Lecture 4: Hill Cipher, Stream Ciphers and the One Time Pad

TTM4135

Relates to Stallings Chapter 3

Spring Semester, 2025

Motivation

- The Hill Cipher is a mathematically defined encryption scheme
- The Hill Cipher illustrates the weakness of linearity in cipher design
- Stream ciphers are constructed from (pseudo-)random number generators.
- The One Time Pad is an unbreakable stream cipher

Outline

Hill cipher

- Lester S. Hill was an American mathematician who published his cipher in 1929.
- The Hill cipher is an example of a *polygram cipher* (also called *polygraphic cipher*). This is a simple substitution cipher on an extended alphabet consisting of multiple characters. The simplest example is digram substitution in which the alphabet consists of all pairs of characters.
- The major weakness of the Hill cipher is that it is linear. This makes known plaintext attacks easy.

Definition of Hill cipher

The Hill cipher performs a linear transformation on d plaintext characters to get d ciphertext characters.

- Encryption involves multiplying a $d \times d$ matrix K by the block of plaintext P.
- Decryption involves multiplying the matrix K^{-1} by the block of the ciphertext C.

$$\begin{array}{ll} \text{Encryption:} & C = KP \\ \text{Decryption:} & P = K^{-1}C \end{array}$$

Encryption example

- We choose $d = 2$ so that encryption takes digrams as input and output blocks
- Write each plaintext pair as a column vector and encode letters as numbers
- Suppose the first pair for encryption is (EG). Then since E=4 and G=6 in our encoding this is represented as $\begin{pmatrix} 4 \\ 6 \end{pmatrix}$
- If there are insufficient letters to fill a block then it must be padded. This can be done with an uncommon letter such as Z
- In these examples the space character is omitted and all computations take place modulo 26

Encrypting and decrypting

$$d = 2, \quad K = \begin{pmatrix} 4 & 5 \\ 1 & 7 \end{pmatrix}, \quad K^{-1} = \begin{pmatrix} 15 & 19 \\ 9 & 16 \end{pmatrix}$$

$$Plaintext : (BC) \rightarrow P = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

$$Encryption : C = KP = \begin{pmatrix} 4 & 5 \\ 1 & 7 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{pmatrix} 14 \\ 15 \end{pmatrix} \rightarrow (OP)$$

$$Decryption : P = K^{-1}C = \begin{pmatrix} 15 & 19 \\ 9 & 16 \end{pmatrix} \begin{pmatrix} 14 \\ 15 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

Cryptanalysis of Hill cipher

- Known plaintext attack is possible given d plaintext-ciphertext matching blocks.
- Suppose we are given blocks (column vectors) P_i, C_i for $i = 0, 1, .., d-1$.
  1. Let $C = [C_0\, C_1\, \ldots\, C_{d-1}]$. Let $P = [P_0\, P_1\, \ldots\, P_{d-1}]$.
  2. Solve $C = KP$ for K.
  3. $P = K^{-1}C$.

Cryptanalysis example

- Suppose that we know $d = 2$.
- Ciphertext: ZIKPWIXPTFUTVPVRQBUTVPJLKB
- Known plaintext is first two blocks (digrams): THER

Step 1 - encode plaintext and ciphertext

$$P_0 = (TH) = \begin{pmatrix} 19 \\ 7 \end{pmatrix},\ P_1 = (ER) = \begin{pmatrix} 4 \\ 17 \end{pmatrix}$$

$$C_0 = (ZI) = \begin{pmatrix} 25 \\ 8 \end{pmatrix},\ C_1 = (KP) = \begin{pmatrix} 10 \\ 15 \end{pmatrix}$$

$$\rightarrow P = [P_0\, P_1] = \begin{pmatrix} 19 & 4 \\ 7 & 17 \end{pmatrix}$$

$$\rightarrow C = [C_0\, C_1] = \begin{pmatrix} 25 & 10 \\ 8 & 15 \end{pmatrix}$$

Step 2 - recover encryption matrix K

We have $C = KP$. Let $K = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$. Then:

$$\begin{pmatrix} 25 & 10 \\ 8 & 15 \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} 19 & 4 \\ 7 & 17 \end{pmatrix}.$$

So

$$\begin{aligned} \begin{pmatrix} a & b \\ c & d \end{pmatrix} &= \begin{pmatrix} 25 & 10 \\ 8 & 15 \end{pmatrix} \begin{pmatrix} 19 & 4 \\ 7 & 17 \end{pmatrix}^{-1} \\ &= \begin{pmatrix} 25 & 10 \\ 8 & 15 \end{pmatrix} \begin{pmatrix} 25 & 14 \\ 5 & 5 \end{pmatrix} \\ &= \begin{pmatrix} 25 & 10 \\ 15 & 5 \end{pmatrix} \end{aligned}$$

.

Step 3 - compute K^{-1} and decrypt ciphertext

$$\rightarrow K = \begin{pmatrix} 25 & 10 \\ 15 & 5 \end{pmatrix} \rightarrow \cdots \rightarrow K^{-1} = \begin{pmatrix} 5 & 16 \\ 11 & 25 \end{pmatrix}$$

$$\begin{aligned} C &= \begin{pmatrix} W & X & T & U & T & \ldots \\ I & P & F & T & V & \ldots \end{pmatrix} \\ &= \begin{pmatrix} 22 & 23 & 17 & 20 & 17 & \ldots \\ 8 & 15 & 5 & 17 & 21 & \ldots \end{pmatrix} \end{aligned}$$

$$P = K^{-1}C$$

Plaintext: THEREARETWOTHINGSTOTHINKOF

Notes on cryptanalysis of Hill cipher

- In known plaintext attacks the equations may not be fully determined. In this case Step 2 will fail because the matrix will not be invertible. Further plaintext/ciphertext character can be examined.
- Ciphertext only attack follows known plaintext attack with the added task of finding probable blocks of matching plaintext and ciphertext. For example, when $d = 2$ the frequency distribution of non-overlapping pairs of ciphertext characters can be compared with the distribution of pairs of plaintext characters.

Stream ciphers

- Stream ciphers are characterised by the generation of a *keystream* of any required length
- Each element of the keystream is used successively to encrypt one or more ciphertext characters
- Stream ciphers are usually symmetric key ciphers: sender and receiver share the same key and can generate the same keystream given the same initialisation value
- The keystream must have good randomness properties

Synchronous stream ciphers

- In the simplest kind of stream cipher the keystream is generated *independently* of the plaintext. In this case the cipher is called a *synchronous* stream cipher.
- Both sender and receiver need to generate the same keystream and synchronise on its usage
- The Vigenère cipher can be seen as a (periodic) synchronous stream cipher where each shift is defined by a key letter
- Later we will see how to use modern block ciphers to generate a keystream

Binary synchronous stream cipher

For each time interval t each of the following are defined:

- a binary sequence $s(t)$ called the *keystream*;
- a binary plaintext $p(t)$;
- a binary ciphertext $c(t)$.

$$\begin{array}{ll} \text{Encryption:} & c(t) = p(t) \oplus s(t) \\ \text{Decryption:} & p(t) = c(t) \oplus s(t) \end{array}$$

One-time pad

- Often attributed to Vernam who made a one-time pad machine using teletype machinery in 1917. Earlier historical uses are known.
- The key is a truly random sequence of characters, all of them independently generated
- Each character in the key is used *one time* only
- The alphabet can be of any length, but usually is either a natural language alphabet or simply the binary alphabet $\{0, 1\}$.
- The binary one time pad is a (non-periodic) binary synchronous stream cipher.
- The one-time pad provides perfect secrecy.

Shannon's definition of perfect secrecy

- To define perfect secrecy, consider a cipher with message set $\{M_1, M_2, \ldots, M_k\}$ and ciphertext set $\{C_1, C_2, \ldots, C_l\}$.
- Then $\Pr(M_i|C_j)$ is the probability that message M_i was encrypted given that ciphertext C_j was observed.
- Note that in most cases the messages M_i will *not* be equally likely.
- We say that the cipher achieves *perfect secrecy* if for all messages M_i and ciphertexts C_j we have

$$\Pr(M_i|C_j) = \Pr(M_i)$$

One time pad using Roman alphabet

- Plaintext characters: $p_1, \ldots, p_r$
- Ciphertext characters: $c_1, \ldots, c_r$
- Keystream: random characters $k_1, \ldots, k_r$
- Encryption:
$$c_i = (p_i + k_i) \bmod 26$$
- Decryption:
$$p_i = (c_i - k_i) \bmod 26$$
- Resulting ciphertext is modulo 26 addition of the plaintext and keystream sequences.

Why the one time pad provides perfect secrecy

- Suppose a particular ciphertext C_j is observed.
- Any message could have been sent depending on the choice of key.
- The probability that message M_i was sent given that C_j is observed is the probability that M_i is chosen, weighted by the probability that the right key was chosen.
- Since each key is chosen with equal probability, the conditional probability $\Pr(M_i|C_j)$ is simply $\Pr(M_i)$.

Example

Plaintext:	HELLO
Keystream:	EZABD
Ciphertext:	LDLMR

- Note that given the ciphertext LDLMR the plaintext can be *any* 5-letter message.

Real one-time pads used by spies in 1960s

Vernam (binary) one time pad

- Plaintext is binary sequence: $b_1, b_2, \ldots, b_r$
- Keystream is random binary sequence: $k_1, k_2, \ldots, k_r$
- Ciphertext is binary sequence: $c_1, c_2, \ldots, c_r$
- Encryption: $c_i \equiv p_i \oplus k_i$
- Decryption: $p_i \equiv c_i \oplus k_i$
- Keystream is same length as plaintext
- Provides perfect secrecy since any ciphertext is equally possible given the plaintext
- Encryption and decryption are identical processes

One-time pad properties

- Shannon showed that any cipher with perfect secrecy *must* have as many keys as there are messages.
- In this sense the one-time pad is the only unbreakable cipher.
- Practical usage is possible for pre-assigned communications between fixed parties.
- Main problem with one time pad as a general tool is how to deal with key management of completely random keys.
- Key generation, key transportation, key synchronization, key destruction are all problematic since the keys are so large.
- In Caesar cipher, the key legth was one integer between $\{0, \ldots, 26\}$ ($\approx$ 5 bits). Now, the key is the length of the message.

Key management issues for one time pad

- How to generate completely random keys?
- How to transport random keys between sender and receiver?
- How to synchronise on usage of keys?
- How to destroy keys after use?

Visual cryptography

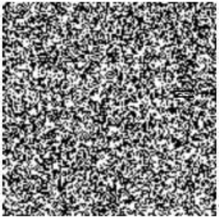
- A fun application of the one time pad is *visual cryptography* which splits an image into two *shares*
- Decryption works by *overlaying* the two shared images
- First proposed by Naor and Shamir in 1994
- We consider the simplest case of monochrome images with black or white pixels — many generalisations are possible
- Each share reveals no information about the image - this is unconditional security as in the one time pad

Two-time Pad

Message:

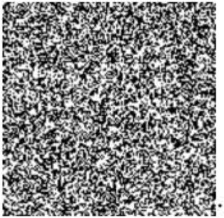
$\oplus$

OTP key:



$=$

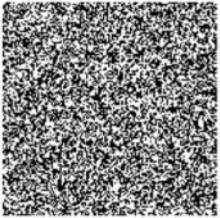
Ciphertext:



Two-time Pad

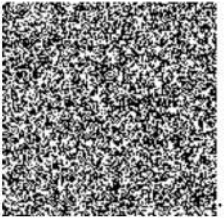
Message:

$\oplus$

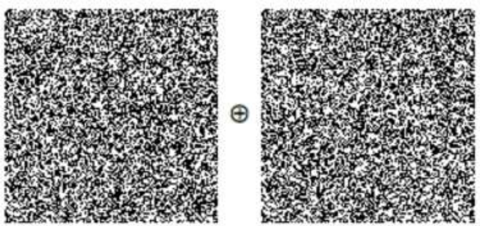


Ciphertext:

$=$



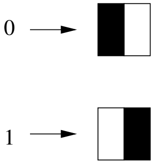
Two-time Pad



Two-time Pad

Encrypting in visual cryptography

- To encrypt an image I, first generate a one time pad P (random string of bits) with length equal to the number of pixels of I
- Generate an image share S_1 by replacing each bit in P using the sub-pixel patterns shown
- Generate the other image share S_2 with pixels as follows:
  - the same as S_1 for all the white pixels of I
  - the opposite (other sub-pixel pattern) of S_1 for all the black pixels of I



Decrypting in visual cryptography

- To reveal the hidden image the two shares are overlayed
- Each black pixel of I is black in the overlay
- Each white pixel of I is half white in the overlay