MATH 3024 Assignment 01

After completing `MATH3024` *you have been hired by a government agency in Canberra, which believes that members of the American Literature Department at Melbourne University are using enciphered messages to hide terrorist activities against the Australian government. Due to educational cutbacks, they were only able to hire a student who completed the first two weeks of* `MATH3024` *at Sydney Uni, and who is unaware of the weaknesses of classical ciphers. Your supervisor presents you with the following problems before promoting you to more challenging tasks within the agency.*

1. (2 pts) [Vigènere cipher] You suspect that the first ciphertext uses a Vigenère cipher. Find the deciphering key and plaintext.

Solution The period can be determined by finding the period, as described in the week 3 tutorial.

```
for m in [1..20] do
    DecCT := Decimation(CT,i,m);
    printf "%2o: %o\n",
        m, &+[ CoincidenceIndex(DecCT) : i in [1..m] ]/m;
end for;
```

Then following the week 5 tutorial, we expect to pick out the translations of the successive decimations.

```
PT := StripEncoding(Read("blackcat.txt"));
FD := FrequencyDistribution(PT);
eps := 0.50;
for i in [1..m] do
    DecCTi := Decimation(CT,i,m);
    TranslationMatches(DecCTi,FD,eps);
end for;
```

The online javascript program can equivalently be used to determine these translations.

2. (2 pts) [Substitution cipher] The second ciphertext has a standard coincidence index for English, and you suspect a simple substitution cipher. Find the deciphering key and plaintext.

Solution We use the digraph probabilities to determine the substitution cipher used. Before getting started we determine the initial one and two characters probabilities.

```
AZ := [ CodeToString(64+i) : i in [1..26] ];
F1 := FrequencyDistribution(CT);
F2 := DigraphFrequencyDistribution(CT);
// Set up some vector spaces -- easiest manner to test distances
// between vectors, just using the built-in function Norm.
V2 := VectorSpace(RealField(),2);
V3 := VectorSpace(RealField(),3);
```

First we look for ciphertext characters which are separated from the others based on the image vectors $(P(X), P(XX|X))$ in $\mathbb{R}^2$.

```
// Create the 26 ciphertext vectors (P(X),P(XX|X)):
Tot := &+[ F2[i,i] : i in [1..26] ];
VV := [ V2![ F1[i], F2[i,i]/Tot ] | i in [1..26] ];
// Enter a few standard vectors from the digraph frequencies:
vv_E := V2![ 0.1291, 0.0389 ];
vv_L := V2![ 0.0402, 0.1364 ];
vv_S := V2![ 0.0567, 0.0782 ];
vv_T := V2![ 0.0936, 0.5923 ];
vv_O := V2![ 0.0747, 0.0312 ]; // not far from vv_R et al.
vv_P := V2![ 0.0293, 0.0634 ]; // also near vv_F
// Now search for nearby ciphertext probability vectors:
for eps in [0.010,0.005,0.002,0.001] do
    "eps:", eps;
    "E:", [ AZ[i] : i in [1..26] | Norm(VV[i]-vvE) lt eps ];
    "L:", [ AZ[i] : i in [1..26] | Norm(VV[i]-vvL) lt eps ];
    "S:", [ AZ[i] : i in [1..26] | Norm(VV[i]-vvS) lt eps ];
    "T:", [ AZ[i] : i in [1..26] | Norm(VV[i]-vvT) lt eps ];
    "O:", [ AZ[i] : i in [1..26] | Norm(VV[i]-vvO) lt eps ];
    "P:", [ AZ[i] : i in [1..26] | Norm(VV[i]-vvP) lt eps ];
end for;
```

Armed with knowledge (or good guesses) of the ciphertext images of characters like `E` and `L` we try to use these to find further characters which tend to associate with them.

```
// Suppose iL is defines as the value in [1..26] which determines
// the character which is the ciphertext image of the character L.
Tot1 := &+[ F2[j,iL] : j in [1..26] ];
Tot2 := &+[ F2[iL,j] : j in [1..26] ];
VL := [ V3![ F1[j], F2[j,iL]/Tot1, F2[iL,j]/Tot2 ] | j in [1..26] ];
vL_A := V3![ 0.0771, 0.1785, 0.1176 ]; // expected image of "A"
vL_T := V3![ 0.0936, 0.0552, 0.0493 ]; // expected image of "T"
for eps in [0.010,0.005,0.002,0.001] do
    "eps:", eps;
    "A:", [ AZ[i] : i in [1..26] | Norm(VL[i]-vL_A) lt eps ];
    "T:", [ AZ[i] : i in [1..26] | Norm(VL[i]-vL_T) lt eps ];
end for;
```

Note that `H` is immediately identified by its association with `T`, and the vowels can be picked out using association with these characters.

3. (2 pts) [Product cipher] After recognizing the weaknesses of previous ciphers, the cryptosystem suddenly changes, and your supervisor believes that the Melbourne University Junta is now using product cipher, composing substitution and transposition ciphers. Find the substitution key, the transposition key, and the plaintext message.

Solution The main strategy for this ciphertext is to determine the transposition first. We do this using the function `CoincidenceDiscriminant` from the week 5 tutorial, which seems to do better as differentiating the correctly associated pairs.

First we try to determine the period of the transposition by looking for the maximum coincidence discriminant among the two-character decimations $(1, j)$ of period m for varying

m and j in $2, \ldots, m$ – the idea being that for the right m there exists a j such that the characters at positions $(1, j)$ *were* adjacent in the plaintext.

```
eps := 0.005;
for m in [1..50] do
    CTjs := [ Decimation(CT,[1,j],m) : j in [2..m] ];
    printf "%2o: %o\n",
        m, Max([ CoincidenceDiscriminant(DecCT) : DecCT in CTjs ]);
end for;
```

Once we know with reasonable certainty the period, we proceed to search for the positions (i, j) which were adjacent in the plaintext – each identified by a high coincidence discriminant.

```
for i in [1..m] do
    for j in [i+1..m] do
        CD := CoincidenceDiscriminant(Decimation(CT,[i,j],m));
        if CD gt eps then
            printf "(%2o,%2o): %o\n", i, j, CD;
        end if;
    end for;
end for;
```

This leads to a reconstructed sequence $i_1, i_2, \ldots, i_n$ for the transposition giving rise to our ciphertext. We now only need to determine the whether this or $i_n, \ldots, i_2, i_1$ is the true order, then to undo the substitution.

4. (2 pts) [N-time pad cipher] The American Literature group suddenly gets wise to the insecurity of classical cryptosystems, and tries to implement the use of one-time pads. However they make the implementation mistake of reusing the key for multiple ciphertexts, of which you intercept three. Find the key and plaintext messages.

Solution For hints and suggestions, see the week 6 tutorial.

5. (2 pts) [Information theory] Your supervisor presents you with a model cryptosystem, giving the probabilities of each the plaintexts and a specification of the enciphering algorithm. Find the entropy of the plaintext, keyspace, and ciphertext languages, and the conditional entropy of the cryptosystem.

Solution The definitions of entropy $H(\mathcal{M})$ is just the sum over the terms $-P(X)\log(P(X))$ for X in $\mathcal{M}$, and similarly for the ciphertext entropy $H(\mathcal{C})$. For the latter we first need to compute the ciphertext probabilities:

$$P(Y) = \sum_{\substack{K \in \mathcal{K}, X \in \mathcal{M} \\ E_K(X) = Y}} P(K)P(X)$$

where $P(X)$ are given and $P(K) = 1/8$ for all 8 keys. The conditional entropy of the cryptosystem is the expectation of the conditional entropies $H(\mathcal{M}|Y)$, i.e. a sum over all $P(Y)H(\mathcal{M}|Y)$, where $H(\mathcal{M}|Y)$ is the entropy of the conditional probability function $P(X|Y)$ on $\mathcal{M}$.

Individual data for this assignment can be accessed from the course home page. Answers to the assignment should be submitted by Tuesday 27 April 2004, following the directions on the course home page.