MATH 587/CSCE 557 - SUMMARY OF CLASS, 3/6/07

I started by summarizing what I consider best strategies for attacking a Vigenere cipher. Namely, a Kasiski attack can often give you a reliable keyword length m fast. Then perform frequency analysis on each coset of the ciphertext, $\{y_1, y_{m+1}, y_{2m+1}, ...\}, \{y_2, y_{m+2}, ...\}, \{y_3, y_{m+3}, ...\}$, to guess each respective shift. At this point you can easily also check that the index of coincidence of each coset is large, confirming your guess of the keyword length. Sometimes decrypting with your guess of the keyword may lead to something that looks close to English - adjusting the keyword then fixes the occasional incorrect letter.

I defined the entropy of a probability distribution $\{p_i\}$ to be $H(\{p_i\}) = -\sum_i p_i \log_2(p_i)$. For example, for a uniform distribution on N symbols, $p_i = 1/N$ and so $H = \log_2(N)$. In the case $N = 26$, we get $H = \log_2(26) = 4.7$. Using that the limit as $x \to 0$ of $x \log_2(x)$ is 0, we see that for a distribution where one p_i is 1 and the rest are 0, $H = 0$. In general, $0 \leq H \leq \log_2(N)$ with equality at the left if and only if the probability is concentrated on one symbol, equality at the right if and only if the distribution is uniform. So H is a measure of the uncertainty in sampling from the distribution.

The units of H are bits per letter, and I gave an example of how clever coding of the letters gets the expected number of bits per letter in a text down to about H. The entropy of the English language will tell us how much compression of English text is possible (gzip compresses a 3700 character file to 1700 characters, a 46% compression), but how do we calculate the entropy of English?

If we use single letters, i.e. p_0 is the frequency of A, etc., we get $H = 4.17$. Using the 26^2 digrams, i.e. $p_{0,0}$ is the frequency of AA, etc., we get $H = 7.1$ bits per digram, or 3.55 bits per letter. Using the 26^3 trigrams, we get $H = 9.9$ bits per trigram, or 3.3 bits per letter. The limit of this using N-grams as $N \to \infty$ is what we want. Claude Shannon, the father of information theory, came up with a clever approach for estimating this, which we followed in class.

I thought of an English sentence and had students guess the next letter (or a space) in turn. Very often they guessed right first time, in fact for 22 of the 35 symbols guessed. They guessed right 2nd time for 8 of the 35 symbols, 3rd time for 2, 4th time never, 5th time once, 11th time once, and 24th time once. Replacing the sentence by the string of numbers representing how many guesses each symbol took is an encoding of the sentence, and its entropy gives us an estimate (actually an upper bound) on the entropy of English. So

$$H = -(22/35)\log_2(22/35) - (8/35)\log_2(8/35) - (2/35)\log_2(2/35) - 3(1/35)\log_2(1/35)$$

This works out as $H = 1.6$ bits per letter (approximately). The redundancy $R = \log_2(N) - H$ and percentage redundancy $r = 1 - H/(\log_2(N))$. We find that English text is about 66% redundant, meaning that it should be compressible to about 1/3 original size. Unicity distance is approximately $\log_2(|K|)/R$.