Classical Ciphers and Frequency Analysis Examples

By Steven Gordon on Sun, 18/11/2012 - 11:38am

The following are some examples of classical ciphers and letter frequency analysis used in my course on Security and Cryptography [3] at SIIT.

1. Caesar Cipher and Brute Force Attack

Lets start with some ciphertext obtained using a Caesar cipher:

```
dvvkzecfssprkkve
```

There are 26 possible keys with a Caesar cipher and so a brute force attack is easy. The following uses the `crypto` [4] script in a for loop, decrypting the ciphertext with each different key. The key and corresponding plaintext is printed.

```
$ for i in `seq 0 25`; do echo -n "$i "; crypto caesar dec dvvkzecfssprkkve $i ; done ;
0 dvvkzecfssprkkve
1 cuujydberroqjjud
2 bttixcadqqnpiitc
3 asshwbzcppmohhsb
4 zrrgvayboolnggra
5 yqqfuzxannkmffqz
6 xppetywzmmjleepy
7 woodsxvyllikddox
8 vnncrwuxkkhjccnw
9 ummbqvtwjjgibbmv
10 tllapusviifhaalu
11 skkzotruhhegzzkt
12 rjjynsqtggdfyyjs
13 qiixmrpsffcexxir
14 phhwlqoreebdwwhq
15 oggvkpnqddacvvgp
16 nffujompcczbuufo
17 meetinlobbyatten
18 lddshmknaaxzssdm
19 kccrgljmzzwyrrcl
20 jbbqfkilyyvxqqbk
21 iaapejhkxxuwppaj
22 hzzodigjwwtvoozi
23 gyynchfivvsunnyh
24 fxxmbgehuurtmmxg
25 ewwlafdgttqsllwf
```

Which plaintext makes sense? "meet in lobby at ten" (I've inserted the spaces for readability). The 25 other plaintexts look like random collections of English characters - they don't form known English words/phrases. Hence if we can recognise valid plaintext, the brute force attack works.

Just to confirm, lets decrypt using key 17 (and to demonstrate the `crypto` script, also encrypt):

```
$ crypto caesar dec dvvkzecfssprkkve 17
meetinlobbyatten
$ crypto caesar enc meetinlobbyatten 17
dvvkzecfssprkkve
```

2. Letter Frequency Analysis

An alternative to a brute force attack is to be more intelligent. Apply cryptanalysis. We've already assumed the plaintext is English text. English text (and in fact text in any language) has structure. A simple measure of the structure is the frequency at which certain letters occur, e.g. letter 'e' occurs much more frequently on average than the letter 'q'. To demonstrate this lets take an English book [5] and count the occurrences of each letter. Of the 447,145 letters in the book, the following shows the percentage that the top 15 letters make up:

```
$ crypto count letters book1.txt percentsort
447145
 12.29 e
  9.06 t
  8.08 a
  7.80 o
  6.99 i
  6.65 n
  6.62 h
  6.25 s
  5.74 r
  4.27 d
  3.94 l
  3.05 u
  2.72 m
  2.58 w
  2.48 c
```

The letter 'e' is the most frequent (about 12% of all letters in the book), followed by 't', and so on. If similar analysis is performed on other English books, similar statistics can be obtained. There are other statistics about expected plaintext that can be obtained as well. For example, the frequency of digrams (pairs of letters) and trigrams (triples of letters). From the same book:

```
$ crypto count digrams book1.txt percentsort
338145
  3.87 th
  3.59 he
  2.36 in
  2.18 er
  2.03 an
  1.74 ha
  1.73 re
  1.72 ou
  1.55 nd
  1.51 at
  1.38 on
  1.33 ed
  1.32 en
  1.27 it
  1.25 hi
```

```
$ crypto count trigrams book1.txt percentsort
235545
  3.39 the
  1.55 and
  1.21 ing
  0.96 hat
  0.91 you
  0.88 her
  0.83 tha
  0.74 his
  0.68 ere
  0.61 was
  0.54 ver
  0.54 for
  0.51 thi
  0.51 ave
  0.48 ter
```

'th' is the most frequency digram and 'the' the most frequent trigram. These statistics aid in the cryptanalysis of a ciphertext. Consider the general Caesar cipher. The letter 'a' in plaintext always maps to the same letter in ciphertext (the actual letter depends on the key). Similar with all other letters in plaintext. Therefore in plaintext we expect the letter 'e' to be most common (about 12%). If we look at the most common letter in the ciphertext then we may try assuming that that ciphertext letter maps to the letter 'e' in plaintext. Similar with the 2nd most common letter. After several attempts (normally one or two, but much less than all 26 used with brute force) it's easy to find the key.

Below is an example ciphertext [6] obtained by encrypting a paragraph of English text with a Caesar cipher.

```
$ cat phrase1-caesar1.txt
ymnxhtzwxjfnrxytuwtanijdtzbnymijyfnqjipstbqjiljtknrutwyfsyyjhmstqtlnjxfsifuuqnh
fyntsymfyfwjzxjinsymjnsyjwsjyizjytymjgwtfisfyzwjtkymnxknjqiymjhtzwxjhtajwxtsqdx
jqjhyjiytunhxkthzxxnslknwxytsxtrjfiafshjiytunhxnsnsyjwsjyyjhmstqtlnjxjlbnwjqjxx
qfsxrtgnqjnsyjwsjyrzqynhfxyfsiymjsfxjqjhyntstkhzwwjsyfsisjcyljsjwfyntsfuuqnhfyn
tsxfsixjwanhjxjluunuyaatnudtzbnqqqjfwsmtbymjnsyjwsjybtwpxfsimtbxjwanhjxfsifuuqn
hfyntsxfwjuwtanijiytzxjwxtkymjnsyjwsjyymnxpstbqjiljbnqqmjqudtz
```

And the statistics about frequencies of letters from this ciphertext:

```
$ crypto count letters phrase1-caesar1.txt percentsort
552
 13.41 j
  9.78 y
  9.24 s
  8.33 t
  8.33 n
  6.52 x
  6.34 f
  5.07 w
  4.71 q
  4.35 i
  3.80 h
  3.62 u
  3.08 m
```

```
  2.54 z
  1.99 b
```

In the ciphertext the letter 'j' is most common. But we expect letter 'e' to be most common in the plaintext. If we assumed plaintext 'e' mapped to ciphertext 'j' then that implies a key of 5. Lets try to decrypt the entire ciphertext with the key:

```
$ crypto caesar dec phrase1-caesar1.txt 5 file
thiscourseaimstoprovideyouwithdetailedknowledgeofimportanttechnologiesandapplic
ationthatareusedintheinternetduetothebroadnatureofthisfieldthecoursecoversonlys
electedtopicsfocussingfirstonsomeadvancedtopicsininternettechnologiesegwireless
lansmobileinternetmulticastandthenaselectionofcurrentandnextgenerationapplicati
onsandservicesegppiptvvoipyouwilllearnhowtheinternetworksandhowservicesandappli
cationsareprovidedtousersoftheinternetthisknowledgewillhelpyouinthedesignandman
agementofcomputernetworksaswellasdevelopmentandexecutionofinternetapplications
```

We've found the plaintext [7] after just 1 decryption! If using key 5 did NOT produce readable plaintext then we could have guessed the key based on the second most common letter and so on.

3. Frequency Analysis of Monoalphabetic Cipher

The Caesar cipher is subject to both brute force and a frequency analysis attack. But what about ciphers with larger key spaces? A monoalphabetic cipher using 26 English characters has 26! possible keys (that is, more than 10^{26}). A brute force attack is no longer feasible. Lets see how frequency analysis still makes breaking a monoalphabetic cipher easy.

Below is the ciphertext [8] output from a monoalphabetic cipher:

```
$ cat phrase3-mono1.txt
ziolegxkltqodlzgofzkgrxetngxzgzithkofeohstlqfrzteifojxtlgyltexkofuegdhxztklqfregd
hxztkftzvgkalvoziygexlgfofztkftzltexkoznzitegxkltoltyytezoctsnlhsozofzgzvghqkzlyo
klzofzkgrxeofuzitzitgkngyeknhzgukqhinofesxrofuigvdqfnesqlloeqsqfrhghxsqkqsugkozid
lvgkaturtlklqrouozqsloufqzxktlqfrltegfrhkgcorofurtzqoslgyktqsofztkftzltexkoznhkgz
gegslqsugkozidlqfrziktqzltuohltecokxltlyoktvqsslitfetngxvossstqkfwgzizitgktzoeqsq
lhtezlgyegdhxztkqfrftzvgkaltexkoznqlvtssqligvziqzzitgknolqhhsotrofzitofztkftzziol
afgvstrutvossitshngxofrtloufofuqfrrtctsghofultexktqhhsoeqzogflqfrftzvgkahkgzgegsl
qlvtssqlwxosrofultexktftzvgkal
```

Lets count the frequency of letters and digrams of the ciphertext:

```
$ crypto count letters phrase3-mono1.txt  percentsort
597
 10.39 t
  8.88 z
  8.38 o
  7.87 l
  7.71 g
  7.04 k
  7.04 f
  5.86 q
  5.36 s
  4.86 e
  3.85 x
```

```
  3.69 h
  3.52 r
  3.52 i
  2.68 u
$ crypto count digrams phrase3-mono1.txt  percentsort
596
  2.85 of
  2.52 zi
  2.18 lt
  2.01 te
  1.68 tz
  1.68 gk
  1.68 fr
  1.51 xk
  1.51 qf
  1.51 lq
  1.51 it
  1.34 oz
  1.34 eg
  1.17 zt
  1.17 qs
```

Now compare to the most frequent letters and digrams from the example English book above. (In the following I will write the plaintext letters in UPPERCASE)

Looking at the four most frequent letters, in the ciphertext we have t, z, o and l while we expect E, T, A and O. In the ciphertext the letter t occurs about 2% more frequently than the second most frequent. So initially lets assume plaintext E maps to ciphertext t, or in short E-->t. Now lets look at the ciphertext, but replacing each occurrence of t with E:

```
ziolegxklEqodlzgofzkgrxeEngxzgziEhkofeohsElqfrzEeifojxElgylEexkofuegdhxzEklqfregd
hxzEkfEzvgkalvoziygexlgfofzEkfEzlEexkoznziEegxklEolEyyEezocEsnlhsozofzgzvghqkzlyo
klzofzkgrxeofuziEziEgkngyeknhzgukqhinofesxrofuigvdqfnesqlloeqsqfrhghxsqkqsugkozid
lvgkaEurElklqrouozqsloufqzxkElqfrlEegfrhkgcorofurEzqoslgykEqsofzEkfEzlEexkoznhkgz
gegslqsugkozidlqfrzikEqzlEuohlEecokxlElyokEvqssliEfeEngxvosssEqkfwgziziEgkEzoeqsq
lhEezlgyegdhxzEkqfrfEzvgkalEexkoznqlvEssqligvziqzziEgknolqhhsoErofziEofzEkfEzziol
afgvsEruEvossiEshngxofrEloufofuqfrrEcEsghofulEexkEqhhsoeqzogflqfrfEzvgkahkgzgegsl
qlvEssqlwxosrofulEexkEfEzvgkal
```

That doesn't help much - it is impossible to make out any English words yet. Lets continue with the second most frequent letter and assume T-->z.

```
TiolegxklEqodlTgofTkgrxeEngxTgTiEhkofeohsElqfrTEeifojxElgylEexkofuegdhxTEklqfregd
hxTEkfETvgkalvoTiygexlgfofTEkfETlEexkoTnTiEegxklEolEyyEeTocEsnlhsoTofTgTvghqkTlyo
klTofTkgrxeofuTiETiEgkngyeknhTgukqhinofesxrofuigvdqfnesqlloeqsqfrhghxsqkqsugkoTid
lvgkaEurElklqrouoTqsloufqTxkElqfrlEegfrhkgcorofurETqoslgykEqsofTEkfETlEexkoTnhkgT
gegslqsugkoTidlqfrTikEqTlEuohlEecokxlElyokEvqssliEfeEngxvosssEqkfwgTiTiEgkEToeqsq
lhEeTlgyegdhxTEkqfrfETvgkalEexkoTnqlvEssqligvTiqTTiEgknolqhhsoErofTiEofTEkfETTiol
afgvsEruEvossiEshngxofrEloufofuqfrrEcEsghofulEexkEqhhsoeqTogflqfrfETvgkahkgTgegsl
qlvEssqlwxosrofulEexkEfETvgkal
```

Still difficult to identify words. Lets try the two most frequent digrams. In the ciphertext we have 'of' and 'zi' while we expect 'IN' and 'TH'. We already have assumed t-->Z, so no lets also try: o-->I, f-->N and i-->H.

```
THIlegxklEqIdlTgINTkgrxeEngxTgTHEhkINeIhsElqNrTEeHNIjxElgylEexkINuegdhxTEklqNregd
hxTEkNETvgkalvITHygexlgNINTEkNETlEexkITnTHEegxklEIlEyyEeTIcEsnlhsITINTgTvghqkTlyI
klTINTkgrxeINuTHETHEgkngyeknhTgukqhHnINesxrINuHgvdqNnesqllIeqsqNrhghxsqkqsugkITHd
lvgkaEurElklqrIuITqslIuNqTxkElqNrlEegNrhkgcIrINurETqIslgykEqsINTEkNETlEexkITnhkgT
gegslqsugkITHdlqNrTHkEqTlEuIhlEecIkxlElyIkEvqsslHENeEngxvIsssEqkNwgTHTHEgkETIeqsq
lhEeTlgyegdhxTEkqNrNETvgkalEexkITnqlvEssqlHgvTHqTTHEgknIlqhhsIErINTHEINTEkNETTHIl
aNgvsEruEvIssHEshngxINrElIuNINuqNrrEcEsghINulEexkEqhhsIeqTIgNlqNrNETvgkahkgTgegsl
qlvEssqlwxIsrINulEexkENETvgkal
```

Can we make out any words? On the second line (and elsehwere) INTEkNET suggests k-->R (giving INTERNET). The first word is THI... Possibly THIS? Lets also try l-->S.

```
THISegxRSEqIdSTgINTRgrxeEngxTgTHEhRINeIhsESqNrTEeHNIjxESgySEexRINuegdhxTERSqNregd
hxTERNETvgRaSvITHygexSgNINTERNETSEexRITnTHEegxRSEISEyyEeTIcEsnShsITINTgTvghqRTSyI
RSTINTRgrxeINuTHETHEgRngyeRnhTguRqhHnINesxrINuHgvdqNnesqSSIeqsqNrhghxsqRqsugRITHd
SvgRaEurESRSqrIuITqsSIuNqTxRESqNrSEegNrhRgcIrINurETqIsSgyREqsINTERNETSEexRITnhRgT
gegsSqsugRITHdSqNrTHREqTSEuIhSEecIRxSESyIREvqssSHENeEngxvIsssEqRNwgTHTHEgRETIeqsq
ShEeTSgyegdhxTERqNrNETvgRaSEexRITnqSvEssqSHgvTHqTTHEgRnISqhhsIErINTHEINTERNETTHIS
aNgvsEruEvIssHEshngxINrESIuNINuqNrrEcEsghINuSEexREqhhsIeqTIgNSqNrNETvgRahRgTgegsS
qSvEssqSwxIsrINuSEexRENETvgRaS
```

Now returning to the most frequent letters. We have already covered E, T, I, N and H. The other two frequeny letters are A and O. In the ciphertext we have covered t, z, o, f and i. So its likely that A and O map to one of l, g, or k. Try different combinations in the text so far and see if any words make sense. Eventually I tried g-->O.

```
THISeOxRSEqIdSTOINTROrxeEnOxTOTHEhRINeIhsESqNrTEeHNIjxESOySEexRINueOdhxTERSqNreOd
hxTERNETvORaSvITHyOexSONINTERNETSEexRITnTHEeOxRSEISEyyEeTIcEsnShsITINTOTvOhqRTSyI
RSTINTROrxeINuTHETHEORnOyeRnhTOuRqhHnINesxrINuHOvdqNnesqSSIeqsqNrhOhxsqRqsuORITHd
SvORaEurESRSqrIuITqsSIuNqTxRESqNrSEeONrhROcIrINurETqIsSOyREqsINTERNETSEexRITnhROT
OeOsSqsuORITHdSqNrTHREqTSEuIhSEecIRxSESyIREvqssSHENeEnOxvIsssEqRNwOTHTHEORETIeqsq
ShEeTSOyeOdhxTERqNrNETvORaSEexRITnqSvEssqSHOvTHqTTHEORnISqhhsIErINTHEINTERNETTHIS
aNOvsEruEvIssHEshnOxINrESIuNINuqNrrEcEsOhINuSEexREqhhsIeqTIONSqNrNETvORahROTOeOsS
qSvEssqSwxIsrINuSEexRENETvORaS
```

Near the start we have INTROrxe. That suggests INTRODUC... (as in introduction, introduce, introducing). Now lets assume r-->D, x-->U and e-->C.

```
THISCOURSEqIdSTOINTRODUCEnOUTOTHEhRINCIhsESqNDTECHNIjUESOySECURINuCOdhUTERSqNDCOd
hUTERNETvORaSvITHyOCUSONINTERNETSECURITnTHECOURSEISEyyECTIcEsnShsITINTOTvOhqRTSyI
RSTINTRODUCINuTHETHEORnOyCRnhTOuRqhHnINCsUDINuHOvdqNnCsqSSICqsqNDhOhUsqRqsuORITHd
SvORaEuDESRSqDIuITqsSIuNqTURESqNDSECONDhROcIDINuDETqIsSOyREqsINTERNETSECURITnhROT
OCOsSqsuORITHdSqNDTHREqTSEuIhSECcIRUSESyIREvqssSHENCEnOUvIsssEqRNwOTHTHEORETICqsq
ShECTSOyCOdhUTERqNDNETvORaSECURITnqSvEssqSHOvTHqTTHEORnISqhhsIEDINTHEINTERNETTHIS
aNOvsEDuEvIssHEshnOUINDESIuNINuqNDDEcEsOhINuSECUREqhhsICqTIONSqNDNETvORahROTOCOsS
qSvEssqSwUIsDINuSECURENETvORaS
```

THIS COURSE qIdSTO INTRODUCE nOU TO THE hRINCIhsES .... Try n-->Y, h-->P and s-->L.

```
THISCOURSEqIdSTOINTRODUCEYOUTOTHEPRINCIPLESqNDTECHNIjUESOySECURINuCOdPUTERSqNDCOd
PUTERNETvORaSvITHyOCUSONINTERNETSECURITYTHECOURSEISEyyECTIcELYSPLITINTOTvOPqRTSyI
```

```
RSTINTRODUCINuTHETHEORYOyCRYPTOuRqPHYINCLUDINuHOvdqNYCLqSSICqLqNDPOPULqRqLuORITHd
SvORaEuDESRSqDIuITqLSIuNqTURESqNDSECONDPROcIDINuDETqILSOyREqLINTERNETSECURITYPROT
OCOLSqLuORITHdSqNDTHREqTSEuIPSECcIRUSESyIREvqLLSHENCEYOUvILLLEqRNwOTHTHEORETICqLq
SPECTSOyCOdPUTERqNDNETvORaSECURITYqSvELLqSHOvTHqTTHEORYISqPPLIEDINTHEINTERNETTHIS
aNOvLEDuEvILLHELPYOUINDESIuNINuqNDDEcELOPINuSECUREqPPLICqTIONSqNDNETvORaPROTOCOLS
qSvELLqSwUILDINuSECURENETvORaS
```

THIS COURSE qIdS TO INTRODUCE YOU TO THE PRINCIPLES qND TECHNIjUES
Oy SECURINu COdPUTERS ...

With a few more steps it is easy to find the remaining letters and the resulting plaintext:

```
THISCOURSEAIMSTOINTRODUCEYOUTOTHEPRINCIPLESANDTECHNIQUESOFSECURINGCOMPUTERSANDCOM
PUTERNETWORKSWITHFOCUSONINTERNETSECURITYTHECOURSEISEFFECTIVELYSPLITINTOTWOPARTSFI
RSTINTRODUCINGTHETHEORYOFCRYPTOGRAPHYINCLUDINGHOWMANYCLASSICALANDPOPULARALGORITHM
SWORKEGDESRSADIGITALSIGNATURESANDSECONDPROVIDINGDETAILSOFREALINTERNETSECURITYPROT
OCOLSALGORITHMSANDTHREATSEGIPSECVIRUSESFIREWALLSHENCEYOUWILLLEARNBOTHTHEORETICALA
SPECTSOFCOMPUTERANDNETWORKSECURITYASWELLASHOWTHATTHEORYISAPPLIEDINTHEINTERNETTHIS
KNOWLEDGEWILLHELPYOUINDESIGNINGANDDEVELOPINGSECUREAPPLICATIONSANDNETWORKPROTOCOLS
ASWELLASBUILDINGSECURENETWORKS
```

So we have demonstrated that this monoalphabetic cipher has too many keys to apply a brute force attack (> 10^{26}), with frequency analysis and some trial-and-error we could manually break the cipher and obtain the plaintext [9] in less than 1 hour. In fact these steps can be automated and solved almost instantly with a computer.

Content: Articles [10]
Topic: Security [11]

Source URL: http://sandilands.info/sgordon/classical-ciphers-frequency-analysis-examples

Links:
[1] http://sandilands.info/sgordon/classical-ciphers-frequency-analysis-examples
[2] http://sandilands.info/sgordon/user/2
[3] http://ict.siit.tu.ac.th/~sgordon/css322/
[4] http://sandilands.info/sgordon/doc/security/crypto.txt
[5] http://www.gutenberg.org/cache/epub/1661/pg1661.txt
[6] http://sandilands.info/sgordon/doc/security/phrase1-caesar1.txt
[7] http://sandilands.info/sgordon/doc/security/phrase1.txt
[8] http://sandilands.info/sgordon/doc/security/phrase3-mono1.txt
[9] http://sandilands.info/sgordon/doc/security/phrase3.txt
[10] http://sandilands.info/sgordon/taxonomy/term/144
[11] http://sandilands.info/sgordon/taxonomy/term/116