Solving the Learning With Errors Problem

Martin R. Albrecht

Information Security Group, Royal Holloway, University of London

Post-Quantum Research
Identifying Future Challenges and Directions
8th - 9th May 2014
Isaac Newton Institute, Cambridge

Contents

Introduction

BDD & SIS: Lattice Reduction

SIS: Combinatorial Algorithms

BDD: Arora & Ge

Learning with Errors

Given $(\mathbf{A}, \mathbf{c})$ with $\mathbf{c} \in \mathbb{Z}_q^m$, $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$, $\mathbf{s} \in \mathbb{Z}_q^n$ and $\mathbf{e} \in \mathbb{Z}_q^{m\times \ell}$ do we have

$$\begin{pmatrix} \\ \\ \mathbf{c} \\ \\ \\ \end{pmatrix} = \begin{pmatrix} \leftarrow & n & \rightarrow \\ & & \\ & \mathbf{A} & \\ & & \\ & & \end{pmatrix} \times \begin{pmatrix} \mathbf{s} \end{pmatrix} + \begin{pmatrix} \\ \\ \mathbf{e} \\ \\ \\ \end{pmatrix}$$

or $\mathbf{c} \leftarrow_{\$} \mathcal{U}(\mathbb{Z}_q^m)$.

We Want to Build Crypto Systems

Not precise enough

"Given m, n, q and χ it takes $2^{\tilde{\mathcal{O}}(n^{2\epsilon})}$ operations in $\mathbb{Z}_q$ to solve LWE."

Solving Strategies

Given $\mathbf{A}, \mathbf{c}$ with $\mathbf{c} = \mathbf{A} \times \mathbf{s} + \mathbf{e}$ or $\mathbf{c} \leftarrow_{\$} \mathcal{U}(\mathbb{Z}_q^m)$

- Solve the **Short Integer Solutions** problem (SIS) in the left kernel of $\mathbf{A}$, i.e.

  $$\text{find a short } \mathbf{w} \text{ such that } \mathbf{w} \times \mathbf{A} = 0$$

  and check if

  $$\langle \mathbf{w}, \mathbf{c} \rangle = \mathbf{w} \times (\mathbf{A} \times \mathbf{s} + \mathbf{e}) = \langle \mathbf{w}, \mathbf{e} \rangle$$

  is short.
- Solve the **Bounded Distance Decoding** problem (BDD), i.e.

  $$\text{find } \mathbf{s}' \text{ such that } \|\mathbf{w} - \mathbf{c}\| \text{ with } \mathbf{w} = \mathbf{A} \times \mathbf{s}' \text{ is minimised.}$$

Solving Strategies

Given $\mathbf{A}, \mathbf{c}$ with $\mathbf{c} = \mathbf{A} \times \mathbf{s} + \mathbf{e}$ or $\mathbf{c} \leftarrow_{\$} \mathcal{U}(\mathbb{Z}_q^m)$

- Solve the **Short Integer Solutions** problem (SIS) in the left kernel of $\mathbf{A}$, i.e.

  $$\text{find a short } \mathbf{w} \text{ such that } \mathbf{w} \times \mathbf{A} = 0$$

  and check if

  $$\langle \mathbf{w}, \mathbf{c} \rangle = \mathbf{w} \times (\mathbf{A} \times \mathbf{s} + \mathbf{e}) = \langle \mathbf{w}, \mathbf{e} \rangle$$

  is short.
- Solve the **Bounded Distance Decoding** problem (BDD), i.e.

  $$\text{find } \mathbf{s}' \text{ such that } \|\mathbf{w} - \mathbf{c}\| \text{ with } \mathbf{w} = \mathbf{A} \times \mathbf{s}' \text{ is minimised.}$$

Contents

Introduction

BDD & SIS: Lattice Reduction

SIS: Combinatorial Algorithms

BDD: Arora & Ge

SIS

Find $\mathbf{w}$ s.t. $\mathbf{w} \times \mathbf{A} = 0$ with $\|\mathbf{w}\| \approx \frac{1}{\alpha}$ to get

$$\| \langle \mathbf{w}, \mathbf{e} \rangle \| \approx \frac{\alpha\, q}{\alpha} = q$$

to distinguish from $\mathcal{U}(\mathbb{Z}_q)$ in poly(n) time. Let $\mathbf{B}$ denote a basis for $\{\mathbf{w} \mid \mathbf{w} \cdot \mathbf{A} = 0\}$. Using standard results from lattice reduction we get

$$\begin{aligned} \frac{1}{\alpha} &= \delta^m \det(\mathbf{B})^{1/m} = \delta^{\sqrt{n\ \log_2 q / \log_2 \delta}} q^{n/\sqrt{n\ \log_2 q / \log_2 \delta}} \\ &= 2^{2\sqrt{n\ \log_2 \delta\ \log_2 q}}. \end{aligned}$$

It follows that lattice reduction with $\delta = 2^{\frac{\log_2^2 \alpha}{4n\ \log_2 q}}$ solves Decision-LWE.

BDD

Lattice reduction produces **short** and relatively **orthogonal bases** not only **short vectors**.

1. Reduce lattice basis to recover short and orthogonal basis $\mathbf{A}'$
2. Use variant of Babai's nearest plane algorithm to find vector close to $\mathbf{c} = \mathbf{A}' \times \mathbf{s} + \mathbf{e}$.

Tradeoff between lattice reduction and decoding stage.

Contents

Introduction

BDD & SIS: Lattice Reduction

SIS: Combinatorial Algorithms

BDD: Arora & Ge

BKW Algorithm I

We revisit Gaussian elimination:

$$\left(\begin{array}{c|c|cccc|c} \mathbf{a}_{11} & \mathbf{a}_{12} & \mathbf{a}_{13} & \cdots & \mathbf{a}_{1n} & c_1 \\ \mathbf{a}_{21} & \mathbf{a}_{22} & \mathbf{a}_{23} & \cdots & \mathbf{a}_{2n} & c_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{a}_{m1} & \mathbf{a}_{m2} & \mathbf{a}_{m3} & \cdots & \mathbf{a}_{mn} & c_m \end{array}\right)$$

$$\stackrel{?}{=} \left(\begin{array}{c|c|cccc|c} \mathbf{a}_{11} & \mathbf{a}_{12} & \mathbf{a}_{13} & \cdots & \mathbf{a}_{1n} & \langle \mathbf{a}_1, \mathbf{s} \rangle + \mathbf{e}_1 \\ \mathbf{a}_{21} & \mathbf{a}_{22} & \mathbf{a}_{23} & \cdots & \mathbf{a}_{2n} & \langle \mathbf{a}_2, \mathbf{s} \rangle + \mathbf{e}_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{a}_{m1} & \mathbf{a}_{m2} & \mathbf{a}_{m3} & \cdots & \mathbf{a}_{mn} & \langle \mathbf{a}_m, \mathbf{s} \rangle + \mathbf{e}_m \end{array}\right)$$

BKW Algorithm II

$$\Rightarrow \quad \left(\begin{array}{c|c|ccc|l} \mathbf{a}_{11} & \mathbf{a}_{12} & \mathbf{a}_{13} & \cdots & \mathbf{a}_{1n} & \langle \mathbf{a}_1, \mathbf{s}\rangle + \mathbf{e}_1 \\ 0 & \tilde{\mathbf{a}}_{22} & \tilde{\mathbf{a}}_{23} & \cdots & \tilde{\mathbf{a}}_{2n} & \langle \tilde{\mathbf{a}}_2, \mathbf{s}\rangle + \mathbf{e}_2 - \frac{\mathbf{a}_{21}}{\mathbf{a}_{11}}\mathbf{e}_1 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \\ 0 & \tilde{\mathbf{a}}_{m2} & \tilde{\mathbf{a}}_{m3} & \cdots & \tilde{\mathbf{a}}_{mn} & \langle \tilde{\mathbf{a}}_m, \mathbf{s}\rangle + \mathbf{e}_m - \frac{\mathbf{a}_{m1}}{\mathbf{a}_{11}}\mathbf{e}_1 \end{array}\right)$$

- $\frac{\mathbf{a}_{i1}}{\mathbf{a}_{11}}$ is essentially random in $\mathbb{Z}_q$ wiping all “smallness”.
- If $\frac{\mathbf{a}_{i1}}{\mathbf{a}_{11}}$ is 1 noise-size doubles because of the addition.

BKW Algorithm III

We considering $a \approx \log n$ 'blocks' of b elements each.

$$\left(\begin{array}{cc|ccc|c}
\mathbf{a}_{11} & \mathbf{a}_{12} & \mathbf{a}_{13} & \cdots & \mathbf{a}_{1n} & c_0 \\
\mathbf{a}_{21} & \mathbf{a}_{22} & \mathbf{a}_{23} & \cdots & \mathbf{a}_{2n} & c_1 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
\mathbf{a}_{m1} & \mathbf{a}_{m2} & \mathbf{a}_{m3} & \cdots & \mathbf{a}_{mn} & c_m
\end{array}\right)$$

BKW Algorithm IV

For each block we build a table of all q^b possible values indexed by $\mathbb{Z}_q^b$.

$$T^0 = \left[\begin{array}{cc|ccc|c} -\lfloor\frac{q}{2}\rfloor & -\lfloor\frac{q}{2}\rfloor & \mathbf{t}_{13} & \cdots & \mathbf{t}_{1n} & c_{t,0} \\ -\lfloor\frac{q}{2}\rfloor & -\lfloor\frac{q}{2}\rfloor + 1 & \mathbf{t}_{23} & \cdots & \mathbf{t}_{2n} & c_{t,1} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \lfloor\frac{q}{2}\rfloor & \lfloor\frac{q}{2}\rfloor & \mathbf{t}_{q^2 3} & \cdots & \mathbf{t}_{q^2 n} & c_{t,q^2} \end{array}\right]$$

For each $\mathbf{z} \in \mathbb{Z}_q^b$ find row in $\mathbf{A}$ which contains $\mathbf{z}$ as a subvector at the target indices.

BKW Algorithm V

Use these tables to eliminate b entries with one addition.

$$
\left(\begin{array}{cc|cccc|c}
\mathbf{a}_{11} & \mathbf{a}_{12} & \mathbf{a}_{13} & \cdots & \mathbf{a}_{1n} & c_0 \\
\mathbf{a}_{21} & \mathbf{a}_{22} & \mathbf{a}_{23} & \cdots & \mathbf{a}_{2n} & c_1 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \\
\mathbf{a}_{m1} & \mathbf{a}_{m2} & \mathbf{a}_{m3} & \cdots & \mathbf{a}_{mn} & c_m
\end{array}\right)
$$

$$
+ \quad \left[\begin{array}{cc|ccc|c}
-\lfloor \frac{q}{2} \rfloor & -\lfloor \frac{q}{2} \rfloor & \mathbf{t}_{13} & \cdots & \mathbf{t}_{1n} & c_{t,0} \\
-\lfloor \frac{q}{2} \rfloor & -\lfloor \frac{q}{2} \rfloor + 1 & \mathbf{t}_{23} & \cdots & \mathbf{t}_{2n} & c_{t,1} \\
\vdots & \vdots & \ddots & \vdots & \vdots & \\
\lfloor \frac{q}{2} \rfloor & \lfloor \frac{q}{2} \rfloor & \mathbf{t}_{q^2 3} & \cdots & \mathbf{t}_{q^2 n} & c_{t,q^2}
\end{array}\right]
$$

$$
\Rightarrow \quad \left(\begin{array}{cc|ccc|c}
\mathbf{a}_{11} & \mathbf{a}_{12} & \mathbf{a}_{13} & \cdots & \mathbf{a}_{1n} & c_0 \\
\mathbf{0} & \mathbf{0} & \tilde{\mathbf{a}}_{23} & \cdots & \tilde{\mathbf{a}}_{2n} & \tilde{c}_1 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \\
\mathbf{a}_{m1} & \mathbf{a}_{m2} & \mathbf{a}_{m3} & \cdots & \mathbf{a}_{mn} & c_m
\end{array}\right)
$$

BKW Algorithm VI

Memory requirement of

$$\approx \frac{q^b}{2} \cdot a \cdot (n+1)$$

and time complexity of

$$\approx (a^2 n) \cdot \frac{q^b}{2}.$$

A detailed analysis of the algorithm for LWE is available as:

M.A., Carlos Cid, Jean-Charles Faugère, Robert Fitzpatrick and Ludovic Perret
On the Complexity of the BKW Algorithm on LWE
In *Designs, Codes and Cryptography*.

BKW with Small Secret

Assume $\mathbf{s} \leftarrow_{\$} \mathcal{U}(\mathbb{Z}_2^n)$, i.e. all entries in secret $\mathbf{s}$ are very small.

Common setting in cryptography

- for performance reasons and
- to to realise some advanced functionality.

A technique called ‘modulus switching’ can be used to improve the performance of homomorphic encryption schemes.

Lazy Modulus Switching

Exploit the same structure to solve such instances faster with BKW.

M.A., Jean-Charles Faugère, Robert Fitzpatrick, Ludovic Perret
Lazy Modulus Switching for the BKW Algorithm on LWE.
In *PKC 2014*, Springer Verlag, 2014.

Complexity

BKW for $q = \text{poly}(n)$

$$\mathcal{O}\left(2^{cn} \cdot n \log_2^2 n\right)$$

BKW + naive modulus switching for $q = \text{poly}(n)$

$$\mathcal{O}\left(2^{\left(c + \frac{\log_2 \mathbf{d}}{\log_2 \mathbf{n}}\right) n} \cdot n \log_2^2 n\right)$$

BKW + lazy modulus switching for $q = \text{poly}(n)$

$$\mathcal{O}\left(2^{\left(c + \frac{\log_2 d - \frac{1}{2} \log_2 \log_2 \mathbf{n}}{\log_2 n}\right) n} \cdot n \log_2^2 n\right)$$

where $0 < d \leq 1$ is a small constant (so $\log d < 0$).

Contents

Introduction

BDD & SIS: Lattice Reduction

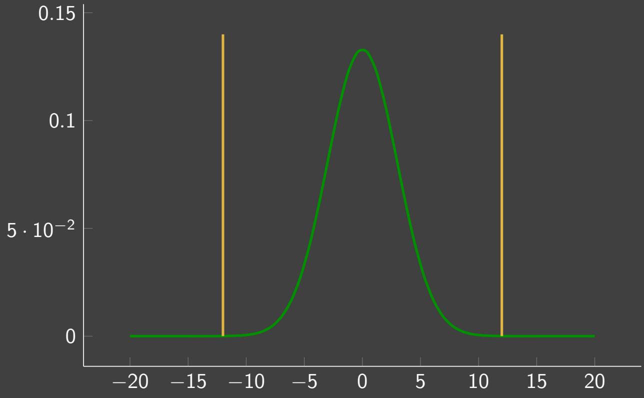
SIS: Combinatorial Algorithms

BDD: Arora & Ge

The Idea I

Noise follows a discrete Gaussian distribution, we have:

$$\Pr[e \leftarrow_{\$} \chi : \|e\| > C \cdot \sigma] \leq \frac{2}{C\sqrt{2\pi}} e^{-C^2/2} \in e^{\mathcal{O}(-C^2)}.$$



The Idea II

If $e \leftarrow_{\$} \chi$ and

$$P(X) = X \prod_{i=1}^{C\cdot\sigma} (X+i)(X-i),$$

we have $P(e) = 0$ with probability at least $1 - e^{\mathcal{O}(-C^2)}$.

If $(\mathbf{a}, c) = (\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$, and $e \leftarrow_{\$} \chi$, then

$$P\big(-c + \sum_{j=1}^{n} \mathbf{a}_{(j)} x_j\big) = 0,$$

with probability at least $1 - e^{\mathcal{O}(-C^2)}$.

The Idea III

Each $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e) = (\mathbf{a}, c)$ generates a **non-linear equation** of degree $2C\sigma + 1$ in the n components of the secret $\mathbf{s}$ which holds with probability $1 - e^{\mathcal{O}(-C^2)}$.

Solve this "noise-free" system of equations with **Gröbner bases**.

Tradeoff

More samples increase

1. the **number of equations** $\rightarrow$ solving is **easier**.
2. the required interval $C\sigma$ and hence the **degree** $\rightarrow$ solving is **harder**.

Complexity

Arora-Ge (Linearisation):

$$\mathcal{O}\left(2^{8\,\omega\,\sigma^2 \log n(\log n - \log(8\,\sigma^2 \log n))}\right)$$

Arora-Ge (Linearisation) with $\sigma = \sqrt{n}$

$$\mathcal{O}\left(2^{8\,\omega\,n \log n(\log n - \log(8\,n \log n))}\right)$$

Gröbner Bases with $\sigma = \sqrt{n}$

$$\mathcal{O}\left(2^{2.16\,\omega\,n}\right)$$

under some regularity assumption.

BinaryError-LWE

- BinaryError-LWE is a variant of LWE where the noise is $\{0,1\}$ but the number of samples severly restricted.
- Given access to $m = \mathcal{O}\left(n \log\log n\right)$ samples we can solve BinaryError-LWE in **subexponential** time:

$$\mathcal{O}\left(2^{\frac{\omega\, n\, \log\log\log n}{8 \log\log n}}\right).$$

M.A., Carlos Cid, Jean-Charles Faugère, Robert Fitzpatrick and Ludovic Perret
Gröbner Bases Techniques in LWE-Based Cryptography
To appear.

Fin

Questions?