Cryptanalysis of Vigenère

Find key length t

Here use Kasiski Examination to find the GCD of distance for all patterns of length 3

```
def find_repeated_patterns(text, length):
    pattern_positions = defaultdict(list)

    for i in range(len(text) - length + 1):
        pattern = text[i:i + length]
        pattern_positions[pattern].append(i)

    repeated_patterns = {pattern: positions for pattern, positions in pattern_positions.items()
if len(positions) > 1}

    return repeated_patterns

def find_gcd_of_distances(positions):
    distances = [positions[i+1] - positions[i] for i in range(len(positions) - 1)]
    if distances:
        return reduce(gcd, distances)
    return None

def kasiski_examination(text, length=2):
    repeated_patterns = find_repeated_patterns(text, length)
    gcd_results = {}

    for pattern, positions in repeated_patterns.items():
        gcd_value = find_gcd_of_distances(positions)
        if gcd_value:
            gcd_results[pattern] = gcd_value

    return gcd_results

# Cipher Text
cipher_text =
"FRSGKIPMJLNZJTAJZAOBJXLZKODOZFGSCPERSUSILENBNVTNQZXKLHJZGCMAFJDGNSVEYOBNZILGFVZTIXSNSQRWOQGYWCEJL


# Running Kasiski Examination for patterns of length 3
kasiski_results_length_2 = kasiski_examination(cipher_text, length=3)
```

Plot the Top 10 Most Frequent GCD Values in Kasiski Examination Results

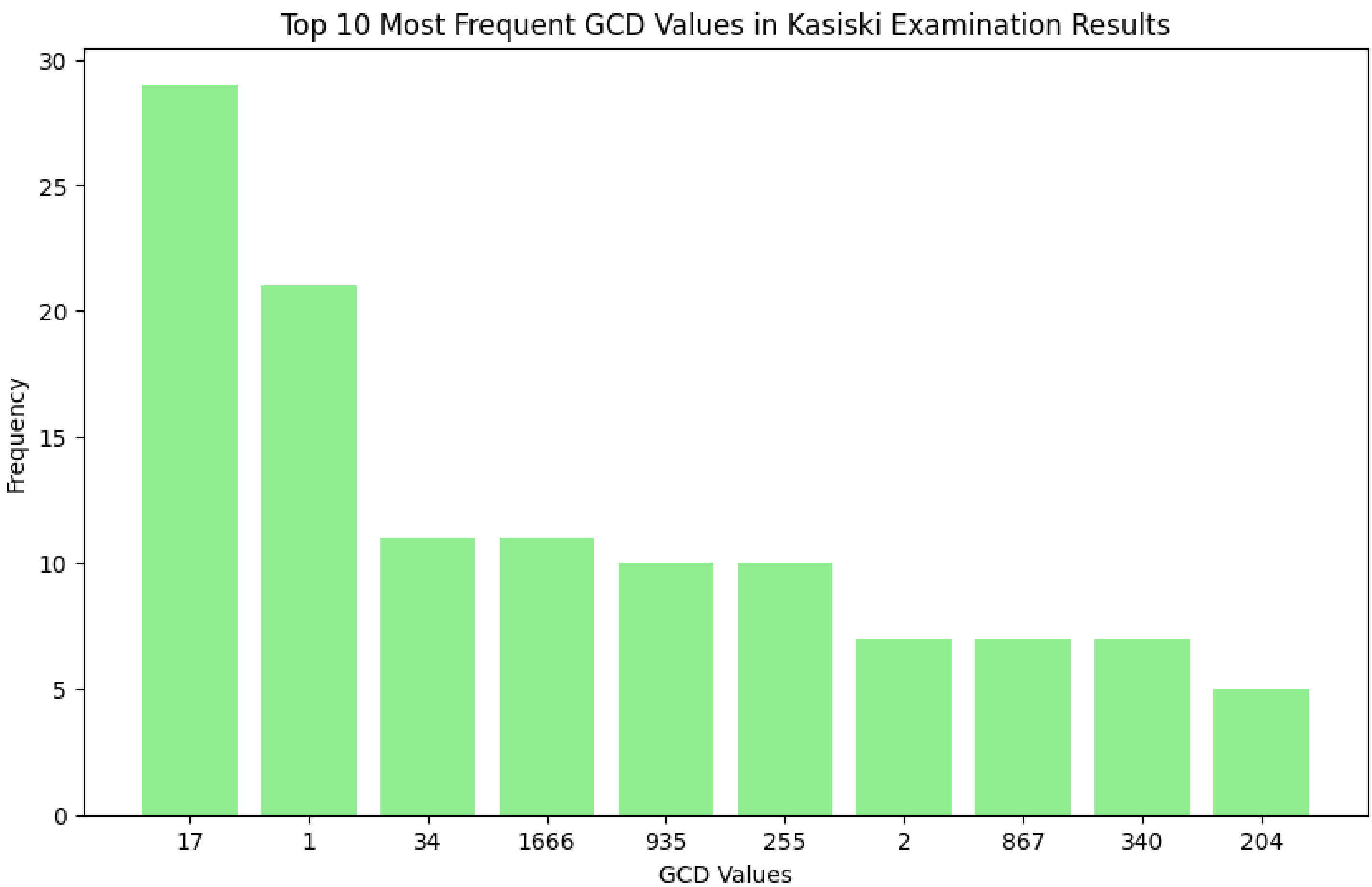
```
gcd_values = list(kasiski_results_length_4.values())
gcd_frequency = defaultdict(int)
for gcd_value in gcd_values:
    gcd_frequency[gcd_value] += 1


sorted_gcd_frequency = sorted(gcd_frequency.items(), key=lambda x: x[1], reverse=True)


top_10_gcd = sorted_gcd_frequency[:10]


gcds_top_10 = [item[0] for item in top_10_gcd]
frequencies_top_10 = [item[1] for item in top_10_gcd]


plt.figure(figsize=(10, 6))
plt.bar(range(len(gcds_top_10)), frequencies_top_10, color='lightgreen')
plt.xlabel('GCD Values')
plt.ylabel('Frequency')
plt.title('Top 10 Most Frequent GCD Values in Kasiski Examination Results')
plt.xticks(range(len(gcds_top_10)), gcds_top_10)
plt.show()
```



Here we find the the most frequent GCD of distance for all patterns of length 3 is 17 and there are also many multiple of 17 were very frequent. So we make the decision to choose the key length $t = 17$ here.

Verify key length t

We can also verify the correctness of $t = 17$ using the method index of coincidence
Here print the sum $S = \sum_{i=0}^{25} q_i^2$ for all sequence $c_a c_{a+t} c_{a+2t\ldots}$, $a \in [1, 17]$

```
# Define the alphabet (A = 0, ..., Z = 25)
alphabet = {chr(i + ord('A')): i for i in range(26)}

def calculate_S_values(ciphertext, period=17):
    n = len(ciphertext)
    S_values = []


    for i in range(period):

        sequence = [ciphertext[j] for j in range(i, n, period) if ciphertext[j] in alphabet]


        frequency_count = Counter(sequence)


        total_count = sum(frequency_count.values())
        q_i = {alphabet[char]: count / total_count for char, count in frequency_count.items()}


        S_i = sum((q_i.get(i, 0) ** 2) for i in range(26))


        S_values.append(S_i)

    return S_values

S_values = calculate_S_values(cipher_text, period=17)
S_values
```

```
[0.07843076058799663,
 0.07711994007677664,
 0.06987754071455624,
 0.06943567731347051,
 0.07183436434793586,
 0.0706350208307032,
 0.07669486175987882,
 0.0765686150738543,
 0.07063502083070318,
 0.06470142658755208,
 0.07820982199217269,
 0.07088751420275215,
 0.06665825022093169,
 0.06482767327357657,
 0.0807978790556748,
 0.06949880065648276,
 0.07852543870723391]
```

We can see sums were all ≈ 0.065 when $t = 17$
Also verify sum ≈ 0.038 when $t \neq 17$ (not shown here since it's a lot…)
So the key length $t = 17$ should be correct

compute key k

Use the method that calculating $I_j = \sum_{i=0}^{25} p_i \cdot q_{i+j}$, and find k by determing $I_k \approx 0.065$ and $I_j < 0.0455, j \neq k$ for all sequence $c_a c_{a+t} c_{a+2t\ldots}$, $a \in [1, 17]$

```python
# English letter frequencies, p_j for j=0 to 25 corresponding to A to Z
english_frequencies = np.array([
    0.08167, 0.01492, 0.02782, 0.04253, 0.12702,  # A - E
    0.02228, 0.02015, 0.06094, 0.06966, 0.00153,  # F - J
    0.00772, 0.04025, 0.02406, 0.06749, 0.07507,  # K - O
    0.01929, 0.00095, 0.05987, 0.06327, 0.09056,  # P - T
    0.02758, 0.00978, 0.02360, 0.00150, 0.01974,  # U - Y
    0.00074  # Z
])


# Alphabet to index dictionary
alphabet = {chr(i): i - ord('A') for i in range(ord('A'), ord('Z') + 1)}
index_to_char = {i: chr(i + ord('A')) for i in range(26)}

# Function to calculate frequency of each letter in a sequence
def calculate_frequencies(sequence):
    total_count = len(sequence)
    frequencies = np.zeros(26)

    counter = Counter([letter for letter in sequence if letter in alphabet])

    for letter, count in counter.items():
        frequencies[alphabet[letter]] = count / total_count

    return frequencies

sequences = []
for i in range(17):
    sequence = cipher_text[i::17]
    sequences.append(sequence)

k_i = []

for i in range(17):
    sequence = sequences[i]
    q_i_j = calculate_frequencies(sequence)

    max_S_i = -float('inf')
    best_k = 0

    for k in range(26):
        S_i = np.sum(english_frequencies * np.roll(q_i_j, -k))

        if S_i >= 0.06:
            best_k = k
            break

    k_i.append(best_k)

key_letters = ''.join([index_to_char[k] for k in k_i])

print(f"The key is: {key_letters}")
```

```
The key is: QAOBKGLTCDVRRTZVL
```

Decryption

Decrypt the cipher text using the key we found

```
# Function to decrypt Vigenère cipher
def vigenere_decrypt(ciphertext, key):
    decrypted_text = []
    for i in range(len(ciphertext)):
        cipher_char = ord(ciphertext[i]) - ord('A')
        key_char = ord(key[i % len(key)]) - ord('A')
        plain_char = (cipher_char - key_char + 26) % 26
        decrypted_text.append(chr(plain_char + ord('A')))
    return ''.join(decrypted_text)


key = "QAOBKGLTCDVRRTZVL"

# Decrypt the ciphertext
plain_text = vigenere_decrypt(cipher_text, key)
plain_text
```

The plaintext is:
PREFACETHISISABOOKONINFORMATIONTHEORETICALLYSECUREMULTIPARTYCOMPUTATIONMPCANDSECRETSHARINGANDABOUTTHEINTIMATEANDFASCINATINGRELATIONSHIPBETWEENTHETWONOTIONSWEDECIDEDTOWRITETHEBOOKBECAUSEWEFELTTHATACOMPREHENSIVETREATMENTOFUNCONDITIONALLYSECURETECHNIQUESFORMPCWASMISSINGINTHELITERATUREINPARTICULARBECAUSESOMEOFTHEFIRSTGENERALPROTOCOLSWEREFOUNDBEFOREAPPROPRIATEDEFINITIONSOFSECURITYHADCRYSTALLIZEDPROOFSOFTHOSEBASICSOLUTIONSHAVEBEENMISSINGSOFARWEPRESENTTHEBASICFEASIBILITYRESULTSFORUNCONDITIONALLYSECUREMPCFROMTHELATEEIGHTIESGENERALIZATIONSTOARBITRARYACCESSSTRUCTURESUSINGLINEARSECRETSHARINGANDASELECTIONOFMORERECENTTECHNIQUESFOREFFICIENCYIMPROVEMENTSWEALSOPRESENTOUROWNVARIANTOFTHEUCFRAMEWORKINORDERTOBEABLETOGIVECOMPLETEANDMODULARPROOFSFORTHEPROTOCOLSWEPRESENTWEALSOPRESENTAGENERALTREATMENTOFTHETHEORYOFSECRETSHARINGANDINPARTICULARSECRETSHARINGSCHEMESWITHADDITIONALALGEBRAICPROPERTIESALSOASUBJECTWHEREATREATMENTONTEXTBOOKLEVELSEEMSTOBEMISSINGTHELITERATUREONEOFTHETHINGSWEFOCUSONISASYMPTOTICRESULTSFORMULTIPLICATIVESECRETSHARINGWHICHHASVARIOUSINTERESTINGAPPLICATIONSTHATWEPRESENTINTHEMPCPARTOURAMBITIONHASBEENTOCREATEABOOKTHATWILLBEOFINTERESTTOBOTHCOMPUTERSCIENTISTSANDMATHEMATICIANSANDCANBEUSEDFORTEACHINGATSEVERALDIFFERENTLEVELSWHEREDIFFERENTPARTSOFTHEBOOKWILLBEUSEDWEHAVETHEREFORETRIEDTOMAKEBOTHMAINPARTSBESELFCONTAINEDEVENIFTHISIMPLIEDSOMEOVERLAPBETWEENTHEPARTSTHISMEANSTHATTHEREARESEVERALDIFFERENTWAYSTOREADTHEBOOKANDWEGIVEAFEWSUGGESTIONSFORTHISBELOWINPARTICULARTHECONCEPTOFSECRETSHARINGOFCOURSEAPPEARSPROMINENTLYINBOTHPARTSINTHEFIRSTMPCPARTHOWEVERITISINTRODUCEDONLYASATOOLONANEEDTOKNOWBASISINTHESECONDPARTWEREINTRODUCETHENOTIONBUTNOWASAGENERALCON

CEPTTHATISINTERESTINGINITSOWNRIGHTANDWITHACOMPREHENSIVETREATMENTOFTHEMATH
EMATICALBACKGROUNDTHEBOOKISINTENDEDTOBESELFCONTAINEDENOUGHTOBEREADBYADV
ANCEDUNDERGRADUATESTUDENTSANDTHEAUTHORSHAVEUSEDLARGEPARTSOFTHEMATERIAL
INTHEBOOKFORTEACHINGCOURSESATTHISLEVELBYCOVERINGASELECTIONOFMOREADVANCE
DMATERIALTHEBOOKCANALSOBEUSEDFORAGRADUATECOURSEHOWTOUSETHISBOOKFORACO
URSEONADVANCEDUNDERGRADLEVELFORCOMPUTERSCIENCESTUDENTSWERECOMMENDTOC
OVERCHAPTERONECHAPTERFIVETHISWILLINCLUDETHEBASICFEASIBILITYRESULTSFORUNCON
DITIONALLYSECUREMPCANDTHEUCMODELFORSOMEEXTRAPERSPECTIVEITMAYALSOBEAGOOD
IDEATOCOVERCHAPTERCHAPTERKWHICHISBASICALLYASURVEYOFCRYPTOGRAPHICALLYSECU
RESOLUTIONSFORAGRADUATELEVELCOMPUTERSCIENCECOURSEWERECOMMENDTOINCLUDE
ALSOCHAPTERLANDCHAPTERMASTHEYCONTAINSSEVERALRECENTTECHNIQUESANDSURVEYS
OMEOPENPROBLEMSFORACOURSEINMATHEMATICSONSECRETSHARINGANDAPPLICATIONSWE
RECOMMENDTOCOVERFIRSTCHAPTERONECHAPTERTHREEANDCHAPTERSIXTHISWILLGIVEANIN
TUITIONFORWHATSECRETSHARINGISANDHOWITISUSEDINMPCTHENPARTTWOSHOULDBECOVE
REDTOPRESENTTHEGENERALTHEORYOFALGEBRAICSECRETSHARINGFINALLYTHELASTPARTOF
CHAPTERNINECANBEUSEDTOPRESENTSOMEOFTHEMOREADVANCEDAPPLICATIONSACKNOWLE
DGEMENTSWETHANKIRENEGIACOMELLIMORTENDAHLJORGENSENANTONIOMACEDONEANDSAR
AHZAKARIASFORDOINGANENORMOUSAMOUNTOFWORKINPROOFREADINGANDCOMMENTINGW
EALSOTHANKSARAHZAKARIASFORWRITINGSOMEOFTHESIMULATIONPROOFSANDHELPINGUSAV
OIDMANYMOREBLUNDERSTHANAREPRESENTNOW