Introduction to Encryption.

Lecture 1.

Chapter 1 of the textbook is an interesting historical account, and worth a read. However the history of the subject is not the focus of this course, so we begin with Chapter 2.

A remark on organization of material:
We'll learn a handful of "ancient" cryptosystems up front, and then learn how to attack them later. (This, as opposed to the approach of presenting a new system, its weaknesses, and then developing a tougher system in response to those weaknesses).

Our focus will be on valid cryptosystems in history.
We call a system valid if:

① ∘ Messages are easy to encrypt

② ∘ Messages are easy to send

③ ∘ Messages are easy to read if you are the intended recipient

④ ∘ Messages are hard to read if you are not the intended recipient.

⑤ ∘ It should be easy to verify that the message comes from the correct source.

As a comical example from ancient Greek times that we <u>won't</u> discuss: Slave-barber system. It satisfies ① - ③ (though it fails ① if you need to encrypt a message quickly, since hair grows <u>slowly</u>), and definitely fails ④ and ⑤.

To describe our first 'valid' system, we introduce terminology:

- Plaintext - Any message we wish to send
- Ciphertext - The output of our encryption scheme, and what we'll send ~~to~~ the recipient.

<u>First cipher</u>: Substitution alphabet cipher.

This is the process of taking your plaintext and replacing each letter with a different (uniquely chosen) letter. More formally:

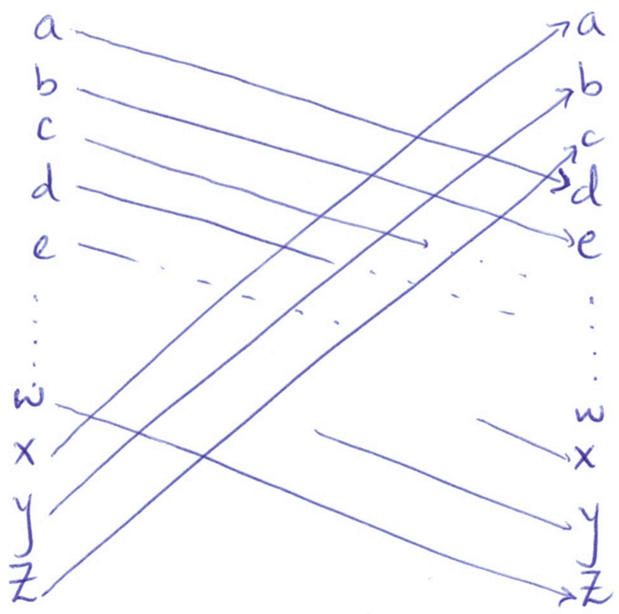
You choose a 1-1 and onto function

$$f : \{a, b, c, \overset{A}{''}\ldots, y, z\} \longrightarrow \{a, b, c, \overset{A}{''}\ldots, y, z\}.$$

Then you replace each letter with f applied to that letter. E.g.

"crypto" becomes "$f(c)f(r)f(y)f(p)f(t)f(o)$"

So if the function f is "shift by 3" then it looks like:



and "crypto" becomes "fubswr".

↑
plaintext

One benefit of this is that there are many ways of encoding plaintext — one for each ~~function~~ 1-1 and onto function $f : A \longrightarrow A$ (here, A is a set with 26 elements — the alphabet).

How many such functions are there?

- There are 26 choices for $f(a)$
- Having decided upon $f(a)$, there are 25 choices for $f(b)$
- Having decided on $f(a)$ and $f(b)$, there are 24 choices for $f(c)$

⋮

and so on.

Overall, we get

$$26! = 26 \cdot 25 \cdot 24 \cdot 23 \cdot \ldots \cdot 2 \cdot 1$$

ways of encrypting a message. Roughly

$$26! \approx 4 \times 10^{26}.$$

Benefit: Anyone trying to decode a message by "testing all functions" will be thwarted, **but** you end up having to (therefore) find a secure way to send your agreed-upon function $f: A \longrightarrow A$ to the intended recipient of the message.

Caesar Cipher: The substitution cipher using the agreed-upon function "shift by 3" is commonly called the Caesar cipher. The easiest way to describe $f: A \longrightarrow A$ is using numbers, not letters.

Make the identification

$$\begin{array}{ccc} a & \longleftrightarrow & 0 \\ b & \longleftrightarrow & 1 \\ c & \longleftrightarrow & 2 \\ & \vdots & \\ y & \longleftrightarrow & 24 \\ z & \longleftrightarrow & 25, \end{array}$$

then f can be written as

$f(x) = x + 3$,

taking care to subtract 26 if $x = 23, 24$ or 25, so that it "loops around" correctly.

Then to decrypt the ciphertext, a recipient need only use the inverse function

$$f^{-1}(x) = x - 3$$

again taking care to add 26 when $x = 0, 1, 2$, so that it "loops around" correctly as before.

With this explanation, it's pretty clear there are 25 total "shifting" functions, so there are 25 possible Caesar ciphers.

To introduce this notion correctly, we need:

Modular arithmetic:

Given two integers x and y, we say that x and y are equivalent modulo m (m some positive integer) if $x - y$ is a multiple of m. We write

$$x \equiv y \bmod m.$$

Every integer x is equivalent to some y in $\{0, 1, \dots, m-1\}$, so we will often replace an integer x with its representative in $\{0, 1, \dots, m-1\}$ when working mod m.

For example, in order to "make sense" of the shift function $f(x) = x+3$ we had to subtract 26 whenever $x+3 \geq 26$. Instead, we could write:

$$f(x) = x+3 \mod 26$$

understanding that "mod 26" means to take the result $(x+3)$ and replace it with a number y in $\{0, \ldots, 25\}$ that is congruent to it.

Example: Use modular arithmetic to encrypt "Adam Clay" with a shift of 21.

Plaintext:	A	d	a	m	C	l	a	y
Numerical:	0	3	0	12	2	11	0	24
+21 :	21	24	21	33	23	32	21	45
mod 26 :	21	24	21	7	23	6	21	19
Ciphertext:	V	y	v	h	x	g	v	t

To decrypt this, we require the recipient to know the secret key of 21, whereupon the do the same steps "in reverse":

Cipher text:	v	y	v	h	x	g	v	t
numerical:	21	24	21	7	23	6	21	19
-21:	0	3	0	-14	2	-15	0	-2
mod 26:	0	3	0	12	2	11	0	24
plain-text:	A	d	a	m	C	l	a	y.

In summary:

- A Caesar cipher is a special case of a substitution cipher. There are $\sim 4 \times 10^{26}$ substitution ciphers, but only 25 Caesar ciphers.
- Caesar ciphers are good because the intended reciptent only needs to know 1 number to decode.
- They're bad because a person who is not the intended recipient can easily try all 25 possibilities.

To make a new, number-based cipher that's more complicated, we need some number theory:

Definitions : Factors, divisors, units, composites, primes, relatively prime:

① Factors and divisors:

If x and y are positive integers, we say that "x divides y" or "x is a factor of y" and write $x|y$ if $y = dx$ for some integer d.

② Nontrivial and proper divisors:

- ~~We~~ If $x|y$, we say x is a divisor of y.
- If $x|y$ and $x<y$, we say x is a proper divisor
- If $x|y$ and $1<x$ we say x is a nontrivial divisor.

③ Prime, composite, unit:

- A positive integer $x>1$ is called prime if its only proper divisor is 1.
- If $x>1$ is not prime, it is composite.
- If $x=1$ it is called a unit.

④ Greatest common divisor.

The gcd of x and y is the largest integer d with $d|x$ and $d|y$.

5 Relatively prime

Two integers x and y are relatively prime if $\gcd(x, y) = 1$.

Lecture 2

Example: $\gcd(30, 12) = 6$

$\gcd(15, 18) = 3$

$\gcd(16, 27) = 1$ so 16 and 27 are relatively prime.

One way of working through computations of gcd's, etc, is to use the following:

Fundamental Theorem of Arithmetic:
Every positive integer can be uniquely expressed as a product of powers of primes.

So revisiting our examples:

Example: $\gcd(30, 12) = ?$

$$30 = 2 \cdot 3 \cdot 5 = 2 \cdot 3 \cdot 5$$
$$12 = 2^2 \cdot 3 = 2 \cdot 2 \cdot 3 \cdot 5$$

This is the largest collection of common factors, so $\gcd = 2 \cdot 3 = 6$.

$\gcd(15, 18) = ?$

$$15 = 3 \cdot 5$$
$$18 = 3 \cdot 6$$

so $\gcd = 3$

Later we will learn an efficient way of computing gcd's, this is good enough for now.

Affine ciphers: §2.6

The affine ciphers are another special case of substitution ciphers, like the Caesar cipher. We'll investigate the functions (affine functions)

$$f(x) = ax + b \mod 26$$

for various values of a, b and try to figure out when they are 1-1 and onto.

One way to approach this problem:

We already know that "shift by b" is 1-1 and onto. So if we write

$$g_b : A \longrightarrow A \quad (A \text{ for "alphabet"})$$

for the function $g_b(x) = x + b \mod 26$

and write f_a for the function

$$f_a(x) = ax,$$

then the affine function $f(x) = ax + b$ is

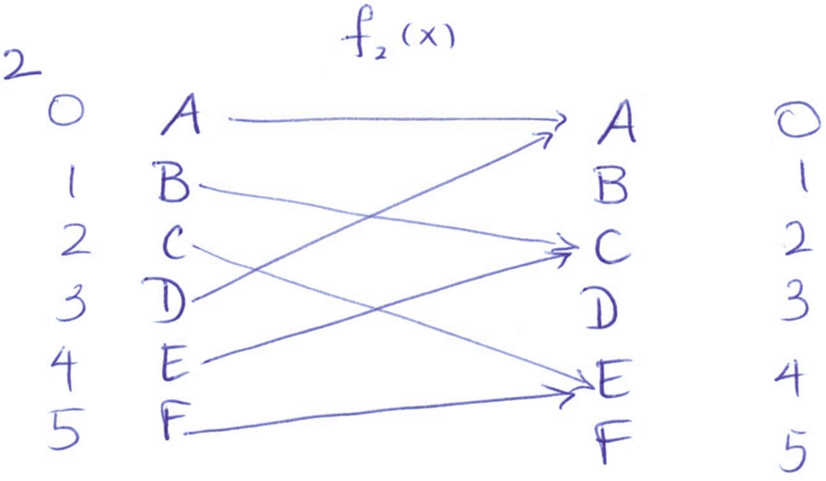
$$f(x) = g_b(f_a(x)) = g_b \circ f_a(x).$$

To prove that the composition $f(x)$ is 1-1 and onto, we only need each of g_b and f_a

to be 1-1 and onto. Since we already know this is true for g_b, (for every value of b) we need to figure out when f_a is 1-1 and onto.

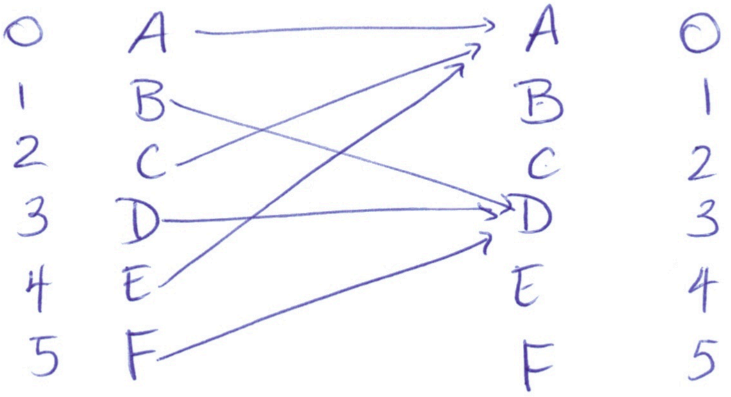
Example: Imagine we have a 6-letter alphabet. Then $f_a(x)$ for various values of "a" would yield the following functions (ie $f_a(x) = ax \bmod 6$ is one of the following)

① $a = 2$



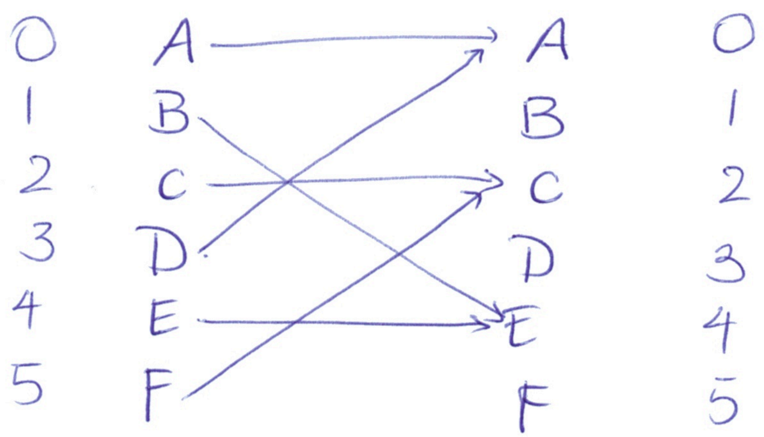
fails to be 1-1 and onto.

② $a = 3$



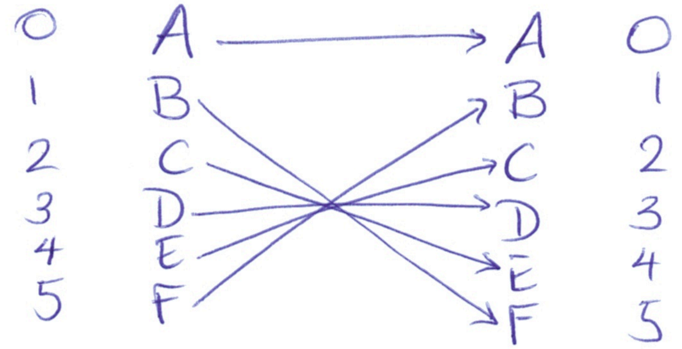
fails to be 1-1 and onto

③ $a = 4$



fails to be 1-1 onto.

④ $a = 5$



works!

Fact, to be revisited later:

The function $f_a(x) = ax \pmod{n}$ is 1-1 and onto (ie it has an inverse) if and only if $\gcd(a, n) = 1$. In all other cases, it fails, and the image of the function is simply all multiples of $\gcd(a, n)$.

Affine ciphers: If $a, b \in \{0, 1, \dots, 25\}$ and $\gcd(a, 26) = 1$ (so $a \neq 13$ and a is not even) then $f_{a,b}(x) = ax + b$ is 1-1 and onto. The substitution ciphers that use these functions are called affine ciphers and the pair (a, b) is called the key.

There are 12 possibilities of a, 26 for b. So overall

$12 \cdot 26 = 312$ affine ciphers (or 311 if we ignore the identity).

§2.7.

We could carry on studying new kinds of substitution ciphers, but they're all weak, as indicated by previous examples. E.g. what english word is this?

x v d v d v

It's "banana". We know this because there's only one word in the alphabet with repeated letters as in this word. (frequency analysis).

So, we introduce a completely new cipher:

The Vignière cipher (discovered by Bellaso, 1553, rediscovered by Vignière in 1586).

Algorithm is as follows:

Choose a key, which is a word / string of letters.

Write it, repeating, under your plaintext (this repeated string is the "keystream").

E.g. Key = "math"

Plaintext : encrypt this text

keystream mathmat hmat hmat

Convert this to numbers, add mod 26 : math = 12 0 19 7

$$\begin{array}{lccccccccccccccc}
 & 4 & 13 & 2 & 17 & 24 & 15 & 19 & 19 & 7 & 8 & 18 & 19 & 4 & 23 & 19 \\
 & 12 & 0 & 19 & 7 & 12 & 0 & 19 & 7 & 12 & 0 & 19 & 7 & 12 & 0 & 19 \\
\hline
\text{ciphertext:} & 16 & 13 & 21 & 24 & 10 & 15 & 12 & 0 & 19 & 8 & 11 & 0 & 16 & 23 & 12
\end{array}$$

Important! Notice that the "t"'s in "encrypt", "this" and "text" all went to different numbers! This is not a function $f: A \longrightarrow A$ anymore, it won't be a substitution cipher.

To decrypt, we subtract mod 26; assuming we know the key of "math" = 12 0 19 7.

$$\begin{array}{cccccccccl}
 & 16 & 13 & 21 & 24 & 10 & 15 & 12 & \ldots & \text{etc} \\
- & 12 & 0 & 19 & 7 & 12 & 0 & 19 & \ldots & \text{etc} \\
\hline
 & 4 & 13 & 2 & 17 & 24 & 15 & 19 & \ldots & \text{etc.} \\
 & e & n & c & r & y & p & t & \ldots &
\end{array}$$

- Note that if such a message is intercepted, the person trying to decode it doesn't even know the length of the key.
- Again, as with everything we covered so far, the message is only as secure as the key. (I.e. if someone gets the key, your code is cracked).
- Much more secure than substitution - not vulnerable to frequency analysis - but we will crack it later.

§2.8 Permutation block ciphers.

~~It turns out that the Vignère cipher is a special case of a "permutation block cipher".~~

One of the key features of the Vignière cipher, which turns out to be an excellent idea, is to chunk the plaintext into "blocks". I.e. the keystream "mathmathmathm..." naturally chunks the plaintext into blocks of length 4, with a single operation applied to each block.

Here's another cipher we can build using this trick.

Definition: A permutation of the set $\{1, 2 \ldots, m-1\}$, where m is a positive integer, is a 1-1 onto function $\sigma : \{1, \ldots, m-1\} \longrightarrow \{1, \ldots, m-1\}$.

These are often written as a matrix:

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 2 & 5 & 3 & 1 \end{pmatrix},$$

indicating where each element is mapped by σ.

So e.g the above is a function $\sigma : \{1, \ldots, 5\} \longrightarrow \{1, \ldots, 5\}$ with $\sigma(1) = 4$, $\sigma(2) = 2$, $\sigma(3) = 5$, $\sigma(4) = 3$, $\sigma(5) = 1$.

Then a <u>permutation cipher</u> is the following algorithm:

- Choose m, and a permutation σ of $\{1, \ldots, m\}$. Say $m = 3$ and $\sigma = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}$.
- Chunk your plaintext into blocks of size m.
- Apply σ to each block to get the the ciphertext.

<u>E.g</u>: With $m=3$ and σ as above,

<u>plaintext</u>: meet at ten

<u>chunked into blocks</u> mee tat ten

<u>apply σ</u> eme tta nte

<u>ciphertext</u> emettante

To decrypt, do the same procedure in reverse using the inverse function σ^{-1}:

$$\sigma^{-1} = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix}$$

ciphertext	emettante
chunked	eme tta nte
apply σ^{-1}	mee tat ten
plaintext	meet at ten.

One slight issue: If we choose a block size of m, and the length of the plaintext is not a multiple of m — then what?

In this case, we pad the plaintext with nonsense to make it the right length.

E.g. meet at five (with $m=3$) needs to be padded, so it becomes

$$\text{meetatfive}\underbrace{\text{ch}}_{}$$

← put anything here before encrypting.

§2.9 The Hill Cipher

Invented by Lester Hill 1929. This is an attempt to improve the Vignère and substitution ciphers. In each of them, the way a particular letter is encrypted depends only on its position in the plaintext.

E.g.: For Vignère, if we use "math" as the key, a "t" in positions 4, 8, 12, etc will always become a $19+7=26 \equiv 0 \bmod 26$, so an "a".

For the permutation block, if $m=3$ an "a" in position 3, 6, 9, etc. will always become a nearby letter.

E.g. if $\sigma = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}$ then an "a" in position 3, 6, 9, etc always becomes the letter to its left.