Math 1580: Cryptography *Lecture Notes*

E. Larson

Spring 2022

These are lecture notes for Math 1580: Cryptography taught at Brown University by Eric Larson in the Spring of 2022.

Contents

§0 January 26, 2022

§0.1 Course Logistics

§0.2 Introduction

§0.3 Simple Substitution Ciphers

§0.4 Divisibility

§1 January 28, 2022

§1.1 Greatest Common Divisors

§1.2 Euclidean Algorithm

§1.3 Linear Combinations

§2 January 31, 2022

§2.1 Linear Combinations *continued*

Recall from last time that we proposed that

$$\text{greatest common divisor} \leq \text{least linear combination}.$$

Example 2.1

$\gcd(2024, 748) = 44$ because we have

$$\begin{aligned}
2024 &= 748 \cdot 2 + 528 \\
748 &= 528 \cdot 1 + 220 \\
528 &= 220 \cdot 2 + 88 \\
220 &= 88 \cdot 2 + \boxed{44} \leftarrow \gcd(2024, 748) \\
88 &= 44 \cdot 2 + 0
\end{aligned}$$

We determine which linear combinations or 2024 znd 748 we can create:

$$\begin{aligned}
2024 &= 1 \cdot 2024 + 0 \cdot 748 \\
748 &= 0 \cdot 2024 + 1 \cdot 748 \\
528 &= 1 \cdot 2024 + (-2) \cdot 748 \\
220 &= 748 - 1 \cdot 528 \\
&= 748 - 1 \cdot (1 \cdot 2024 + (-2) \cdot 748) \\
&= -1 \cdot 2024 + 3 \cdot 748 \\
88 &= 528 - 2 \cdot 220 \\
&= \underbrace{[1 \cdot 2024 + (-2) \cdot 748]}_{528} - 2 \cdot \underbrace{[-1 \cdot 2024 + 3 \cdot 748]}_{220} \\
&= 3 \cdot 2024 - 8 \cdot 748 \\
44 &= 220 - 2 \cdot 88 \\
&= [-1 \cdot 2024 + 3 \cdot 748] - 2 \cdot [3 \cdot 2024 - 8 \cdot 748] \\
&= -7 \cdot 2024 + 19 \cdot 748
\end{aligned}$$

Following this example, we have shown that every common divisor of a and b can be written as a linear combination of a and b, and since the greatest common divisor has to be less than the least linear combination (as shown last time), the greatest common divisor *is* the least linear combination[1].

We realize that there is a *recurrence* happening here. If we call every set of coefficients x, y and z, w for a and b respectively, such that

$$\begin{aligned}
a &= x \cdot a_0 + y \cdot b_0 \\
b &= z \cdot a_0 + y \cdot b_0
\end{aligned}$$

where a_0 and b_0 are the original numbers, we can use a sliding window approach[2] again to determine the next set of x, y, z, w, a, b.

Recall from last time we had

$$\begin{aligned}
a' &= b \\
b' &= a \mod b
\end{aligned}$$

We can extend this algorithm for our new coefficients:

$$\begin{aligned}
x' &= z \\
y' &= w \\
z' &= w - \left\lfloor \frac{a}{b} \right\rfloor \cdot z \\
w' &= y - \left\lfloor \frac{a}{b} \right\rfloor \cdot w
\end{aligned}$$

[1]Assume for contradiction that the gcd were any less, then that would also be a linear combination. $\lightning$

[2]Updating our iterators on every loop by sliding our window of coefficients down.

where $\left\lfloor \frac{a}{b} \right\rfloor$ are the quotients from our Euclidean Algorithm. Note that initially, we have

$$a = 1 \cdot a_0 + 0 \cdot b_0$$
$$b = 0 \cdot a_0 + 1 \cdot b_0$$

so we have initial values of $x = 1, y = 0, z = 0, w = 0$.

so our code for the *extended Euclidean's Algorithm* is now

```
def ext_gcd(a, b):
    x, y, z, w = 1, 0, 0, 1
    while b != 0:
        x, y, z, w = z, w, w − (a // b) ∗ z, y − (a // b) ∗ w
        a, b = b, a % b
    return (x, y)
```

§2.2 Modular Arithmetic

Recall: We used a substitution/shift cipher to encrypt text:

$$\begin{matrix} \mathtt{Y} & \mathtt{E} & \mathtt{S} \\ \downarrow & \downarrow & \downarrow \\ \mathtt{D} & \mathtt{J} & \mathtt{X} \end{matrix}$$

by incrementing 5 letters for each lecture.

$a = 0,\ b = 1,\ \ldots,\ z = 25$.

We had this notion of

$$\text{ciphertext} = \text{plaintext} + 5$$
$$\mathtt{d} = \mathtt{y} + 5$$
$$3 = 24 + 5 = 29$$

Definition 2.2

We say $a \equiv b \mod m$ if $m \mid a - b$.

We say "a is congruent[a] to b modulo m".

[a] Congruence is a "behave like" equality.

Example 2.3

$$\begin{aligned} 24 + 5 &\equiv 3 \mod 26 \\ 22 + 2 &\equiv 1 \mod 12 \end{aligned}$$

The first example is from our shift sipher, the second example is equivalent to "two hours after 11:00, it is 1:00".

Proposition 2.4

If we have

$$\begin{aligned} a_1 &\equiv a_2 \mod m \\ b_1 &\equiv b_2 \mod m \end{aligned}$$

Then we have the following:

$$a_1 + b_1 \equiv a_2 + b_2 \mod m \tag{1}$$

$$a_1 - b_1 \equiv a_2 - b_2 \mod m \tag{2}$$

$$a_1 \cdot b_1 \equiv a_2 \cdot b_2 \mod m \tag{3}$$

Proof. For eq. (1), realize that we have

$$(a_1 + b_1) - (a_2 + b_2) = (a_1 - a_2) + (b_1 - b_2)$$

and the two terms on the right are each divisible by m by our premise. We can also write out

$$\begin{aligned} a_1 + b_1 &= (a_2 + \alpha m) + (b_2 + \beta m) \\ &= (a_2 + b_2) + (\alpha + \beta) \cdot m. \end{aligned}$$

Similarly, for eq. (2), we have

$$\begin{aligned} a_1 - b_1 &= a_2 + \alpha m - (b_2 + \beta m) \\ &= a_2 - b_2 + (\alpha - \beta) \cdot m. \end{aligned}$$

and for eq. (3), we have

$$\begin{aligned} a_1 \cdot b_1 &= (a_2 + \alpha m) \cdot (b_2 + \beta m) \\ &= a_2 \cdot b_2 + \alpha m b_2 + \beta m a_2 + \alpha\beta m^2 \\ &= a_2 \cdot b_2 + (\alpha b_2 + \beta a_2 + \alpha\beta m) \cdot m. \end{aligned}$$

which concludes the proofs of the premod rules. □

Proposition 2.5

There exists b with

$$a \cdot b \equiv 1 \mod m$$

if and only if $\gcd(a, m) = 1$.

Proof. We can write linear combination equation

$$a \cdot b + m \cdot k = 1$$

and we have that the following are equivalent (we cascade down the list and can easily prove the iff relations):

i. such a b exists,

ii. there is a solution b, k to this equation,

iii. 1 is a linear combination of a and m,

iv. 1 is the *least* linear combination of a and m,

v. $1 = \gcd(a, m)$.

so we have that $1 = \gcd(a, m)$ if and only if a's inverse b exists. □

§3 February 2, 2022

§3.1 Inverses mod m

Recall: Last time, we showed in proposition 2.5 that there exists an integer b with with $a \cdot b \equiv 1 \mod m$ iff $\gcd(a, m) = 1$.

Claim 3.1 — We further claim that if such a b exists, then it is unique mod m.

That is, if we have

$$a \cdot b_1 \equiv 1 \pmod m$$
$$a \cdot b_2 \equiv 1 \pmod m$$

then we have that $b_1 \equiv b_2 \pmod m$.

Proof. We consider $b_1 a b_2$. We have

$$b_2 \equiv (b_1 a) b_2 = b_2 (a b_2) \equiv b_2$$

all taking mod m. □

How, then, could we compute this inverse b efficiently?

Recall that last class, we used the extended Euclidean algorithm to compute the linear combination of a and m efficiently,

$$\begin{aligned} 1 &= a \cdot u + m \cdot v \\ &\equiv a \cdot \boxed{u} \mod m \end{aligned}$$

where u is b.

§3.2 Modular Arithmetic *continued*

Definition 3.2 (Ring of Integers mod m)

$\mathbb{Z}/m\mathbb{Z} = \{0, 1, 2, \dots, m-1\}$ with operations $+, -, \times \pmod{m}$.

Example 3.3

$\mathbb{Z}/4\mathbb{Z} = \{0, 1, 2, 3\}$. We have the following operation tables for $\mathbb{Z}/4\mathbb{Z}$:

$+$	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

$\times$	0	1	2	3
0	0	0	0	0
1	0	1	2	3
2	0	2	0	2
3	0	3	2	1

Definition 3.4 (Group of Units mod m)

We have the set of units in $\mathbb{Z}/m\mathbb{Z}$ as

$$\begin{aligned} (\mathbb{Z}/m\mathbb{Z})^\times &= \{a \in \mathbb{Z}/m\mathbb{Z} \mid \exists b \text{s.t. } a \cdot b \equiv 1\} \\ &= \{a \in \mathbb{Z}/m\mathbb{Z} \mid \gcd(a, m) = 1\} \end{aligned}$$

Example 3.5

$(\mathbb{Z}/4\mathbb{Z})^\times = \{1, 3\}$.

Definition 3.6 (Euler Totient Function)

We have

$$\varphi(m) = \#(\mathbb{Z}/m\mathbb{Z})^\times$$

which counts the number of units modulo m.

Example 3.7

$\varphi(4) = 2$.

Let's investigate the properties of units. Let's say a_1, a_2 are units. Which of the following have to be units?

	Does this have to be a unit?
$a_1 \cdot a_2$	Yes! Since $\gcd(a_1, m) = 1$ and $\gcd(a_2, m) = 2$ so we have $\gcd(a_1 a_2, m) = 1$. We also have $a_1 b_1 \equiv 1 \mod m$ and $a_2 b_2 \equiv 1 \mod m$, we have $(a_1 a_2)(b_2 b_1) \equiv 1 \mod m$.
$a_1 + a_2$	No. We have counterexample $m = 4$: $1 + 1$ is not a unit.
$a_1 - a_2$	Also no. For any a, $a - a = 0$ which is never a unit.

Definition 3.8 (Prime Number)

An integer $n \geq 2$ is prime if its only (positive) divisors are 1 and n.

Example 3.9

Numbers like $2, 3, 5, 7, 11, 12, \ldots$.

What if m is a prime number? Then we have

$$(\mathbb{Z}/m\mathbb{Z})^\times = \{1, 2, \ldots, m-1\}$$

so we can divide by elements of $\mathbb{Z}/m\mathbb{Z}$, just like in $\mathbb{Q}, \mathbb{R}, \mathbb{C}$. We can divide by any nonzero element of $\mathbb{Z}/m\mathbb{Z}$. We call these fields!

§3.3 Fast*ish* Powering

Problem. How might we compute $g^a \mod m$?

A naïve solution might be

```
def pow_mod(g, a, m):
    return g ** a % m
```

What if we tried to compute `pow_mod(239418762304, 12349876234, 12394876123482783641)` or something of the like? Something like this...

We could do something a bit more clever, like taking a mod every time we multiply:

```
def pow_mod(g, a, m):
    p = 1
    for i in range(a):
        p = (p * q) % m
    return p
```

Yet we *still* couldn't do `pow_mod(239418762304, 12349876234, 12394876123482783641)` since that takes the amount of time proportional to `a`[3].

Example 3.10

Let's try to compute 3^{37} by hand.

$$\begin{aligned}
3^1 & \equiv 3 \mod 100 \\
3^2 & \equiv 9 \mod 100 \\
3^4 = (3^2)^2 = & \equiv 81 \mod 100 \\
3^8 = (3^4)^2 = 81^2 = 6561 & \equiv 61 \mod 100 \\
3^{16} = (3^8)^2 \equiv 61^2 = 3721 & \equiv 21 \mod 100 \\
3^{32} = (3^{16})^2 \equiv 21^2 = 441 & \equiv 41 \mod 100
\end{aligned}$$

Since $37 = 32 + 4 + 1$, we can simply do

$$3^{37} = 3^{32} \cdot 3^4 \cdot 3^1 = 41 \cdot 81 \cdot 3 = 1863 \equiv 63 \mod 100$$

§4 February 4, 2022

§4.1 Fast Powering *continued*

[3] Which can become big...

Example 4.1

Recall: we wanted to compute $3^{37} \mod 100$

$$\begin{aligned} 3^1 &\equiv 3 \pmod{100} \\ 3^2 &\equiv 9 \\ 3^4 &\equiv 81 \\ 3^8 &\equiv 61 \\ 3^{16} &\equiv 21 \\ 3^{32} &\equiv 41 \end{aligned}$$

so we have

$$37 = 1 + 4 + 32 \qquad 3^{37} = 3^1 \cdot 3^4 \cdot 3^{32} \equiv 3 \cdot 81 \cdot 41 \equiv 63$$

How might we do this as an algorithm? We want to keep track of a few things, such as g (the current power), p (the multiple we are building), a (the remaining powers). This is akin to *deconstructing the power in binary and composing our product.*

```
def pow_mod(g, a, m):
    p = 1
    while a != 0:
        if a % 2 == 1:
            p = (p * g) % m
        a = a // 2
        g = g**2 % m
    return p
```

Example 4.2

$37 = 100101_2$, so we peel off last digits and multiply g into p.

Thinking about iterations, we have

g	p	a	a_2
3	1	37	$10010\underline{1}$
9	3	18	$10010\underline{0}$
81	3	9	$100\underline{1}$
61	43	4	$10\underline{0}$
21	43	2	$1\underline{0}$
41	43	1	$\underline{1}$
	$\boxed{63}$	0	$\underline{0}$

This algorithm takes approximately $\log_2(a)$ time to run, since it does as many steps for each digit in the binary representation of a.

§4.2 Fun Integers

Recall: An integer p is prime if $p \geq 2$ and

$$a \mid p \Rightarrow a = \pm 1, \pm p$$

Proposition 4.3

Let p be prime. Then $p \mid ab \Rightarrow p \mid a$ or $p \mid b$.

Example 4.4

p is not prime, this doesn't work. $p = 6$. $p \mid 4 \cdot 9 = 36$ but $6 \nmid 4$ and $6 \nmid 9$.

Proof. Let $g = \gcd(p, a)$. g is either 1 or p.

If $g = p$, then we have that $p = g \mid a$.

If $p = 1$, we can write this as

$$\begin{aligned} 1 = g &= p \cdot u + a \cdot v \\ b &= p \cdot ub + ab \cdot v \end{aligned}$$

since p is a multiple of p and ab is a multiple of p, we have that $p \mid b$. □

Theorem 4.5 (Fundamental Theorem of Arithmetic)

Any integer $a \geq 1$ can be factored into product of primes

$$a = p_1^{e_1} \cdots p_n^{e_n}$$

and this product of primes is *unique* up to rearrangement.[a]

[a]This is to say, $\mathbb{Z}$ is a UFD!

Example 4.6

Instead of thinking about integers, we think about $\mathbb{Z}[\sqrt{-5}]$, like

$$\mathbb{Z}[\sqrt{-5}] = \{a + b\sqrt{-5} \mid a, b \in \mathbb{Z}\}$$

Consider

$$6 = (1 + \sqrt{-5})(1 - \sqrt{-5}) = 2 \cdot 3$$

and each of $(1 + \sqrt{-5})$, $(1 - \sqrt{-5})$, 2, 3 have no divisors besides themselves and ± 1 (units).

Proof. We begin by working out an example:

Example 4.7

Let's factor 60, we can write this as

$$60 = 6 \cdot 10 = (2 \cdot 3) \cdot (2 \cdot 5) = 2^2 \cdot 3 \cdot 5.$$

What if we had different answers

$$p_1 p_2 \cdots p_t = a = q_1 q_2 \cdots q_s$$

We have that

$$\begin{aligned} p_1 \mid p_1 \cdots p_t &= q_1 \cdots q_s \\ &= q_1(q_2 \cdots q_s) \end{aligned}$$

So we have that $p_1 \mid q_1$ or $p_1 \mid q_2 \cdots q_s$, and we go on. So p_1 has to divide *one* of q_i. But both are primes, so they are equal $p_1 = q_i$. We rearrange so q_i is q_1. We strip off p_1 and q_1 and we have

$$p_2 \cdots p_t = q_2 \cdots q_s$$

we continue until we have no factors left[4] □

Definition 4.8

We define the <u>order</u>

$$\mathrm{ord}_p(a) = \text{the power of } p \text{ in the factorization of } a$$

such that we have

$$a = \prod_p p^{\mathrm{ord}_p(a)}$$

(This makes sense since $\mathrm{ord}_p(a)$ *is finite for finitely many* p*.)*

Theorem 4.9 (Fermat's Little Theorem)

Let p be prime, $a \in \mathbb{Z}/p\mathbb{Z}$,

$$a^{p-1} \equiv \begin{cases} 0 & \text{if } a \equiv 0 \\ 1 & \text{otherwise} \end{cases}$$

In abstract algebra, this directly follows from Lagrange's Theorem for $\mathbb{Z}/p\mathbb{Z}$, we give another argument.

[4] We could also have taken a well-ordering approach to this statement, taking a to be the least such non-uniquely factorizable number and showing that by peeling off p_1 and q_1, we get a smaller such a, which is a contradiction.

Proof. If $a \equiv 0$, this is sufficiently clear.

Let $a \not\equiv 0$. We look at the numbers

$$a, 2a, 3a, \ldots, (p-1)a$$

We consider 2 questions:

i. Are any of these divisible by p?

   No! $p \nmid a$ and $p \nmid i$ so $p \nmid ia$ for $1 \leq i < p$.

ii. Are any of these equal? i.e. $ia \equiv ja \mod p$.

   No again! a has an inverse mod p.

So we have that this list is a permutation of $\{1, 2, \ldots, p-1\}$, that is,

$$\{1, 2, \ldots, p-1\} = \{a, 2a, \ldots, (p-1)a\} \mod p$$

we multiply these sets together[5],

$$\begin{aligned} 1 \cdot 2 \cdot 3 \cdots (p-1) &\equiv a \cdot 2a \cdots (p-1)a \mod p \\ &\equiv (1 \cdot 2 \cdots p-1)a^{p-1} 1 \cdot 2 \cdot 3 \cdot (p-1)(a^{p-1} - 1) \equiv 0 \mod p \\ \implies a^{p-1} &\equiv 1 \mod p. \end{aligned}$$

Which is as desired. ∎

[5] This is truly a pro-gamer move