Problem 5- Stream Vignere Cipher Decryption

February 27, 2020

[5]:
```
import sys
sys.path.append('../')
import crypto_utils as utils
%load_ext autoreload
%autoreload 2

# probabilities of occurrence of 26 letters english alphabet
eng_alph_probs = [.082, .015, .028, .043, .127, .022, .020, .061, .070, .002, .
 ↪008, .040, .024, .067, .075, .019, .001, .060, .063, .091, .028, .010, .023,␣
 ↪.001, .020, .001]
alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
```

```
The autoreload extension is already loaded. To reload it, use:
  %reload_ext autoreload
```

[6]:
```
cipher_text =␣
 ↪"TOSIEBCBBPKZINTEKMKEZTIBIMTSXMFADGZETGXIQWUQSVFTVCPWRSQHGXLLBVGFBDIWKDTBJXDFLBKVLSWEMMGONF

"""
This problem is based on Stinson's problem 2.29 p.58.
First I need to use a similar technique to what I did in problem 1. The main␣
 ↪adjustment I need to make is when building my y1,y2,etc. They are supposed␣
 ↪to still be representative of the letter frequencies in the alphabet overall.
 ↪ But I need to account for the incrementing of the key. So for example: if␣
 ↪I'm checking a potential key length of 4 to see what the IC's of the y's␣
 ↪are, I add first character to y1, second to y2, third to y3, and fourth to␣
 ↪y4. Next in a normal Vignere I would add the fifth to y1, however, I know␣
 ↪the keyword has been incremented. So I need to add a character one less than␣
 ↪that to account for the shift in keyword. When I get around to the ninth␣
 ↪character, I need to subtract the character by two before adding to y1 to␣
 ↪preserve the appropriate character frequencies. This worked and I discovered␣
 ↪my key length is 6.
"""
```

[6]:
```
"\nThis problem is based on Stinson's problem 2.29 p.58.\nFirst I need to use a
similar technique to what I did in problem 1. The main adjustment I need to make
is when building my y1,y2,etc. They are supposed to still be representative of
```

the letter frequencies in the alphabet overall. But I need to account for the incrementing of the key. So for example: if I'm checking a potential key length of 4 to see what the IC's of the y's are, I add first character to y1, second to y2, third to y3, and fourth to y4. Next in a normal Vignere I would add the fifth to y1, however, I know the keyword has been incremented. So I need to add a character one less than that to account for the shift in keyword. When I get around to the ninth character, I need to subtract the character by two before adding to y1 to preserve the appropriate character frequencies. This worked and I discovered my key length is 6.\n"

[7]:
```
k = 6  # looks to be the appropriate keyword

print("k= "+ str(k))
[y1,y2,y3,y4,y5,y6,y7] = utils.calc_ys_mod(k, cipher_text)
print(utils.index_of_coincidence(y1))
print(utils.index_of_coincidence(y2))
print(utils.index_of_coincidence(y3))
print(utils.index_of_coincidence(y4))
print(utils.index_of_coincidence(y5))
print(utils.index_of_coincidence(y6))
# print(utils.index_of_coincidence(y7))
```

```
k= 6
0.061038961038961045
0.060389610389610396
0.0551948051948052
0.062337662337662345
0.07532467532467534
0.04870129870129871
```

[8]:
```
# now calculate Mg(y)s
print("y1-")
utils.calc_M(y1)  # T? 0.06
print("y2-")
utils.calc_M(y2)  # H? 0.063
print("y3-")
utils.calc_M(y3)  # E? 0.056
print("y4-")
utils.calc_M(y4)  # O? 0.066
print("y5-")
utils.calc_M(y5)  # R? 0.063
print("y6-")
utils.calc_M(y6)  # y? 0.062
# print("y7-")
# utils.calc_M(y7)  # E here 0.066
```

```
y1-
A-G: 0.041 0.031 0.033 0.038 0.042 0.037 0.045
```

```
H-N: 0.036 0.036 0.042 0.041 0.033 0.03 0.047
O-U: 0.031 0.041 0.037 0.033 0.03 0.06 0.048
V-Z: 0.033 0.036 0.048 0.034 0.039
y2-
A-G: 0.04 0.037 0.032 0.042 0.034 0.035 0.04
H-N: 0.063 0.038 0.035 0.04 0.039 0.028 0.041
O-U: 0.036 0.032 0.041 0.041 0.042 0.036 0.05
V-Z: 0.038 0.038 0.039 0.034 0.029
y3-
A-G: 0.049 0.026 0.025 0.037 0.056 0.037 0.034
H-N: 0.044 0.04 0.037 0.037 0.046 0.035 0.037
O-U: 0.036 0.038 0.033 0.038 0.043 0.046 0.036
V-Z: 0.027 0.044 0.038 0.045 0.037
y4-
A-G: 0.037 0.047 0.047 0.046 0.03 0.027 0.041
H-N: 0.042 0.032 0.032 0.044 0.029 0.032 0.043
O-U: 0.066 0.037 0.036 0.036 0.041 0.033 0.041
V-Z: 0.039 0.03 0.034 0.038 0.041
y5-
A-G: 0.028 0.037 0.04 0.032 0.038 0.056 0.052
H-N: 0.042 0.029 0.036 0.037 0.039 0.03 0.037
O-U: 0.03 0.027 0.052 0.063 0.043 0.025 0.046
V-Z: 0.049 0.038 0.029 0.04 0.027
y6-
A-G: 0.03 0.038 0.044 0.031 0.037 0.041 0.038
H-N: 0.035 0.039 0.039 0.039 0.042 0.042 0.042
O-U: 0.036 0.033 0.034 0.042 0.035 0.034 0.04
V-Z: 0.035 0.035 0.041 0.062 0.037
```

[9]:
```python
# keyword is THEORY
# Now another change needs to be made so that every time the shift is used (i.e.
 ↪ every k=6 letters), the values in the array are incremented by one
shift = [alphabet.index("T"), alphabet.index("H"), alphabet.index("E"),␣
 ↪alphabet.index("O"), alphabet.index("R"), alphabet.index("Y")]
numerical_cipher_text = [0]*len(cipher_text)
for letter in range(len(cipher_text)):
  numerical_cipher_text[letter] = alphabet.index(cipher_text[letter])

# decrypt using keyword
for dec_let in range(len(numerical_cipher_text)):
  if dec_let % k == 0:
    numerical_cipher_text[dec_let:dec_let+k] = [(x - y)%26 for x, y in␣
 ↪zip(numerical_cipher_text[dec_let:dec_let+k], shift)]
    shift = [x+y for x,y in zip(shift, [1,1,1,1,1,1])]

for i in range(len(numerical_cipher_text)):
  numerical_cipher_text[i] = alphabet[numerical_cipher_text[i]]
```

```
print(''.join(numerical_cipher_text))
```

```
AHOUNDITWASANENORMOUSCOALBLACKHOUNDBUTNOTSUCHAHOUNDASMORTALEYESHAVEEVERSEENFIREB
URSTFROMITSOPENMOUTHITSEYESGLOWEDWITHASMOULDERINGGLAREWITHLONGBOUNDSTHEHUGECREAT
UREWASFOLLOWINGHARDUPONTHEFOOTSTEPSOFOURFRIENDWESAWSIRHENRYLOOKINGBACKHISFACEWHI
TEINTHEMOONLIGHTHISHANDSRAISEDINHORRORGLARINGHELPLESSLYATTHEFRIGHTFULTHINGWHICHW
ASHUNTINGHIMDOWN
```