

SECRET CODES

Last time we talked about error-detecting and error-correcting codes. These were codes that added some additional information to a “message” (like the UPC information) in such a way that certain types of copying or transmission errors would be detected and possibly corrected.

Today we want to look at a different kind of code called a secret code. Now we want to encode messages in such a way that only the intended recipients can figure out the message. The study of secret codes is called *cryptology*. Obviously this is tremendously important for electronic financial transactions (we don’t want anyone else to have access to our money at the ATM) and for national security (we can’t have enemies eavesdropping on our war plans). In fact, the National Security Agency of the United States is the single largest employer of mathematicians in the country, and one of the main things they study is code-making and code-breaking.

Many of the coding schemes use modular arithmetic. Let’s look at two of them.

Secret Key Cryptography & the Caesar Cipher

Let’s start out with an old coding scheme. It is said that this scheme was used by Julius Caesar to code messages to his generals so that if the messenger were overtaken, the message would remain secret. Of course, the general knew the scheme and could recover the message if the messenger made the trip alive.

Here is the scheme. First the message is converted from letters to numbers using the following table.

A	B	C	D	E	F	G	H	I	J	K	L	M
0	1	2	3	4	5	6	7	8	9	10	11	12

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
13	14	15	16	17	18	19	20	21	22	23	24	25

1. Convert the message “I LOVE MATH!” into its 9-number code. Notice that the blanks and punctuation are ignored.

Next, some number (Caesar reportedly used 3) was added to each number in the code. (Of course, we should work mod 26 – why?) This number (3 for Caesar) is called the **encoding key**.

2. Convert the 9-number code into a new 9-number code by adding 3 (mod 26) to each of the 9 numbers.

Finally, the numeric encoding was converted back to a letter encoding (using the table again).

3. Convert the new 9-number code into a new 9-letter code using the letter-to-number conversion table.
4. It is possible to describe this process without using any numbers. Explain how to encode “I LOVE MATH!” without using any numbers.
5. MORE MESSAGES. Encode each of these messages using a Caesar cipher with encoding key 3.
 - a) “HELLO”
 - b) “GO KNIGHTS!”
 - c) “TO INFINITY AND BEYOND!”

Now redo them with an encoding key of 9.

Decoding the Caesar Cypher

OK, we're half done. We can *encode* messages. But how does the recipient decode them? You can probably figure this one out.

6. Decode the following messages:
 - a) FRUUHFW
 - b) BRX JRW LW
 - c) LWLVJRRG
 - d) QJAMNA FRCQXDC TNH

In the Caesar cipher, once you know the encoding key, it is easy to figure out how to decode.

7. Describe how to decode if the encoding key (e) is a) 3, b) 7, c) 9.
8. If your method for decoding involved subtraction (or going backwards), try to find another (equivalent method) that uses addition (or going forwards).

So decoding and encoding are the same thing (but with different keys)! The problem is that once someone knows the encoding key (e), they can easily figure out the decoding key (d). This has some big disadvantages. In particular, how does the sender tell the recipient what key to use? If you send it with the messenger, you might as well let the messenger know the message.

Soon we will learn about a different encoding scheme that doesn't have this disadvantage. But before we do, let's consider an advanced, modern version of the Caesar cypher: CaesarME!

CaesarME

The "ME" in CaesarME stands for Multiplication Edition. The Caesar cypher works by adding the key to the message. CaesarME multiplies instead.

9. Try coding the message "O BANANA" by (a) converting the message to a numeric code, (b) then multiplying each number by 2, and (c) then converting the result to letters.
What bad thing is happening?

This is a serious problem. But a different choice of encoding key takes care of it.

10. Show that using the encoding key $e = 3$ avoids the problem we just had. (Hint: multiply each number 0–25 by 3 (mod 26) and see what you get.)
11. Now find another encoding key that *does not* work.
12. It turns out that there is a fairly easy way to tell if a number e will make a suitable encoding key. Can you guess what it is? You may need to try a few more examples to figure it out.

- 13.** Use the encoding key $e = 3$ to encode the following messages using the CaesarME cypher:
- O BANANA
 - THIS WORKS GREAT
 - ET TU BRUTE
 - MULTIPLICATION ROCKS

OK. So $e = 3$ would make a usable encoding key. But how do we decode? The intuition is that we should be able to divide by three to get back the original message, but division by three is not quite right, since that leaves fractions for some numbers, and we don't know what to do with them. It turns out that this code can be decoded by multiplying by a different number d . (This should sound familiar. Remember that decoding the original Caesar cypher could be done by addition using a different key.)

- 14.** Determine the number d that works as a decoding key when the encoding key $e = 3$ is used in the multiplicative version of the Caesar cypher. Then use d decode the following message: JMGQJM OYFV NYNM.

So how do we determine the decoding key in general? Well, let's see. If M is a numeric message, then we want

$$M \cdot e \cdot d \equiv M \pmod{26}$$

since that means that if we first encode (multiply by e , then decode multiply by d , we get back to the original message (M). So what we are really looking for are numbers e and d such that $e \cdot d \equiv 1 \pmod{26}$

- 15.** If we use the encoding key $e = 5$, what is the decoding key?

(Hints: Trial and error will, of course, eventually find it for you. There is also a slightly cleverer way that will probably find it with fewer tries – depending on how lucky you are at guessing things to try. You may use either method.)

One must be careful to choose an encoding key that works and then determine the proper decoding key for that encoding key. But once one has found this pair of keys, CaesarME works very much like the original Caesar cypher. Although this coding scheme is a bit more complicated than the original Caesar cypher, and so makes decoding trickier for a novice, it really isn't any better. It is no more challenging for computer-aided cracking than the original Caesar cypher. It is also easy to figure out the decoding key once the encoding key is known (by a method known as Euclid's algorithm), so the guesswork involved in the previous problem is not really necessary at all.

But even if we coded blocks of letters into larger numbers and worked mod a much larger number than 26, both versions of the Caesar cypher would still be relatively easy for an expert with (computer assistance) to crack. The amazing thing, is that if we make just one more little change to the system, we get an encoding scheme known as RSA (named after its inventors: Rivest, Shamir and Adelman). RSA is one of the most important and widely used schemes in current practice. Furthermore, it has the added feature that even if the encoding key is known, it is still hard to decode. This allows RSA to be used for *public-key cryptography*.

Public Key Cryptography & RSA

Now we want to find a system such that knowing e is not enough to figure out d . That way, we can receive messages by posting our encoding key (e) in a directory. Anyone who wants to send a message secretly can simply encode it with our **public key**. Of course, we keep the decoding key a secret and use it to decode the message. We call this the **private key**.

Let's suppose we want to send numbers as messages (that saves us the conversion back and forth to letters, but we could do that if we needed to.) Our directory might look like this:

<u>Name</u>	<u>modulus</u>	<u>public key</u>	<u>private key</u>
John Q Public	26	5	only John knows
Jane Doe	65	5	only Jane knows
Sam I Am	91	7	only Sam knows
Fred Flintstone	91	29	only Fred knows

To send Jane a message, we work mod 65 and encode with $e = 5$. Of course, we can't use the Caesar cipher, because then everyone would know how to decode, and we only want Jane to be able to decode. So what can we do?

RSA is one particular system that is amazingly like the Caesar cipher. This time instead of using addition or multiplication we will use exponentiation. That little change makes all the difference.

So here is the scheme. To send a message to Jane we take our number (let's call it M for message) and raise it to the power 5 (mod 65). So to send Jane the number 10, we code it as

$$10^5 = 100000 \equiv 30 \pmod{65}$$

16. Check that you understand the system. What encoding would you use for sending the following number messages to Jane?

- a) 2 b) 3 c) 4 d) 20

Decoding uses the same process, but we need to know the **secret key**. Jane's secret key is 29 (don't tell). So she can decode her secret message from us (30) as follows:

$$30^{29} \pmod{65} \equiv 686303773648830000000000000000000000000000000000 \pmod{65} \equiv 10 \pmod{65}$$

It worked! Provided you believe my modular arithmetic. Notice how large 30^{29} is. Imagine using large numbers with 200-400 digits! (That's the size number used by RSA in practice.) The exponentiation would get so large, even a computer could not store all the digits.

The situation is very similar to trying to do this computation on a calculator that can only display 10-digit numbers. Given such a calculator, how do we compute $30^{29} \pmod{65}$? The trick is to do the modular arithmetic as we go along rather than at the end.

Here's how it works in our example. First notice that $10^5 = 1000000000$ is the smallest 11-digit number, so we can raise 2-digit numbers to about the fifth power on our calculator (with smaller numbers we can go a bit higher).

$$30^4 = 810000 \equiv 35 \pmod{65}$$

$$30^5 = 24300000 \equiv 10 \pmod{65}$$

$$30^{25} = (30^5)^5 \equiv 10^5 \equiv 100000 \equiv 30 \pmod{65}$$

$$30^{29} = 30^{25} \times 30^4 \equiv 30 \times 35 \equiv 1050 \equiv 10 \pmod{65}$$

By taking somewhat smaller steps, we could do this using numbers with at most 4-digits (twice the number of digits in 65).

17. Explain why $30^{25} = (30^5)^5$.

Where did the keys come from?

The numbers $n = 65$, $e = 5$, and $d = 29$ form the three ingredients for an RSA public key cryptography system. Not just any numbers will work, but there are many that do, and they can be found using a computer and some facts about prime numbers. Notice that what we need is

$$(M^e)^d = M^{ed} \equiv M \pmod{n}$$

Or in our case,

$$(M^5)^{29} = M^{145} \equiv M \pmod{65}$$

This says that if we take a message and code it, then decode it, we get the original message back, which is just what we want.

If n is a product of two prime numbers (like $65 = 5 \times 13$), then we can find numbers e and d that work. Thus for RSA to work, Jane must also keep secret how to factor n . Fortunately (for cryptography) it is much easier to multiply than to factor, so Jane can find two large primes (about 200-digits) multiply them together (to get a number with about 400 digits), but no one knows how to factor such a large number, even with the fastest computers available today running non-stop for weeks and months.

We won't go into the theory of all this (but you can read some of it in the text book, if you are interested). Suffice it to say, it makes use of some special properties of prime numbers and modular arithmetic, the two things we have been studying.

18. If you want to send the number-message 15 to Fred, what number do you send him?

19. If Jane receives the coded message 62, what was the secret number that someone sent to her?