

# Introduction to LLL “Cryptography”

Di Santi Giovanni

May 27, 2021

# Contents

<b>1</b>	<b>Linear Algebra Background</b>	<b>2</b>
1.1	Vector Spaces . . . . .	2
1.2	Lattices . . . . .	4
1.3	Problems . . . . .	5
1.3.1	SVP . . . . .	5
1.3.2	CVP . . . . .	5
<b>2</b>	<b>LLL</b>	<b>7</b>
2.1	Introduction . . . . .	7
2.2	Algorithm . . . . .	8
2.3	Applications . . . . .	8
<b>3</b>	<b>Cryptanalysis</b>	<b>9</b>
3.1	RSA introduction . . . . .	9
3.2	Lattices against RSA . . . . .	9
3.2.1	Example . . . . .	10
3.3	Key recovery ECDSA . . . . .	12

# Chapter 1

## Linear Algebra Background

### 1.1 Vector Spaces

**Definition 1.1.1** *Vector space.*

A vector space  $V$  is a subset of  $\mathbb{R}^m$  which is closed under finite vector addition and scalar multiplication, with the property that

$$a_1v_1 + a_2v_2 \in V \text{ for all } v_1, v_2 \in V \text{ and all } a_1, a_2 \in \mathbb{R}$$

**Definition 1.1.2** *Linear Combinations*

Let  $v_1, v_2, \dots, v_k \in V$ . A linear combination of  $v_1, v_2, \dots, v_k \in V$  is any vector of the form

$$\alpha_1v_1 + \alpha_2v_2 + \dots + \alpha_kv_k \text{ with } \alpha_1, \dots, \alpha_k \in \mathbb{R}$$

**Definition 1.1.3** *Linear Independence*

A set of vectors  $v_1, v_2, \dots, v_k \in V$  is linearly independent if the only way to get

$$a_1v_1 + a_2v_2 + \dots + a_kv_k = 0$$

is to have  $a_1 = a_2 = \dots = a_k = 0$ .

**Definition 1.1.4** *Bases*

Given a set of linearly independent vectors  $b = (v_1, \dots, v_n) \in V$  we say that  $b$  is a basis of  $V$  if  $\forall w \in V$  we can write

$$w = a_1v_1 + a_2v_2 + \cdots + a_nv_n$$

### Definition 1.1.5 *Vector's length*

The vector's length or **Euclidean norm** of  $v = (x_1, x_2, \dots, x_m)$  is

$$\|v\| = \sqrt{x_1^2 + x_2^2 + \cdots + x_m^2}$$

### Definition 1.1.6 *Dot Product*

Let  $v, w \in V \subset \mathbb{R}^m$  and  $v = (x_1, x_2, \dots, x_m), w = (y_1, y_2, \dots, y_m)$ , the dot product of  $v$  and  $w$  is

$$v \cdot w = x_1y_1 + x_2y_2 + \cdots + x_my_m$$

or

$$v \cdot w = \|v\|\|w\|\cos\theta$$

where  $\theta$  is the angle between  $v$  and  $w$  if we place the starting points of the vectors at the origin  $O$ .

Geometrically speaking  $v \cdot w$  is the length of  $w$  projected to  $v$  multiplied by the length of  $v$  as shown in 1.1

### Definition 1.1.7 *Orthogonal Basis*

An orthogonal basis for a vector space  $V$  is a basis  $v_1, \dots, v_m$  with the property that

$$v_i \cdot v_j = 0 \text{ for all } i \neq j$$

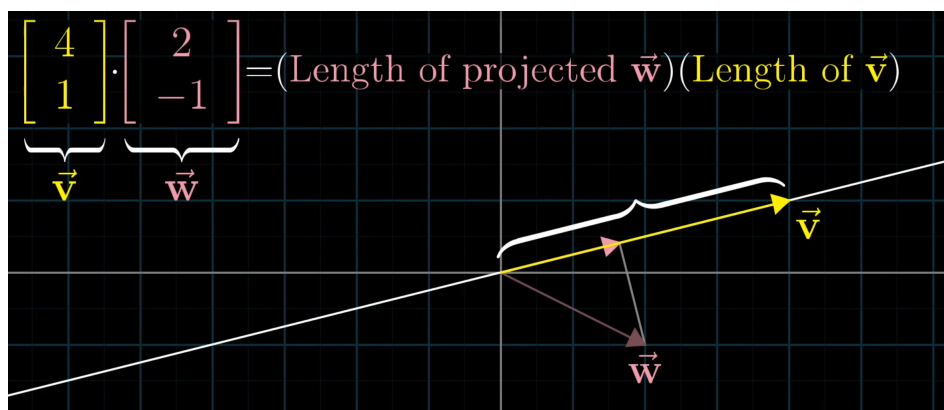


Figure 1.1: Dot Product By 3Blue1Brown

---

## Gram-Schmidt Algorithm

---

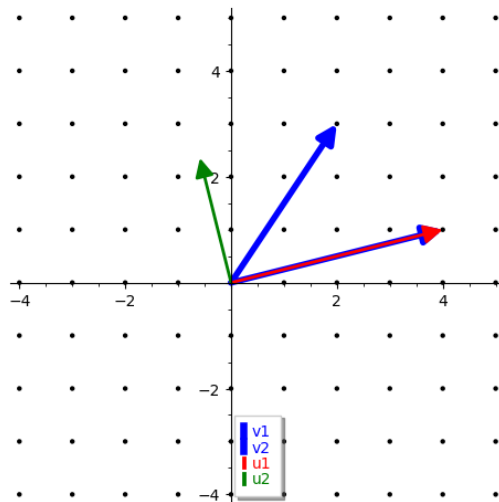


Figure 1.2: Gram Schmidt orthogonalization

If  $\|v_i\| = 1$  for all  $i$  then the basis is **orthonormal**.

Let  $b = (v_1, \dots, v_n)$ , be a basis for a vector space  $V \subset \mathbb{R}^m$ . There is an algorithm to create an orthogonal basis  $b^* = (v_1^*, \dots, v_n^*)$ . The two bases have the property that  $\text{Span}\{v_1, \dots, v_i\} = \text{Span}\{v_1^*, \dots, v_i^*\}$  for all  $i = 1, 2, \dots, n$

If we take  $v_1 = (4, 1), v_2 = (2, 3)$  as basis and apply gram schmidt we obtain  $u_1 = v_1 = (4, 1), u_2 = (-10/17, 40/17)$  as shown in 1.2

## 1.2 Lattices

### Definition 1.2.1 *Lattice*

Let  $v_1, \dots, v_n \in \mathbb{R}^m, m \geq n$  be linearly independent vectors. A **Lattice**  $L$  spanned by  $\{v_1, \dots, v_n\}$  is the set of all integer linear combinations of  $v_1, \dots, v_n$ .

$$L = \left\{ \sum_{i=1}^n a_i v_i, a_i \in \mathbb{Z} \right\}$$

If  $v_i$  for every  $i = 1, \dots, n$  has integer coordinates then the lattice is called **Integral Lattice**.

On the figure 1.3 we show a lattice  $L$  with bases  $v = (3, 1)$  and  $w = (-1, 1)$ , and on 1.4 the same lattice  $L$  with a different basis.

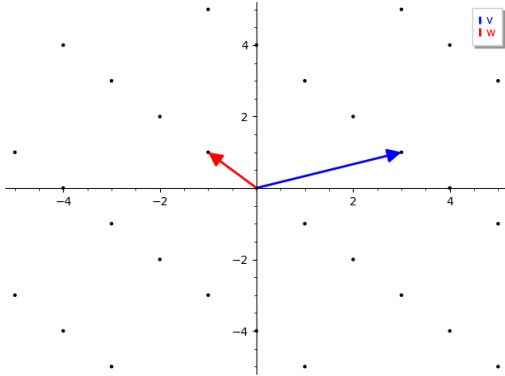


Figure 1.3: Lattice  $L$  spanned by  $v, w$

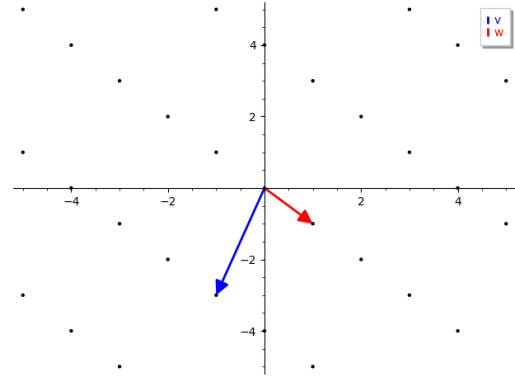


Figure 1.4: Lattice  $L$  spanned by  $v', w'$

## 1.3 Problems

### 1.3.1 SVP

**The Shortest Vector Problem (SVP):** Find a nonzero vector  $v \in L$  that minimizes the Euclidean norm  $\|v\|$ .

---

#### Gauss Reduction

---

Gauss's developed an algorithm to find an optimal basis for a two-dimensional lattice given an arbitrary basis. The output of the algorithm gives the shortest nonzero vector in  $L$  and in this way solves the SVP.

If we take for example  $v_1 = (10, 4), v_2 = (7, 5)$  and apply the gauss reduction algorithm we obtain  $w_1 = (3, -1), w_2 = (4, 6)$  1.5.  $w_1$  is the shortest nonzero vector in the lattice  $L$  spanned by  $v_1, v_2$ .

However the bigger the dimension of the lattice, the harder is the problem and there isn't a polynomial algorithm to find such vector.

### 1.3.2 CVP

**The Closest Vector Problem (CVP):** Given a vector  $w \in \mathbb{R}^m$  that is not in  $L$ , find a vector  $v \in L$  that is closest to  $w$ , in other words find a vector  $v \in L$  that minimizes the Euclidean norm  $\|w - v\|$ .

Example in 1.6

TODO: CVP and SVP are related.

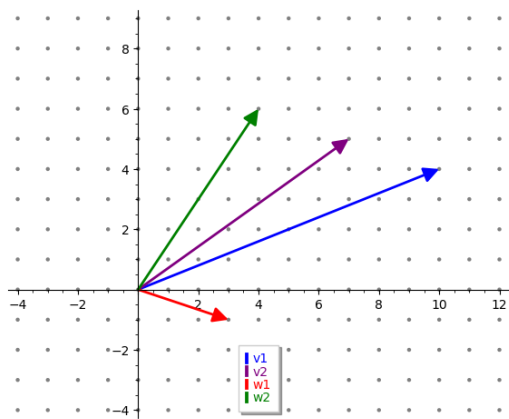


Figure 1.5: Gauss reduction

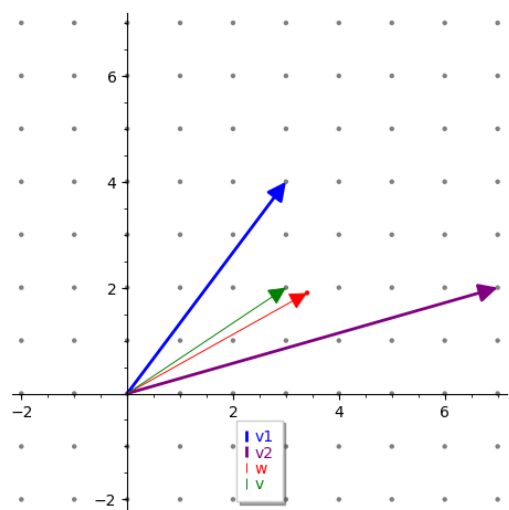


Figure 1.6: CVP

# Chapter 2

## LLL

### 2.1 Introduction

The **Lenstra-Lenstra-Lovász LLL** or  $L^3$  is a polynomial time algorithm to find a "shorter" basis.

#### Theorem 2.1.1 *LLL*

*Let  $L \in \mathbb{Z}^n$  be a lattice spanned by  $B = \{v_1, \dots, v_n\}$ . The LLL algorithm outputs a reduced lattice basis  $\{w_1, \dots, w_n\}$  with*

$$\|w_i\| \leq 2^{\frac{n(n-1)}{4(n-i+1)}} \det(L)^{\frac{1}{n-i+1}} \text{ for } i = 1, \dots, n$$

*in time polynomial in  $n$  and in the bit-size of the entries of the basis matrix  $B$ .*

Basically the first vector of the new basis will be as short as possible, and the other will have increasing lengths. The new vectors will be as orthogonal as possible to one another, i.e., the dot product  $w_i \cdot w_j$  will be close to zero.

#### Example

For example we can take the following basis (the rows are the vector) that span a lattice  $L$ .

$$L = \begin{pmatrix} 4 & 9 & 10 \\ 2 & 1 & 30 \\ 3 & 7 & 9 \end{pmatrix}$$

Applying the LLL algorithm we obtain



$$LLL(L) = \begin{pmatrix} -1 & -2 & -1 \\ 3 & -2 & 1 \\ -1 & -1 & 5 \end{pmatrix}$$

Where the first row is the shortest vector in the lattice  $L$ , and so solves the **SVP** problem. For higher dimensions however the LLL algorithm outputs only an approximation for the **SVP** problem.

## 2.2 Algorithm

TODO: Write algorithm and explain some steps

## 2.3 Applications

There are many applications of LLL

1. Factoring polynomials over the integers. For example, given  $x^2 - 1$  factor it into  $x + 1$  and  $x - 1$ .
2. Integer Programming. This is a well-known **NP**-complete problem. Using LLL, one can obtain a polynomial time solution to integer programming with a fixed number of variables.
3. Approximation to the **CVP** or **SVP**, as well as other lattice problems.
4. Application in cryptanalysis.

# Chapter 3

## Cryptanalysis

### 3.1 RSA introduction

**RSA** is one of the earliest and most used asymmetric cryptosystem. The usual step to generate a public/private key for **RSA** is the following

1. Fix  $e = 65537$  or  $e = 3$  (public).
2. Find two primes  $p, q$  such that  $p - 1$  and  $q - 1$  are relatively prime to  $e$ , i.e.  $\gcd(e, p - 1) = 1$  and  $\gcd(e, q - 1) = 1$ .
3. Compute  $N = p * q$  and  $\phi(n) = (p - 1) * (q - 1)$
4. Calculate  $d$  (private) as the multiplicative inverse of  $e$  modulo  $\phi(n)$ .
5.  $(N, e)$  is the public key,  $(N, d)$  is the private key.

To encrypt a message  $m$  with **textbook RSA**

$$c = m^e \mod N$$

To decrypt a ciphertext  $c$

$$m = c^d \mod N$$

### 3.2 Lattices against RSA

It's easy to find the roots of a univariate polynomial over the integers. Finding the roots of **modular** polynomial is hard, example:

$$f(x) \equiv 0 \mod N$$

Suppose  $N$  is an **RSA** modulus and we don't know the factorization of it. Let's have an univariate integer polynomial  $f(x)$  with degree  $n$

$$f(x) = x^n + a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_1x + a_0$$

Coppersmith showed how we can recover the value  $x_0$  such that  $f(x_0) \equiv 0 \pmod{N}$ , with  $x_0 < N^{\frac{1}{n}}$  in polynomial time using the following theorem

**Theorem 3.2.1 *Howgrave-Graham***

*Let  $g(x)$  be an univariate polynomial with  $n$  monomials and  $m$  be a positive integer. If we have some restraint  $X$  and the following equations hold*

$$g(x_0) \equiv 0 \pmod{N^m}, |x_0| \leq X \tag{3.1}$$

$$\|g(xX)\| < \frac{N}{\sqrt{n}} \tag{3.2}$$

Then  $g(x_0) = 0$  holds over the integers.

**3.2.1 Example**

We have a 100-bit **RSA** modulus

$$N = 0xf046522fb555a90bdc558fc93 \text{ and } e = 3.$$

Before the encryption the message  $m$  is padded as

$$z = \text{pad} || m = 0x74686973206b65793a00 || m$$

where  $||$  is the concatenation.

The ciphertext is

$$c = z^e \pmod{N} = 0x8185d7de1d7cb8801e69c0bf$$

Suppose that we don't know the factorization of  $N$  and we would like to know the message  $m$ . However we know the padding and the length of  $m < 2^{16}$ .

Let's define

$$a = 0x74686973206b65793a0000.$$

which is the known padding string.

Thus we have that  $c = (a + m)^3 \bmod N$ , for an unknown small  $m$ . We can define  $f(x) = (a + x)^3 - c$ , and so we setup the problem to find a small root  $m$  such that  $f(m) \equiv 0 \bmod N$

$$f(x) = x^3 + 0x15d393c596142306bae0000x^2 + 0x1b53c5e184a49b39f9ad9eedbx \\ + 0x9bb23cef d94fac9cc9cee9943$$

**Lattice construction.** Let the coefficients of  $f$  be  $f(x) = x^3 + f_2x^2 + f_1x + f_0$  and  $X = 2^{16}$  be the upper bound of the size of the root  $m$ . We can construct the matrix

$$B = \begin{pmatrix} X^3 & f_2X^2 & f_1X & f_0 \\ 0 & NX^2 & 0 & 0 \\ 0 & 0 & NX & 0 \\ 0 & 0 & 0 & N \end{pmatrix}$$

If we apply LLL to the lattice we get

We can use lattices to solve the problem:

- If we have some polynomial which share the same root  $x_0$  over  $N^m$  we can represent each of them as a row from lattice, and every linear integer combination of rows between them will yield another polynomial having a root  $x_0$ .
- Using **LLL** on the lattice we can find a reduced lattice basis, such that the norm of the shortest vector will satisfy the inequality of 3.2. **LLL** will only use **linear integer operations** on the basis vectors, in this way we preserve the root  $x_0$  and the bound  $X$ .
- Define the shortest vector in the reduced basis as  $v = (v_0, v_1, \dots, v_n)$ . We can construct the polynomial

$$g(x) = v_0 + \frac{v_1}{X}x + \frac{v_2}{X^2}x^2 + \dots + \frac{v_n}{X^n}x^n$$

Since  $g(x)$  is on the lattice we know that

$$g(x_0) \equiv 0 \pmod{N^m} \tag{3.3}$$

$$|x_0| \leq X \tag{3.4}$$

$$\|g(xX)\| < \frac{N^m}{\sqrt{\deg(g) + 1}} \tag{3.5}$$

### 3.3 Key recovery ECDSA

**End of Paper**

$gg^2$

# Bibliography