# Breaking Vigenere

Last time – gave two techniques to find the keyword length.

Once we have it, what do we do???

Let the keyword length be k.

Step 1: Break letters into k bins as you would have done using the Index of Coincidence test.

Bin 0: c[0], c[k], c[2*k],…

Bin 1: c[1], c[k+1], c[2*k+1]

..

Bin k-1: c[k-1], c[2*k-1], c[3*k-1], …

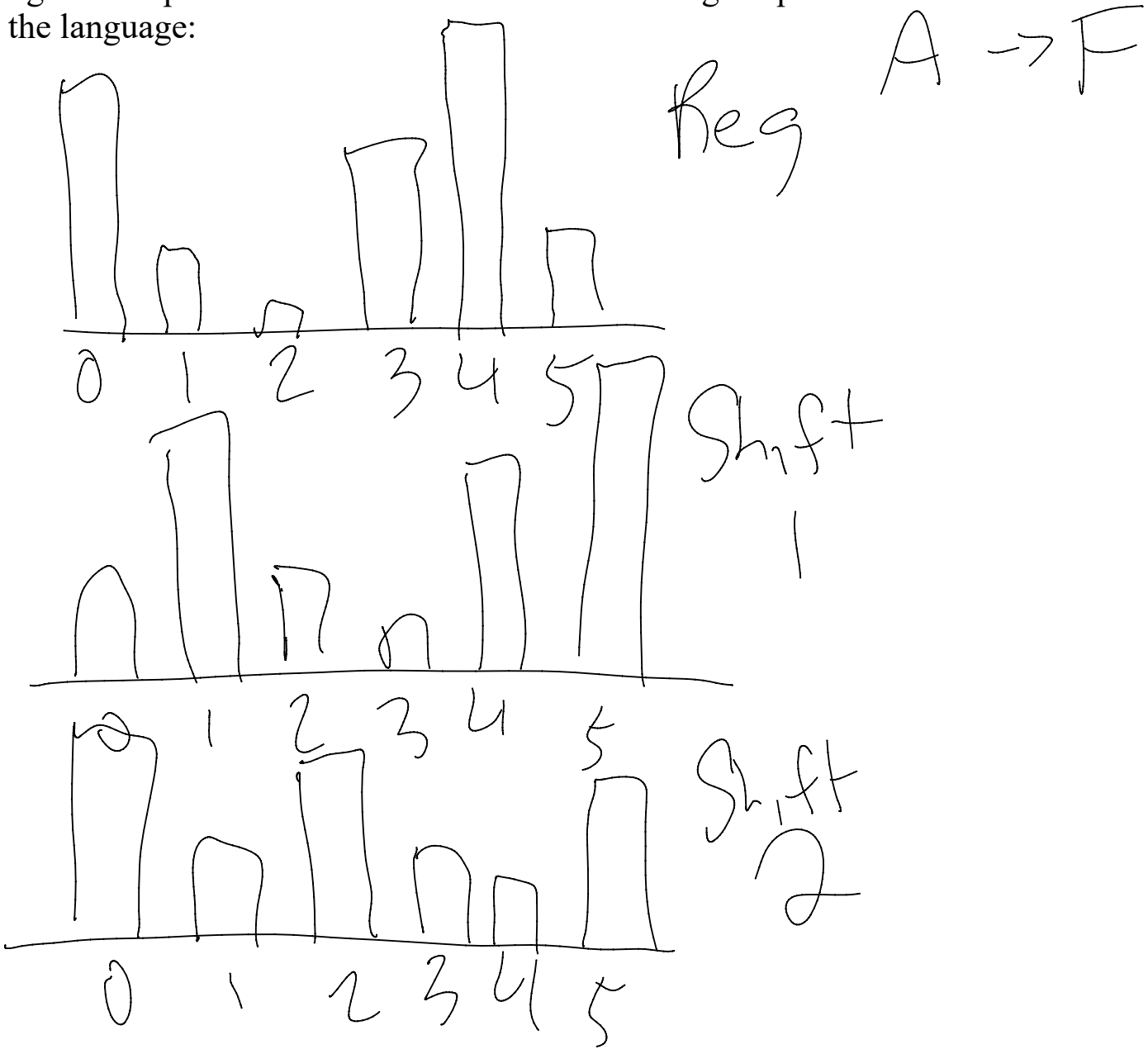
We had previously done the IC of each bin and found them to be “high”. (Consistently closer to 6% or 7% vs. 4% or 5%.)

Each bin represents letters that were shifted by some shift.

# Vigenere Cryptanalysis

Friday, September 4, 2020 11:56 AM

Imagine an alphabet with 6 letters with the following frequencies in the language:




If the keyword started with a C, then the frequency shown above is what we would expect bin 0 to look like.

If the keyword had a letter E for its second letter, we would expect its frequency chart to look like this:


WE brute force the shift on character 0. So we try A, B, C, etc.

For the rest of the characters, we want to calculate a relative shift so that the two charts match up.

So the question is - how many spots do I have to move chart 1 backwards to make it look like chart 0?

If I shift the second graph backwards (cyclically), if I shift it 2 places, it SHOULD look a lot like the first graph.

So the value 2, represents the **RELATIVE SHIFT** between bin 0 and bin 1. That means, if the shift for bin 0 is w, then the shift for bin 1 is w+ 2.

In this manner, we can repeat this process for the following pairs of bins:

0 and 2
0 and 3
0 and 4
…
0 and k-1

When you finish, you should have k-1 relative shift values:

Shift from 0 to 1 is 2
Shift from 0 to 2 is 5
Shift from 0 to 3 is 1
Shift from 0 to 4 is 0
Shift from 0 to 5 is 2

Then our possible keywords are (I am brute forcing the first letter, then the other letters are forced based on the information above.)

ACFBAC
BDACBD
CEBDCE
DFCEDF
EADFEA
FBEAFB

These are the only strings that obey the relative shift values shown above.

Normally when I do this, some of the guesses are WRONG. So if you happen to have 4 letter right, you can usually make out parts of words, and then start playing around with the keyword, making substitutions. Sometimes, you recognize the keyword from the partial letters that are right and get it right away. Other times, you make a guess at a plaintext word and work the keyword out backwards. Either way, after some investigation, it usually works out.

Question: How on earth do we tell if two bar graphs are similar????

Imagine two sets of numbers x and y that add up to 1.

.05, .25, .45, .1, .15
.2, .2, .5, .05, 0.05

Goal: pair up each number from the first list with a single number from the second list and multiply the pair and add all pairs together. Do this in such a way that maximizes that sum.

Greedy works: match up the two maximums, then the next two maximums and so forth.

You'll notice that when the bar graphs match up, that high number match up with high numbers and low numbers match up with low numbers. Which means this sum of products would get maximized if the bar graphs represented the same set of letters.

Mutual Index of Coincidence
---
Given two multisets of objects, what is the probability that if you randomly draw an item from the first set and randomly draw an item from the second set, that the two are the same.

2 bags of candies, grab one candy from each, what is the probability they are the same candy.

20 As, 10 Bs, 50 Cs, 20 Ds (20 + 10 + 50 + 20=100 )
5 As, 20 Bs, 15 Cs, 10 Ds. (5+20+15+10=50)

Sample Space = 100 x 50 (# letters in each bin multiplied)

# of ways for success = 20 x 5 + 10 x 20 + 50 x 15 + 20 x 10 = 1,250
1250

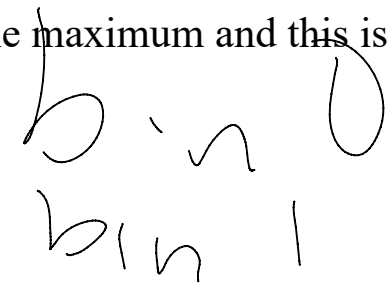
Probability = 1250/5000 = 1/4

So when we are comparing bin 0 to bin 1,

we try all 26 relative shifts of bin 1 to bin 0.
For each relative shift, recalculate the mutual index of coinicidence between the bins.

Of all of these 26 calculations, take the maximum and this is likely your relative shift.

| | | | | |
|---|---|---|---|---|
| 10 | 5 | 20 | 3 | 2 |
| 3 | 12 | 6 | 19 | 4 |




shift 0 = 10 x 3 + 5 x 12 + 20 x 6 + 3 x 19 + 2 x 4 = 275
shift 1 = 10 x 12 + 5 x 6 + 20 x 19 + 3 x 4 + 2 x 3 = 548
shift 2 = 2 10 x 6 + 5 x 19 + 20 x 4 + 3 x 3 + 2 x 12 = 268
shift 3 = 10 x 19 + 5 x 4 + 20 x 3 + 3 x 12 + 2 x 6 = 318
shift 4 = 4 10 x 4 + 5 x 3 + 20 x 12 + 3 x 6 + 2 x 19 = 351

So, the relative shift is likely 1 (bin 1 is shifted over 1 more than bin 0)

Usually some of these are wrong and the max value isn't always the right shift…

Advice: Store the two largest values, and try both…if the keyword is length 7, then this means that you have $2^6$ choices to try for each shift of bin 0.