 0, 19, 2, 3, 6, 25, 4, 7, 9, 17, 0, 1, 6, 10, 17, 18, 0, 0, 15]

['m', 'a', 'b'] (Most common ciphertext characters)

///////// Frequency Analysis on Cipher 3 /////////

[19, 19, 1, 0, 16, 10, 21, 6, 0, 6, 0, 5, 0, 13, 3, 2, 9, 26, 3, 4, 17, 16, 1, 3, 9, 5]

['r', 'g', 'a'] (Most common ciphertext characters)

///////// Frequency Analysis on Cipher 4 /////////

[14, 0, 2, 7, 6, 18, 17, 3, 0, 18, 12, 16, 3, 3, 4, 0, 4, 0, 19, 3, 6, 10, 25, 2, 10, 12]

['w', 's', 'f'] (Most common ciphertext characters)

7.) Identify the potential key used to encrypt the plaintext based on the most common ciphertext characters. For instance, if the character ‘v’ occurs the most in Cipher0, then the first guess is that v is the character e. So, v - e = ? mod26 or 21-4=17 mod 26. r is the character at index 17, therefore, the alphabet is likely shifted by 17. And the first character of the keyword begins with r. You do the same calculation for each of the shift-by-n ciphers. Again, see the lecture notes if you can’t remember how this calculation works.

///////// Cipher 0 Potential Key Values /////////

['v', 'z', 'k'] (Most frequently occurring ciphertext characters from left to right)

['r'] (Most likely first character in the keyword)

///////// Cipher 1 Potential Key Values /////////

['o', 't', 'e'] (Most frequently occurring ciphertext characters from left to right)

['k'] (Most likely second character in the keyword)

///////// Cipher 2 Potential Key Values /////////

['m', 'a', 'b'] (Most frequently occurring ciphertext characters from left to right)

['i'] (Most likely third character in the keyword)

///////// Cipher 3 Potential Key Values   /////////
['r', 'g', 'a'] (Most frequently occurring ciphertext characters from left to right)
['n'] (Most likely fourth character in the keyword)

///////// Cipher 4 Potential Key Values   /////////
['w', 's', 'f'] (Most frequently occurring ciphertext characters from left to right)
['s'] (Most likely fifth character in the keyword)

8.) Use the suspected key (‘rkins’) to decrypt the plaintext. One way to do this is to write a reverse shift-by-n function with two parameters (the character to be decrypted and an int). This function would and returns the char shifted to the **left** by X amount. Given the fact that the most likely key is [‘r’, ‘k’, ‘i’, ‘n’, ‘s’] we get the following plaintext. We can make out some words, but the key is not 100% correct because it does not result in a fully readable plaintext. The key value that seems to produce the incorrect plaintext is from Cipher1. We go back to the frequencies for Cipher1 and use the second most frequently occurring ciphertext character (t) as the substitute for e. 19-4 = 15 mod 26 which is p or a shift of 15. Our new keyword is [‘r’, ‘p’, ‘i’, ‘n’, ‘s’]. That didn’t really decrypt the message successfully. So, the third most frequently occurring ciphertext character (e) would mean a shift of 0. 4 -4 = 0 mod26. The new keyword is [‘r’, ‘a’, ‘i’, ‘n’, ‘s’]. Continue through the letter frequencies until you find a key that successfully provides the plaintext. **Include all your iterations in your lab report.**

///////// Plaintext using the key: rkins /////////
i **x** e a r t **x** e d r u **c** s e c h **e** i n g t **e** n i g h **j** b u t s x e h e a h s o n l o w h i s f e r s o
v s o m e g u i e t s o n v e h s a t i e n s h e i c o m i d g i n t m e l v e j h i r t o f l i g x t t
h e c o o n l y t w i n w s r e f b e c t t x e s t a h s t h a j g u i d u m e t o m a r d s i a l v a
j i o n i i t o p p u d a n o b d m a n q l o n g j h e w a o h o p i d g t o f y n d s o c e o l d v
o r g o j t e n w e r d s o h a n c i u n t m e b o d i e i h e t u h n e d t e m e a s y f t o s q y
h u r h y b o y y t s w a y t i n g j h e r e v o r y o k i t s g e n n a t q k e a l e t t o d h a g m
e q w a y f h o m y o k t h e r u s n o t x i n g t x a t a h k n d r e t m e n o h m o r e s o u l
d u v e r d e i b l e i s t h e h a i n s t o w n i d a f r i s a g o n d a t a k u s o m e j i m e t e
d o t h u t h i n w s w e n u v e r h q d t h e m i l d d e g s c r o o u t i d t h e n y g h t a i t
h e y w r o w r u s t l e i s l o n w i n g f e r s o m u s o l i j a r y c e m p a n o i k n o m t h
a t y m u s t t o w h a j s r i g x t a s s k r e a s a i l i m q n j a r e r i s e i l i k e e l y m p k
s a b o l e t h e i e r e n w e t i i i e e k t e c u r e m h a t s t e e p i d s i d e v r i g h j e n e
d e f t h i i t h i n w t h a t y v e b e s o m e i j s g o n d a t a k u a l o t j o d r a w m e a w
q y f r o c y o u t x e r e s d o t h i d g t h a j a h u n t r e d m u n o r m e r e c o k l d e v u
r d o i r l e s s j h e r a y n s d o m n i n a v r i c a w o n n a j a k e s e m e t i c e t o d e t h
e t x i n g s m e n e v u r h a d x u r r y r o y s h u s w a i j i n g t x e r e f e r y o u y t s g o
d n a t a a e a l o j t o d r q g m e a m a y f r e m y o u j h e r e i n o t h y n g t h q t a h u d
d r e d c e n o r c o r e c e u l d e l e r d o y b l e s i t h e r q i n s d e w n i n q f r i c q i b l
e i s t h e h a i n s t o w n i d a f r i s a i b l u s s t h u r a i n i d o w n y n a f r y c a i b b e
s s t x e r a i d s d o w d i n a f h i c a i r l e s s j h e r a y n s d o m n i n a v r i c a w o n n
a j a k e s e m e t i c e t o d e t h e t x i n g s m e n e v u r h a d

///////// Plaintext using the key: rpins /////////

*i s e a r t s e d r u x s e c h z i n g t z n i g h e b u t s s e h e a c s o n l j w h i s a e r s o q
s o m e b u i e t n o n v e c s a t i z n s h e d c o m i y g i n t h e l v e e h i r t j f l i g s t t h
e x o o n l t t w i n r s r e f w e c t t s e s t a c s t h a e g u i d p m e t o h a r d s d a l v a e
i o n i d t o p p p d a n o w d m a n l l o n g e h e w a j h o p i y g t o f t n d s o x e o l d q o
r g o e t e n w z r d s o c a n c i p n t m e w o d i e d h e t u c n e d t z m e a s t f t o s l y h
u r c y b o y t t s w a t t i n g e h e r e q o r y o f i t s g z n n a t l k e a l z t t o d c a g m e l
w a y f c o m y o f t h e r p s n o t s i n g t s a t a h f n d r e o m e n o c m o r e n o u l d p
v e r d z i b l e d s t h e c a i n s o o w n i y a f r i n a g o n y a t a k p s o m e e i m e t z d
o t h p t h i n r s w e n p v e r h l d t h e h i l d d z g s c r j o u t i y t h e n t g h t a d t h e y
r r o w r p s t l e d s l o n r i n g f z r s o m p s o l i e a r y c z m p a n j i k n o h t h a t t m
u s t o o w h a e s r i g s t a s s f r e a s v i l i m l n j a r z r i s e d l i k e z l y m p f s a b o
g e t h e d e r e n r e t i i d e e k t z c u r e h h a t s o e e p i y s i d e q r i g h e e n e d z f t
h i d t h i n r t h a t t v e b e n o m e i e s g o n y a t a k p a l o t e o d r a r m e a w l y f r o
x y o u t s e r e s y o t h i y g t h a e a h u n o r e d m p n o r m z r e c o f l d e v p r d o i
m l e s s e h e r a t n s d o h n i n a q r i c a r o n n a e a k e s z m e t i x e t o d z t h e t s
i n g s h e n e v p r h a d s u r r y m o y s h p s w a i e i n g t s e r e f z r y o u t t s g o y n
a t a v e a l o e t o d r l g m e a h a y f r z m y o u e h e r e d n o t h t n g t h l t a h u y d r
e d x e n o r x o r e c z u l d e g e r d o t b l e s d t h e r l i n s d z w n i n l f r i c l i b l e d s
t h e c a i n s o o w n i y a f r i n a i b l p s s t h p r a i n d d o w n t n a f r t c a i b w e s s t
s e r a i y s d o w y i n a f c i c a i m l e s s e h e r a t n s d o h n i n a q r i c a r o n n a e a
k e s z m e t i x e t o d z t h e t s i n g s h e n e v p r h a d*

///////// Plaintext w/ key: rains /////////

*i h e a r t h e d r u m s e c h o i n g t o n i g h t b u t s h e h e a r s o n l y w h i s p e r s o
f s o m e q u i e t c o n v e r s a t i o n s h e s c o m i n g i n t w e l v e t h i r t y f l i g h t t
h e m o o n l i t w i n g s r e f l e c t t h e s t a r s t h a t g u i d e m e t o w a r d s s a l v a t
i o n i s t o p p e d a n o l d m a n a l o n g t h e w a y h o p i n g t o f i n d s o m e o l d f o
r g o t t e n w o r d s o r a n c i e n t m e l o d i e s h e t u r n e d t o m e a s i f t o s a y h u
r r y b o y i t s w a i t i n g t h e r e f o r y o u i t s g o n n a t a k e a l o t t o d r a g m e a w
a y f r o m y o u t h e r e s n o t h i n g t h a t a h u n d r e d m e n o r m o r e c o u l d e v
e r d o i b l e s s t h e r a i n s d o w n i n a f r i c a g o n n a t a k e s o m e t i m e t o d o t
h e t h i n g s w e n e v e r h a d t h e w i l d d o g s c r y o u t i n t h e n i g h t a s t h e y g
r o w r e s t l e s s l o n g i n g f o r s o m e s o l i t a r y c o m p a n y i k n o w t h a t i m u
s t d o w h a t s r i g h t a s s u r e a s k i l i m a n j a r o r i s e s l i k e o l y m p u s a b o v
e t h e s e r e n g e t i i s e e k t o c u r e w h a t s d e e p i n s i d e f r i g h t e n e d o f t h
i s t h i n g t h a t i v e b e c o m e i t s g o n n a t a k e a l o t t o d r a g m e a w a y f r o
m y o u t h e r e s n o t h i n g t h a t a h u n d r e d m e n o r m o r e c o u l d e v e r d o i
b l e s s t h e r a i n s d o w n i n a f r i c a g o n n a t a k e s o m e t i m e t o d o t h e t h i
n g s w e n e v e r h a d h u r r y b o y s h e s w a i t i n g t h e r e f o r y o u i t s g o n n a
t a k e a l o t t o d r a g m e a w a y f r o m y o u t h e r e s n o t h i n g t h a t a h u n d r e
d m e n o r m o r e c o u l d e v e r d o i b l e s s t h e r a i n s d o w n i n a f r i c a i b l e s
s t h e r a i n s d o w n i n a f r i c a i b l e s s t h e r a i n s d o w n i n a f r i c a i b l e s s t*

h e r a i n s d o w n i n a f r i c a i b l e s s t h e r a i n s d o w n i n a f r i c a g o n n a t a k e s o m e t i m e t o d o t h e t h i n g s w e n e v e r h a d

Lab 03 Template

1) **List the five most common trigrams**
   (10 total points. 2 points each)

2) **Show the difference between the starting indexes of the five most common trigrams.**
   (10 total points, 2 points each)

3) **Based on your findings, what do you suspect the key length is? Justify your answer.**
   (10 total points, 5 points length value, 5 points justification)

4) **Separate the ciphertext into X shift-by-N ciphers where X is the length of the key and perform monoalphabetic frequency analysis on each. What are the three most common ciphertext characters in each of the shift-by-N ciphers?**
   (20 total points, 10 points for separating into X shift-by-N ciphers, 10 points for the common ciphertext characters in each shift-by-N cipher.)

5) **Decrypt the ciphertext using the potential key values you found in question 5. Show all iterations, the final keyword, and the final plaintext.**
   (25 total points, 15 points for all iterations, 5 points for correct keyword, 5 points for correct plaintext)

6) **Submit your documented python code on canvas.**
   (25 total points, 10 points for documentation, 15 points for working code. No points awarded if the code is copy/pasted from someone else.)