The University of Sydney

Math3024 Elementary Cryptography and Protocols

Semester 1 **Exercises for Week 5** 2004

Code Breaking

So far we have focused on Vigenère ciphers, and their reduction to monoalphabetic substitutions. Here we show how to use Magma to complete the final step of breaking these ciphers. Recall that the reduction to monoalphabetic substitution is done by the process of *decimation*, by which we lose all 2-character frequency structure of the language. A more sophisticated approach will be necessary for breaking more complex ciphers.

Correlation. We first introduce the concept of correlation of two functions. Let $X = x_1, x_2, \ldots, x_n$ and $Y = y_1, y_2, \ldots, y_n$ be two finite sequences of real numbers, each of length n. We define the correlation of the two sequences to be

$$\mathrm{Corr}(X,Y) = \frac{\sum_{i=1}^{n}(x_i - \mu_X)(y_i - \mu_Y)}{\sigma(X)\sigma(Y)}$$

and where μ_X and μ_Y are the respective *means* of the sequences X and Y:

$$\mu_X = \frac{1}{n}\sum_{i=1}^{n} x_i, \quad \mu_Y = \frac{1}{n}\sum_{i=1}^{n} y_i,$$

and the terms in the denomiators are:

$$\sigma(X) = \Big(\sum_{i=1}^{n}(x_i - \mu_X)^2\Big)^{1/2}, \quad \sigma(Y) = \Big(\sum_{i=1}^{n}(y_i - \mu_Y)^2\Big)^{1/2},$$

called the *standard deviations* of X and Y.

The correlation of two sequences will be a real number between 1 and -1, which measures the linear relation between two sequences. On the following page we give a Magma function which computes the sequence of correlations of each of the cyclic translations of two sequences.

1. **Correlations of sequence translations**. The following code from the course cryptography package finds the correlations between the affine translations of two sequences. Do you understand the syntax? Ask or refer to the online Magma handbook where necessary.

```
function TranslationCorrelations(S1,S2)
    // The sequence of correlations of the sequence S1 with the
    // cyclic translations of the sequence S2.
    n := #S1;
    error if n ne #S2, "Arguments must be of the same length.";

    // Compute the mean value of each sequence:
    mu1 := &+[ S1[k] : k in [1..n] ]/n;
    mu2 := &+[ S2[k] : k in [1..n] ]/n;

    // Compute the standard deviations of each sequence:
    sig1 := Sqrt(&+[ (S1[k]-mu1)^2 : k in [1..n] ]);
    sig2 := Sqrt(&+[ (S2[k]-mu2)^2 : k in [1..n] ]);

    sig := sig1*sig2;
    CorrSeq := [ ];
    for j in [1..n] do
        Corr := &+[ (S1[i] - mu1) * (S2[ij] - mu2) / sig
            where ij := ((i+j-1) mod n) + 1 : i in [1..n] ];
        Append(~CorrSeq,<j,Corr>);
    end for;
    return CorrSeq;
end function;
```

2. **Breaking Vigenère ciphers**. A Vigenère cipher is reduced to an translation cipher by the process of decimation. How does the above function solve the problem of finding the affine translation?

   For completeness we give a function, using the previous one, which matches the frequency distribution of a string with a given standard distribution. This uses the function `FrequencyDistribution`, which is part of the course cryptography package.

```
function TranslationMatches(S,F,r)
    // INPUT:
    // S : Test string.
    // F : Sequence of standard frequencies for the language.
    // r : A real number between 0 and 1.
    // OUTPUT:
    // Returns integers k such that affine translation
    // of S by k has correlation at least r with the standard
    // frequencies given by the real sequence F.

    X := FrequencyDistribution(S);
    CorrSeq := TranslationCorrelations(X,F);
    return [ x[1] : x in CorrSeq | x[2] gt r ];
end function;
```

Use this function on the Vigenére ciphertext samples from the web page the break the enciphering. Recall that you will have to use the functions `Decimation` and `CoincidenceIndex` to first reduce a Vigenère cipher to a monoalphabetic one.

3. **Breaking substitution ciphers**. Suppose that rather than an affine translation, you have reduced to an arbitrary simple substitution. We need to undo an arbitrary permutation of the alphabet. For this purpose we define maps into Euclidean space:

   a. $\mathcal{A} \to \mathcal{A}^2 \to \mathbb{R}^2$ defined by

   $$X \mapsto XX \mapsto (P(X), P(XX)).$$

   b. $\mathcal{A} \to \mathcal{A}^2 \to \mathbb{R}^3$ defined by

   $$X \mapsto XY \mapsto (P(X), P(XY|Y), P(YX|Y)$$

   for some fixed character Y.

   See the document `digraph_frequencies.pdf` for standard vectors for the English language.

4. **Breaking transposition ciphers**. In order to break transposition ciphers it is necessary to find the period m, of the cipher, and then to identify positions i and j within each block $1 + km \leq i, j \leq (k + 1)m$ which were adjacent prior to the permutation of positions. Suppose we guess that m is the correct period. Then for a ciphertext sample $C = c_1c_2 \ldots$, and a choice of $1 \leq i < j \leq m$, we can form the digraph decimation sequence $c_ic_j, c_{i+m}c_{j+m}, c_{i+2m}c_{j+2m}, \ldots$.

   Two statistical measures that we can use on ciphertext to determine if a digraph sequence is typical of the English language are a digraph *coincidence index*

   $$\sum_{X\in\mathcal{A}}^{n} \sum_{Y\in\mathcal{A}}^{n} \frac{n_{XY}(n_{XY} - 1)}{N(N - 1)}$$

   where N is the total number of character pairs, and n_{XY} is the number of occurrences of the pair XY, and the *coincidence discriminant*:

   $$\sum_{X\in\mathcal{A}} \sum_{Y\in\mathcal{A}} \Big(\frac{n_{XY}}{N} - \sum_{Z\in\mathcal{A}} \frac{n_{XZ}}{N} \sum_{Z\in\mathcal{A}} \frac{n_{ZY}}{N}\Big)^2.$$

   The first term is the frequency of XY, and the latter is the product over the frequencies of X as a first character and Y as a second character. The coincidence discriminant measures the discrepancy between the probability space of pairs XY and the product probability space.

   What behavior do you expect for the coincidence index and coincidence discriminant of the above digraph decimation, if i and j were the positions of originally adjacent characters? Test your hypotheses with decimations of "real" English text.

```
function CoincidenceDiscriminant(S)
    // INPUT: A sequence of 2-character strings, produced
    // as decimation of transposition ciphertext, or of
    // adjacent characters in some sample plaintext.
    // OUTPUT: A measure of the difference of probability
    // of association of two characters, relative to their
    // independent probabilities.
    C2S := CodeToString;
    AA := [ C2S(64+i)*C2S(64+j) : i, j in [1..26] ];
    FD1 := FrequencyDistribution(&*[ s[1] : s in S ]);
    FD2 := FrequencyDistribution(&*[ s[2] : s in S ]);
    N := #S;
    F2D := [ RealField() | 0 : i in [1..26^2] ];
    for s in S do
        F2D[Index(AA,s)] +:= 1/N;
    end for;
    return &+[ (F2D[i+26*(j-1)]-FD1[i]*FD2[j])^2 : i,j in [1..26] ];
end function;
```

Why can we assume that $i < j$ in the digraph sequence? What is the obstacle to extending these statistical measures from two to more characters?