

# Lattices and their applications in cryptography and cryptanalysis

Steven Galbraith

The University of Auckland

December 9, 2012

- ▶ Apology.
- ▶ Mathematical background on lattices.
- ▶ Computational problems.
- ▶ Algorithms to solve computational problems.
- ▶ Cryptanalysis: subset-sum and approx-GCD.
- ▶ Learning with errors.

Please ask questions at any time.

The required mathematical and crypto background will vary.

- ▶ Let  $\underline{b}_1, \dots, \underline{b}_n$  be linearly independent vectors in  $\mathbb{R}^n$ .
- ▶ The set  $L = \{\sum_{i=1}^n x_i \underline{b}_i : x_i \in \mathbb{Z}\}$  is a (full rank) lattice. Call its elements **points** or **vectors**.
- ▶ Alternative definition: A discrete subgroup of  $\mathbb{R}^n$ .
- ▶ Everyone working with lattices should declare whether their vectors are **rows** or **columns**. I am using **rows**.
- ▶ The **basis matrix** is the  $n \times n$  matrix  $B$  whose rows are the vectors  $\underline{b}_1, \dots, \underline{b}_n$ .
- ▶ A lattice has many different bases.
- ▶ **Exercise:** Verify that the lattice  $\mathbb{Z}^2$  has the basis  $\{(1, 0), (0, 1)\}$  and the basis  $\{(3, 2), (2, 1)\}$  and infinitely many other bases.

- ▶ The basis vectors define a **parallelepiped**.
- ▶ The volume of the parallelepiped is given by  $|\det(B)|$ .
- ▶ **Exercise:** Prove that if  $B_1$  and  $B_2$  are basis matrices for a lattice  $L$  then there exists an  $n \times n$  integer matrix  $U$  such that  $B_2 = UB_1$  and  $\det(U) = \pm 1$ .
- ▶ **Exercise:** Let  $L_1$  and  $L_2$  be lattices such that  $L_2 \subseteq L_1$  and both have the same volume. Prove that  $L_1 = L_2$ .
- ▶ **Exercise:** Let  $A$  be an  $m \times n$  matrix ( $m \leq n$ ) and let  $q \in \mathbb{N}$ .  
Let

$$L_q(A) = \{\underline{v} \in \mathbb{Z}^n : \underline{v} \equiv \underline{x}A \pmod{q} \text{ for some } \underline{x} \in \mathbb{Z}^m\}$$

and

$$L_q^\perp(A) = \{\underline{y} \in \mathbb{Z}^n : \underline{y}A^T \equiv 0 \pmod{q}\}.$$

Prove that  $L_q(A)$  and  $L_q^\perp(A)$  are (full rank) lattices.

Harder: Give algorithms to compute a basis for  $L_q(A)$  and  $L_q^\perp(A)$ . [Hint: You need to use the Hermite normal form.]

# Computational Problems (Informally)

- ▶ Shortest vector problem (SVP): Given a basis matrix  $B$  for a lattice  $L$  find a non-zero vector  $\underline{v} \in L$  such that  $\|\underline{v}\|$  is minimal.

The norm here is usually the standard Euclidean norm in  $\mathbb{R}^n$ , but it can be any norm such as the  $l_1$  norm or  $l_\infty$  norm.

- ▶ Closest vector problem (CVP): Given a basis matrix  $B$  for a full rank lattice  $L \subseteq \mathbb{R}^n$  and an element  $\underline{t} \in \mathbb{R}^n$  find  $\underline{v} \in L$  such that  $\|\underline{v} - \underline{t}\|$  is minimal.

- ▶ Let  $L \subseteq \mathbb{R}^n$  be a lattice and  $B$  a basis matrix. The successive minima  $0 < \lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n$  are defined by

$$\lambda_i(L) = \inf\{r : i = \dim \text{span}\{\underline{v} : \underline{v} \in L \text{ and } \|\underline{v}\| \leq r\}\}.$$

- ▶ Minkowski's theorem:  $\lambda_1(L) < \sqrt{n} |\det(B)|^{1/n}$ .

- ▶ Let  $\underline{b}_1 = (1, 0)$  and  $\underline{b}_2 = (0, 1000)$  and  $\underline{t} = (0, 500)$ . Then  $\lambda_1(L) = 1$  but the nearest lattice point  $\underline{v}$  to  $\underline{t}$  has  $\|\underline{v} - \underline{t}\| = 500$ .  
Note that CVP is easy when given this lattice basis!
- ▶ **Exercise:** Give an algorithm that determines, for a basis matrix  $B$  of a lattice  $L$  and a vector  $\underline{t} \in \mathbb{R}^n$ , whether  $\underline{t}$  lies in the lattice. Does it help if  $L \subseteq \mathbb{Z}^n$  or  $L \subseteq \mathbb{Q}^n$ ?
- ▶ **Exercise:** Let  $L = \mathbb{Z}^n$ . Show that  $\lambda_i(L) = 1$  for all  $1 \leq i \leq n$ . Show that there exists an element  $\underline{t} \in \mathbb{R}^n$  such that  $\|\underline{v} - \underline{t}\| = \sqrt{n}/2$  for all  $\underline{v} \in L$ .

# Computational Problems

These problems depend on the basis matrix  $B$ , not on the lattice  $L$  itself. For complexity, the running time is a function of the number of bits needed to represent the basis matrix.

- ▶ Search-CVP: Given  $(B, \underline{t})$ , find  $\underline{v} \in L$  such that  $\|\underline{v} - \underline{t}\|$  is minimal.
- ▶ Decision-CVP: Given  $(B, \underline{t})$  and  $r > 0$ , decide whether or not there is  $\underline{v} \in L$  such that  $\|\underline{v} - \underline{t}\| \leq r$ .
- ▶ Search-SVP: Given  $B$ , find non-zero  $\underline{v} \in L$  such that  $\|\underline{v}\|$  is minimal.
- ▶ Decision-SVP: Given  $B$  and  $r > 0$ , decide whether or not  $\lambda_1(L) \leq r$ .
- ▶ SIVP: Given  $B$ , find  $n$  linearly independent vectors  $\underline{v}_1, \dots, \underline{v}_n \in L$  minimising  $\max \|\underline{v}_i\|$ .
- ▶  $\gamma$ -approx SVP:
- ▶  $\gamma$ -approx CVP:
- ▶ GapSVP $_\gamma$ :



- ▶ Decision-SVP is NP-complete (see Chapter 3 of Micciancio and Goldwasser).
- ▶ SVP is “easier”, but still hard (see Chapter 4 of Micciancio and Goldwasser).
- ▶ **Exercise:** Show that Decision-CVP is polynomial-time equivalent to Search-CVP. In other words, given an oracle for Decision-CVP, give an algorithm to solve Search-CVP.  
[Hint: Given basis  $\{\underline{b}_1, \dots, \underline{b}_n\}$  and  $\underline{t}$  see if answer is same when run oracle on that basis and the set  $\{2\underline{b}_1, \underline{b}_2, \dots, \underline{b}_n\}$ .]

## Part 2: Algorithms for computational problems

Any questions about the first part?



- ▶ SVP and CVP can be easy when given certain bases for certain lattices.
- ▶ Consider the lattice with basis  $\{(1, 0, 0), (0, 2, 0), (0, 0, 5)\}$ . Then SVP and CVP are easy.
- ▶ A good lattice basis has vectors that are “close to orthogonal”.
- ▶ The invariance of lattice volume implies that such vectors are also relatively short.

# Lattice reduction

- ▶ The goal of lattice reduction is to take as input a basis for a lattice and to compute a new basis for the same lattice. The new basis should have vectors that are “as close to orthogonal as possible” and “as short as possible”.
- ▶ The famous Lenstra-Lenstra-Lovasz (LLL) algorithm is polynomial-time in the input and outputs a basis with relatively good properties. (Note that it is exponential-time in terms of the rank/dimension  $n$ .)
- ▶ The LLL algorithm is based on Gram-Schmidt. It’s goal is to ensure that the Gram-Schmidt orthogonal basis does not decrease in size too quickly.
- ▶ Theorem: Let  $B$  be an LLL-reduced lattice basis (with  $\delta = 3/4$ ). Then the first row  $\underline{b}_1$  of  $B$  satisfies

$$\|\underline{b}_1\| \leq 2^{(n-1)/2} \lambda_1.$$

# Lattice reduction

- ▶ The exponential approximation factors mean that LLL usually becomes useless once the rank is large enough.
- ▶ The 2-dimensional case of LLL is essentially the Euclid/continued fraction algorithm.
- ▶ There are also exponential-time enumeration algorithms that are guaranteed to output the shortest vector in a lattice. They are easily prevented when  $n$  is large enough.
- ▶ There are many variants of LLL. Block LLL combines LLL with enumeration algorithms performed on low-rank sublattices.
- ▶ See the LLL+25 conference proceedings (Nguyen and Vallée, editors).

# Algorithms for CVP

- ▶ There are also enumeration algorithms for CVP. They are exponential-time, but guaranteed to output the closest lattice vector to  $\underline{t} \in \mathbb{R}^n$ .
- ▶ The **Babai rounding algorithm** is fast and simple, but is not guaranteed to output the closest lattice vector:  
Given a basis  $\{\underline{b}_1, \dots, \underline{b}_n\}$  and  $\underline{t} \in \mathbb{R}^n$ , compute real numbers  $x_i$  such that  $\underline{t} = \sum_i x_i \underline{b}_i$ . Then compute the lattice vector  $\underline{v} = \sum_i [x_i] \underline{b}_i$ .
- ▶ **Exercise:** Show that  $\underline{v}$  lies in the parallelepiped centered on  $\underline{t}$ . Show that if  $\underline{t} \notin L$  then there is a unique such lattice vector.
- ▶ The Babai nearest plane algorithm is a little better. There are also nice variants of it by Klein and Lindner-Peikert.

# Prehistoric crypto applications (GGH)

- ▶ Let  $B$  be a “nice” lattice basis for a lattice in  $\mathbb{Z}^n$  with large volume.  
Let  $U$  be a “random”  $n \times n$  integer matrix with  $\det(U) = \pm 1$ .
- ▶ The GGH public key is  $B' = UB$  and the private key is  $B$ .
- ▶ To encrypt a message  $\underline{m} \in \{-M, \dots, M\}^n \subseteq \mathbb{Z}^n$  choose a “small” error vector  $\underline{e} \in \mathbb{Z}^n$  and compute the ciphertext  $c = \underline{m}B' + \underline{e}$ .
- ▶ To decrypt one uses the nice lattice basis to solve the closest vector problem and hence find a lattice point  $\underline{v}$  such that  $c = \underline{v} + \underline{e}$ . One then computes  $\underline{m} = \underline{v}(B')^{-1}$ .
- ▶ **Exercise:** Show that the GGH cryptosystem does not have indistinguishability security under a passive attack.
- ▶ **Exercise:** A variant of GGH is to swap the roles of the message and the randomness. Explain the scheme. Show that this variant also does not have indistinguishability security under passive attacks.

# Cryptanalysis of GGH

- ▶ One can attempt to break GGH using lattice reduction on  $B'$ , followed by Babai rounding or some other CVP algorithm. This is hopeless if  $n > 200$ .
- ▶ Nguyen cryptanalysed the original GGH proposal (which had errors of a specific form).



# GGH signatures

- ▶ Let  $B'$  be a GGH public key as before.
- ▶ Given a message  $m$ , hash it to a “random” element  $H(m) \in \mathbb{Z}^n$ .  
Then, using the private key, compute a lattice vector  $\underline{s}$  close to  $H(m)$ . The signature on message  $m$  is then  $\underline{s}$ .
- ▶ To verify the signature one checks that  $\underline{s}$  lies in the lattice and that  $\|\underline{s} - H(m)\|$  is sufficiently small.
- ▶ Problem:  $\underline{s} - H(m)$  lies in the parallelepiped corresponding to the nice basis  $B$ .  
Nguyen-Regev (and more recently Ducas-Nguyen at ASIACRYPT 2012) have given a powerful attack to “learn” the nice basis from the statistical properties of many samples  $\underline{s} - H(m)$ .
- ▶ Lyubashevsky gives better approaches to lattice signatures.

## Part 3: Lattices in cryptanalysis

But first, any questions?



# Short history of lattices in cryptanalysis

- ▶ Subset-sum/knapsack cryptosystems.
- ▶ Simultaneous Diophantine approximation.
- ▶ Coppersmith's algorithm for small roots of polynomial equations.
- ▶ Variants of RSA (zillions of papers; Subhamoy Maitra knows all about this).
- ▶ NTRU.
- ▶ Fixed pattern RSA signature forgery.
- ▶ Side-channel attacks (e.g., dlog signatures with some known bits or poor randomness).
- ▶ Noisy Chinese remainder theorem.
- ▶ Approximate GCD.

See survey paper Phong Nguyen, Public-Key Cryptanalysis, or my book.

# The subset-sum problem

- ▶ Let  $S = (m_1, \dots, m_k)$  be a list of (large) integers  
 $0 < m_i \leq M$ .
- ▶ Let  $s = \sum_{i=1}^k x_i m_i$  where  $x_i \in \{0, 1\}$ .
- ▶ The problem is: Given  $S$  and  $s$  to compute the values  $x_i$ .
- ▶ **Exercise:** Show that the subset-sum problem is well-defined (i.e., there is a unique solution) for “random” lists of weights  $S$  as long as  $2^k$  is much smaller than  $kM$ .
- ▶ **Exercise:** Show that if  $m_i = 2^{i-1}$  then subset-sum has a unique solution, and that the solution is easy to compute.
- ▶ Subset-sum is NP-hard in general. But variants of it arise in knapsack cryptosystems.
- ▶ **Exercise:** Describe a “time-memory tradeoff” algorithm to solve the subset-sum problem that requires  $\tilde{O}(2^{k/2})$  time and space. (Orr Dunkelman’s talk does better.)

# Lattice attack on subset-sum

Let

$$B = \begin{pmatrix} 1 & 0 & \cdots & 0 & m_1 \\ 0 & 1 & & 0 & m_2 \\ \vdots & & \ddots & \vdots & \vdots \\ 0 & 0 & & 1 & m_k \\ 0 & 0 & \cdots & 0 & -s \end{pmatrix}$$

and note that

$$(x_1, x_2, \dots, x_k, 1)B = (x_1, x_2, \dots, x_k, 0)$$

might be a short vector compared with the Minkowski bound

$$\lambda_1 \leq \sqrt{n}|s|^{1/(k+1)}.$$

Hence, subset-sum can be solved (in some cases) by solving SVP. For further details see Coster, Joux, LaMacchia, Odlyzko, Schnorr and Stern.

# Approximate GCD

- ▶ Let  $p$  be a fixed secret.
- ▶ Suppose given  $X_i = q_i p + e_i$  for  $1 \leq i \leq k$  where  $q_i, e_i \in \mathbb{Z}$ ,  $q_i > 0$ , and  $|e_i|$  are “small” compared with  $p$ .  
The goal is to compute  $p$ .
- ▶ This is well-defined if  $k$  is large enough.
- ▶ **Exercise:** Recall that the extended Euclid algorithm on  $X_1$  and  $X_2$  computes a sequence of triples of integers  $(s_i, t_i, r_i)$  such that  $s_i X_1 + t_i X_2 = r_i$  and  $|r_i s_i| < X_2$ .  
If  $q_2 e_1 - q_1 e_2 < p$  then show that this process is likely to yield  $(s_i, t_i) = (q_1, q_2)$ , and thus  $p = [X_1/q_1]$ .

# Lattice attack on approx GCD

Since  $X_i = q_i p + e_i$  we have  $q_i X_1 - q_1 X_i = q_i e_1 - q_1 e_i$ .

Let

$$B = \begin{pmatrix} E & -X_2 & -X_3 & \cdots & -X_k \\ 0 & X_1 & 0 & \cdots & 0 \\ 0 & 0 & X_1 & & 0 \\ \vdots & \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & X_1 \end{pmatrix}$$

and note that

$$(q_1, q_2, \dots, q_k)B = (Eq_1, q_2 e_1 - q_1 e_2, \dots, q_k e_1 - q_1 e_k)$$

which is shorter than  $(0, X_1, 0, \dots, 0)$  and might be very a short vector in the lattice.

Hence, one can try to attack approx-GCD using lattice reduction.

- ▶ The approx-GCD problem seems to be hard if the  $q_i$  are large enough.
- ▶ The van Dijk, Gentry, Halevi, Vaikuntanathan homomorphic encryption scheme:  
A ciphertext encrypting  $m \in \{0, 1\}$  is an integer  $c = pq + 2r + m$  where  $|r| < p/2$ .
- ▶ To decrypt, knowing the secret  $p$ , we reduce modulo  $p$  and then reduce modulo 2.
- ▶ The sum and product of two ciphertexts correspond to the sum and product (mod 2) of the corresponding messages, as long as the errors remain small enough.
- ▶ **Exercise:** Give a brute force attack on approx-GCD by “trying all errors”.  
See Chen-Nguyen (EUROCRYPT 2012) for a faster solution.



# Lattices in cryptanalysis

The important point is that lattices can be used to solve all sorts of computational problems, even apparently “non-linear” problems like finding small roots of modular polynomials, or approximate GCD.

## Part 4: Learning with errors

But first, any questions?



Steven Galbraith

# Learning with Errors (LWE)

Oded Regev (2005)

- ▶ Let  $q$  be a prime and  $n, m \in \mathbb{N}$ . [Example:  $n = 200$ ,  $m = 2300$ ,  $q = 40009$ .]
- ▶ Let  $\underline{s} \in \mathbb{Z}_q^n$  be secret (**column** vector).
- ▶ Suppose one is given an  $m \times n$  matrix  $A$  chosen uniformly at random with entries in  $\mathbb{Z}_q$  and a length  $m$  vector

$$\underline{c} \equiv A\underline{s} + \underline{e} \pmod{q}$$

where the vector  $\underline{e}$  has entries chosen independently from a “discrete normal distribution” on  $\mathbb{Z}$  with mean 0 and standard deviation 3.

- ▶ The task is to find the vector  $\underline{s}$ .

# Discrete Gaussians

- ▶ The Gaussian distribution (= normal distribution) on  $\mathbb{R}$  with mean 0 and variance  $s^2$  has probability density function

$$f(x) = \frac{1}{s\sqrt{2\pi}} e^{-x^2/(2s^2)}.$$

- ▶ To define the discrete Gaussian on  $\mathbb{Z}$  compute

$$M = 1 + 2 \sum_{k=1}^{\infty} e^{-k^2/(2s^2)}$$

and define the distribution on  $x \in \mathbb{Z}$  by

$$\Pr(x) = \frac{1}{M} e^{-x^2/(2s^2)}.$$

- ▶ Sampling closely from this distribution in practice is non-trivial!

# Remarks on Learning with Errors

- ▶ LWE: Given  $A$  and  $\underline{c} \equiv A\underline{s} + \underline{e} \pmod{q}$  to find  $\underline{s}$ .
- ▶ If  $\underline{e} = 0$  then easy.
- ▶ The solution  $\underline{s}$  is not uniquely determined, but one value  $s$  is significantly more likely than the others.  
Hence LWE is well-defined as a maximum likelihood problem.
- ▶ LWE is essentially a special case of CVP: We are given a matrix  $A$  generating the modular lattice  $L_q(A^T)$  in  $\mathbb{Z}^m$  and a target  $\underline{c} \in \mathbb{Z}^m$  and want to find a lattice point  $\underline{y} \equiv A\underline{s} \pmod{q}$  close to  $\underline{c}$ .  
Hence, the natural way to solve LWE is to perform lattice reduction on  $A$  and then apply Babai nearest plane (see Lindner-Peikert).

# Remarks on Learning with Errors

- ▶ Conversely, Regev showed that if one can solve LWE in the average case then one can solve a variant of the closest vector problem in a lattice in the **worst-case**.  
Regev further showed a quantum average-worst reduction to decision-SVP (also see Peikert).
- ▶ Decision-LWE: Given  $(A, \underline{c})$  decide whether or not  $\underline{c} \equiv A\underline{s} + \underline{e} \pmod{q}$  for some  $\underline{c}$  and error vector  $\underline{e}$ .
- ▶ **Exercise:** Show that if one has an oracle that solves decision-LWE then one can solve search-LWE.

# Classical Linear Regression

- ▶ Let  $\underline{s} \in \mathbb{R}^n$ ,  $A$  be an  $m \times n$  matrix and  $\underline{e}$  an error vector in  $\mathbb{R}^m$  with entries identically and independently chosen from some distribution (e.g., normal with mean 0).
- ▶ Given  $A$  and  $\underline{y} = A\underline{s} + \underline{e}$  the problem is to compute  $\underline{s}$ . This is a well-defined question if  $m \gg n$  (depending on the error distribution).
- ▶ This is solved by the least squares method. A good estimator for  $\underline{s}$  is

$$\hat{\underline{s}} = (A^T A)^{-1} A^T \underline{y}.$$

- ▶ In other words, solving linear regression is “easy”.

# Linear Regression mod $q$

- ▶ Since linear algebra works over any field it is natural to replace the field  $\mathbb{R}$  by the field  $\mathbb{Z}_q$ .
- ▶ Let  $\underline{s} \in \mathbb{Z}_q^n$ ,  $A$  be an  $m \times n$  matrix with entries in  $\mathbb{Z}_q$ , and  $\underline{e}$  be an error vector in  $\mathbb{Z}_q^m$ .
- ▶ Given  $A$  and  $\underline{y} = A\underline{s} + \underline{e} \pmod{q}$  the problem is to compute  $\underline{s}$ .
- ▶ Is

$$\hat{\underline{s}} \equiv (A^T A)^{-1} A^T \underline{y} \pmod{q}$$

a good estimator for  $\underline{s}$ ?

In other words, is  $\underline{s} - \hat{\underline{s}}$  small?



# Example

$$\underline{s} = \begin{pmatrix} 5 \\ 76 \end{pmatrix}, \quad A = \begin{pmatrix} 22 & 102 \\ 191 & 176 \\ -26 & 104 \end{pmatrix}, \quad \underline{e} = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}.$$

Least squares computes

$$\hat{\underline{s}} \approx \begin{pmatrix} 4.993 \\ 76.003 \end{pmatrix}.$$

## Example

$$\underline{s} = \begin{pmatrix} 5 \\ 76 \end{pmatrix}, \quad A = \begin{pmatrix} 22 & 102 \\ 191 & 176 \\ -26 & 104 \end{pmatrix}, \quad \underline{e} = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}.$$

Least squares computes

$$\hat{\underline{s}} \approx \begin{pmatrix} 4.993 \\ 76.003 \end{pmatrix}.$$

Now work over  $\mathbb{Z}_{311}$ . The formula gives

$$\hat{\underline{s}} = \begin{pmatrix} 274 \\ 223 \end{pmatrix}.$$

**Exercise:** Explain why linear regression does not work modulo  $q$ .

# Public Key Cryptography from LWE

- ▶ Private key:  $\underline{s}$  (column vector)
- ▶ Public key:  $A, \underline{c} = A\underline{s} + \underline{e} \pmod{q}$
- ▶ To encrypt  $M \in \{0, 1\}$ :
  - ▶ Choose  $\underline{u} \in \{0, 1\}^m$  (row vector)
  - ▶ Set  $c_1 = \underline{u}A \pmod{q}, c_2 = \underline{u}\underline{c} + M(p-1)/2 \pmod{q}$
- ▶ To decrypt: Compute  $v = c_2 - c_1\underline{s} \pmod{q}$  reduced to the interval  $\{-(q-1)/2, \dots, -1, 0, 1, \dots, (q-1)/2\}$ .  
If  $|v| < q/4$  then output 0, else output 1.
- ▶ To break the cryptosystem one could try to compute  $\underline{s}$  or  $\underline{u}$ .  
Note that  $c_1$  can be viewed as multiple modular subset-sum instances on the same secret  $\underline{u}$ .

# Why LWE?

LWE has a number of amazing applications:

- ▶ Hierarchical identity-based encryption.
- ▶ Homomorphic encryption.
- ▶ Lossy trapdoor functions.

# Thank You

