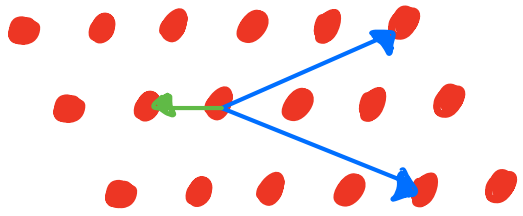
Some Things about Lattices



Matthew Fox

I : The Basics

II : Computational Problems

III : Crypto

Part I

The Basics

Def: • Given linearly independent $b_1, b_2, \dots, b_n \in \mathbb{R}^m$, which form the columns of

$$B = (b_1, b_2, \dots, b_n) \in \mathbb{R}^{m \times n},$$

a **lattice** $\mathcal{L} \subseteq \mathbb{R}^n$ is

$$\mathcal{L} = \mathcal{L}(B) = \{Bx \mid x \in \mathbb{Z}^n\}$$

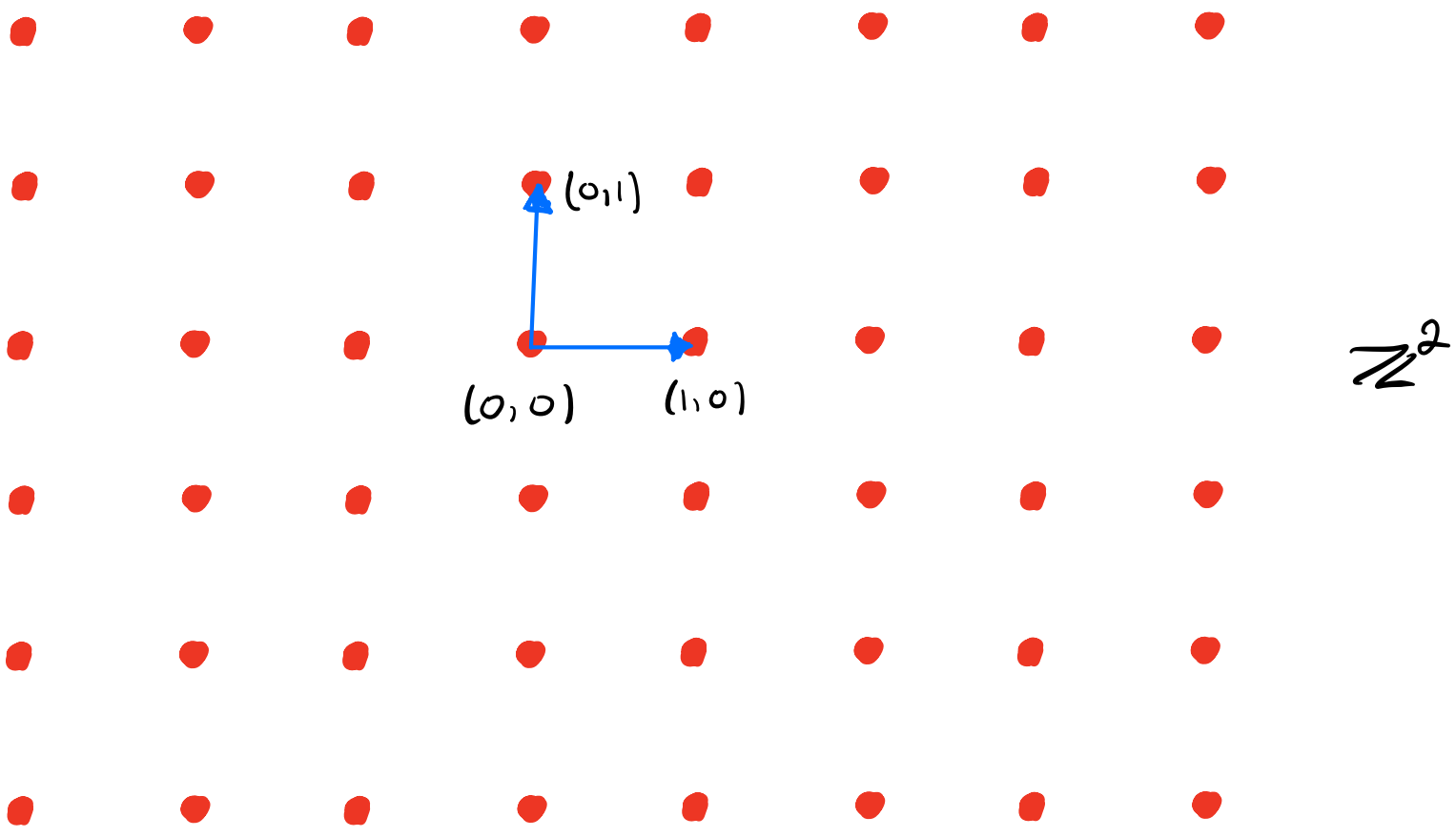
$$= \{x_1 b_1 + x_2 b_2 + \cdots + x_n b_n\}.$$

• B is a **basis**.

Ex:

$$B = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

What is $\mathcal{L}(B) = \{Bx \mid x \in \mathbb{Z}^2\}$?

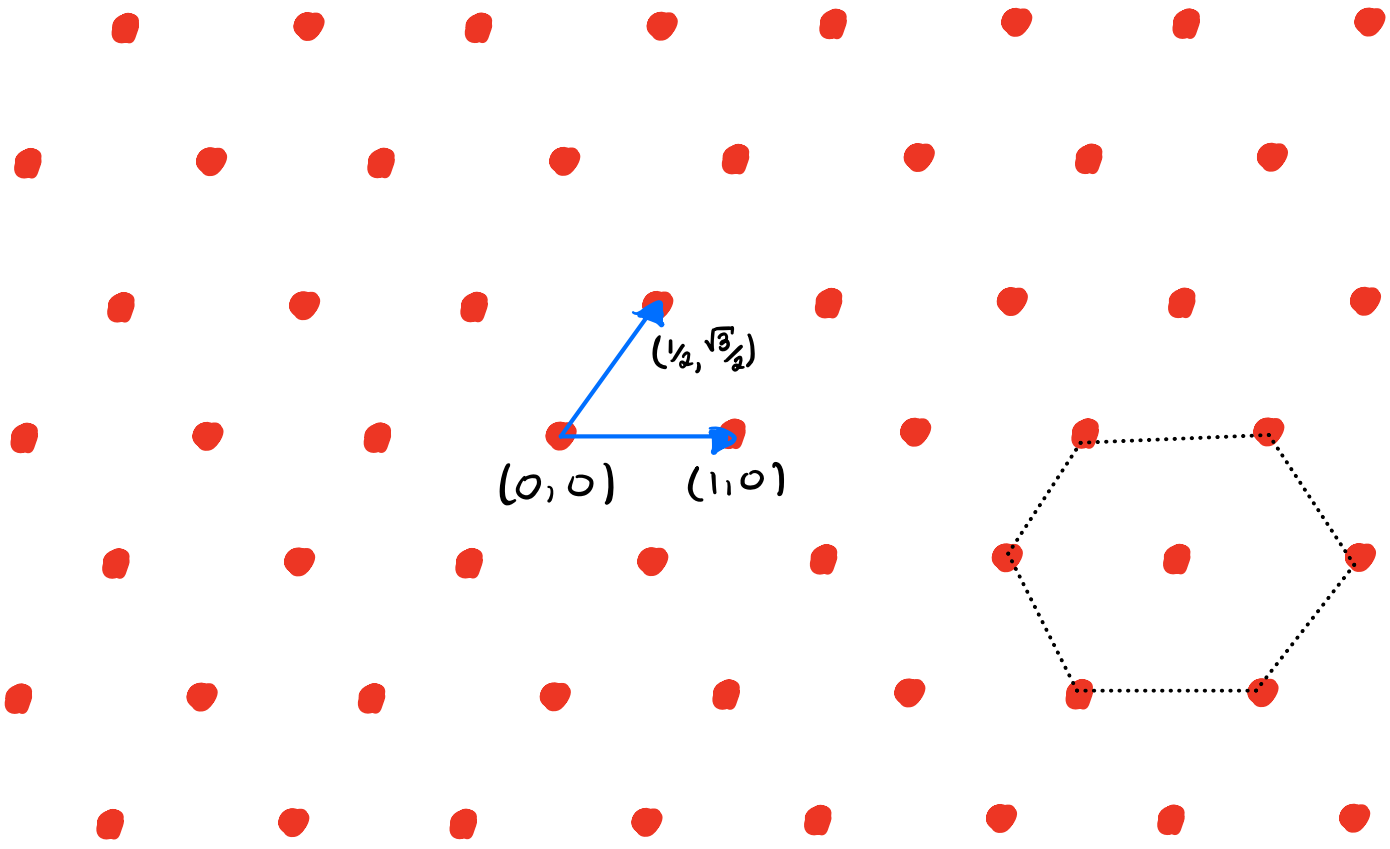

(0,1)
(0,0)
(1,0)
$\mathbb{Z}^2$

Ex:

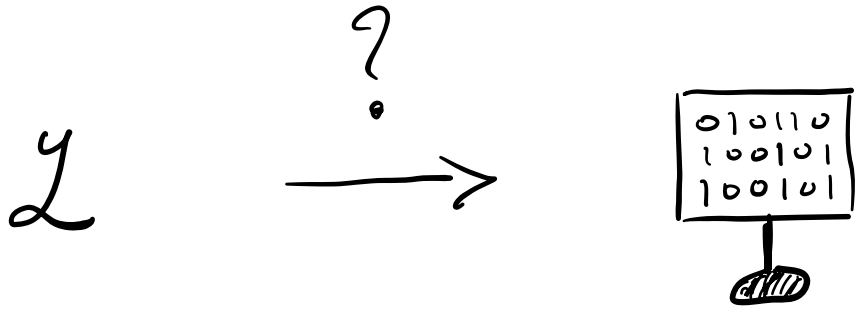
$$B = \begin{pmatrix} 1 & 1/2 \\ 0 & \sqrt{3}/2 \end{pmatrix}$$

What is $\mathcal{L}(B) = \{Bx \mid x \in \mathbb{Z}^2\}$?

"Honeycomb Lattice"



Q: How to represent $\mathbb{Z}$ on a Computer?



A: Input B s.t. $\mathcal{L} = \mathcal{L}(B)$!

However...

B might contain irrational (perhaps even uncomputable) $a \in \mathbb{R}$.

Fix: Restrict to integer lattices $\mathcal{L} \subseteq \mathbb{Z}^n$.

- Given $B \in \mathbb{Z}^{m \times n}$ and $\ell = \max_{i,j} \log B_{ij}$,

$$\text{encoded size}(\mathcal{L}(B)) \leq m \cdot n \cdot \ell .$$

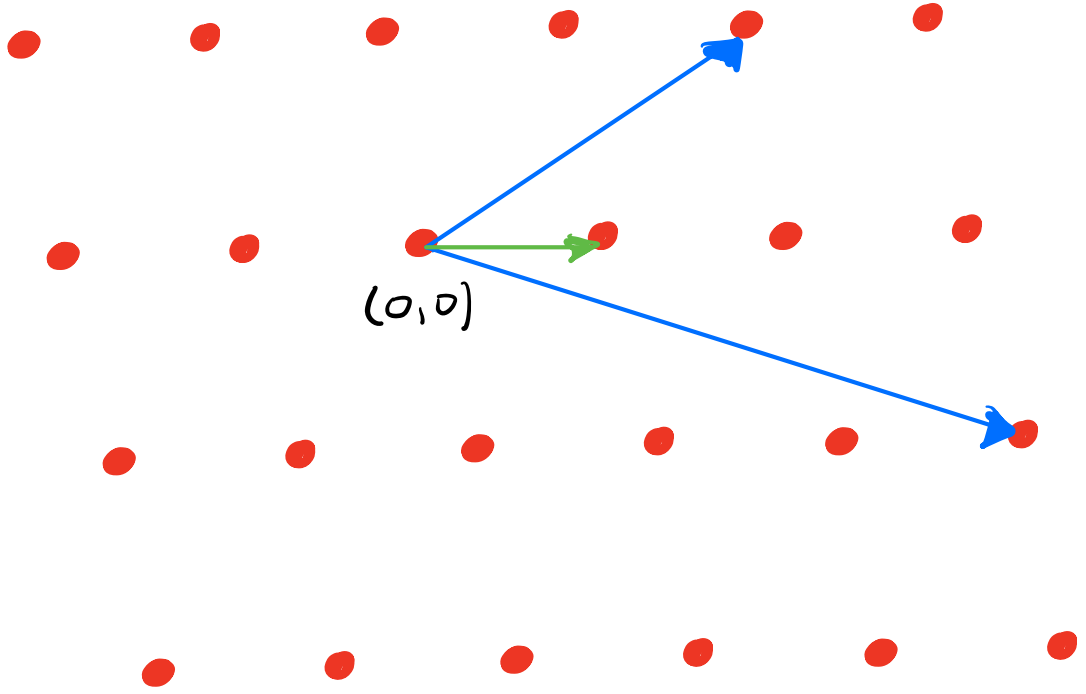
- Therefore, say algorithm is efficient iff

$$\text{runtime} = \text{Poly}(m, n, \ell).$$

Part II

Computational Problems

The Shortest Vector Problem (SVP)

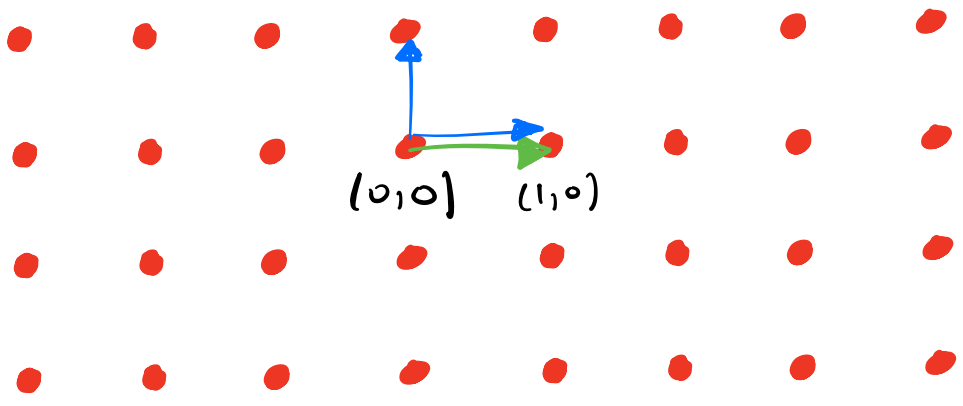


$$\lambda_1(\mathcal{L}) := \min_{x \in \mathcal{L} \setminus \{0\}} \|x\|_2$$

Ex: $\mathcal{L} = \mathcal{L}\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \mathbb{Z}^2$.

What is

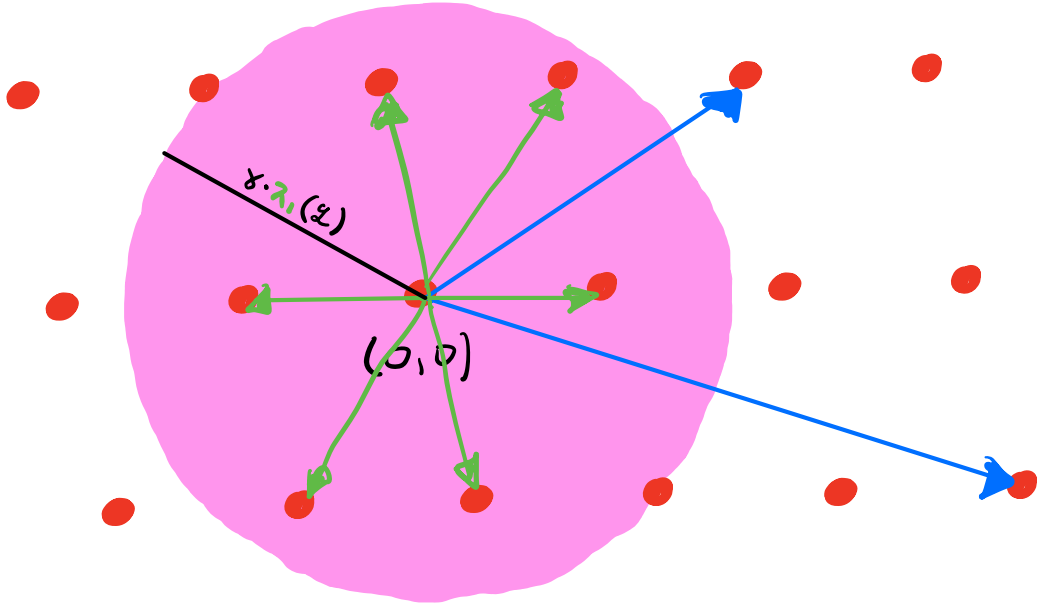
$$\lambda_1(\mathbb{Z}^2) = \min_{x \in \mathbb{Z}^2 \setminus \{0\}} \|x\|_2 \quad ?$$



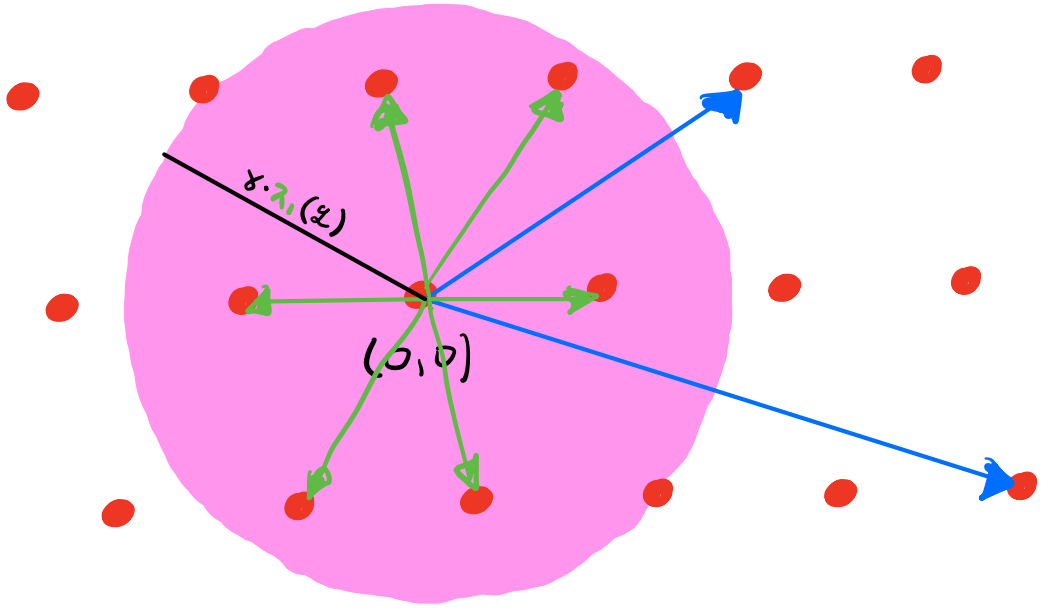
Def: For $\gamma = \gamma(n) \geq 1$, the **$\gamma$-approximate Shortest Vector Problem (γ-SVP)** is:

Input: $B \in \mathbb{Z}^{m \times n}$ of lattice $\mathcal{L} = \mathcal{L}(B)$

Output: $v \in \mathcal{L} \setminus \{0\}$ s.t. $\|v\|_2 \leq \gamma \cdot \lambda_1(\mathcal{L})$.



Fact: The larger γ, the easier γ-SVP.

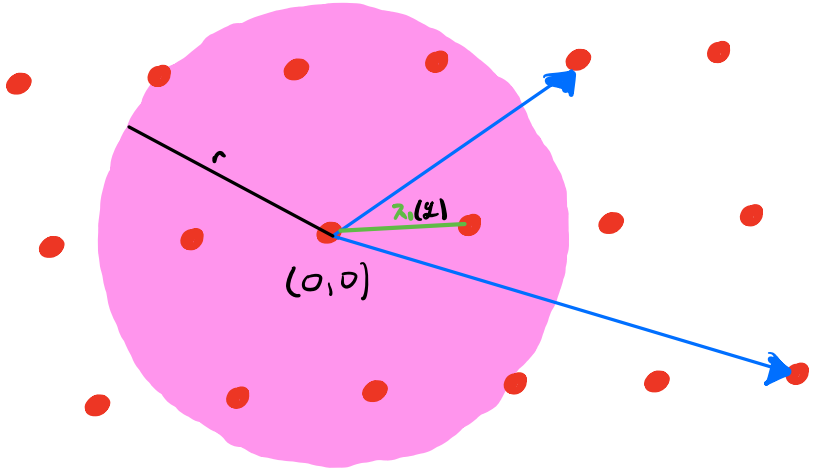


Def: For $\gamma = \gamma(n) \geq 1$, the **$\gamma$-approximate decisional SVP (γ-GapSVP)** is the Promise Problem:

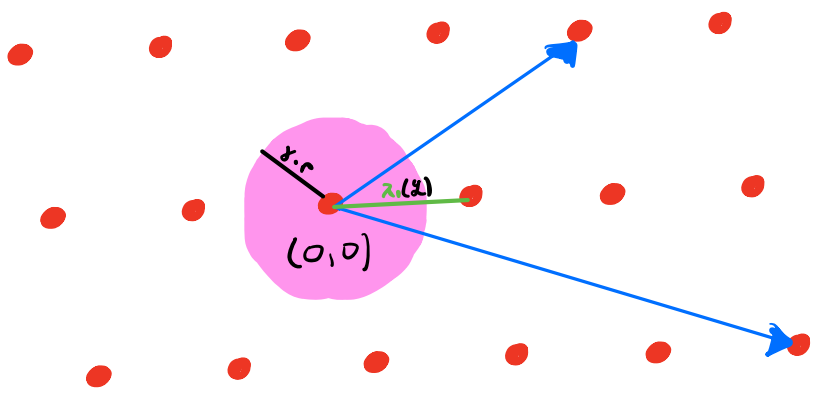
Input: (B, r), where
- $B \in \mathbb{Z}^{m \times n}$ of lattice $\mathcal{L} = \mathcal{L}(B)$
- $r > 0$

Output: "YES" if $\lambda_1(\mathcal{L}) \leq r$

"NO" if $\lambda_1(\mathcal{L}) > \gamma r$



"YES"



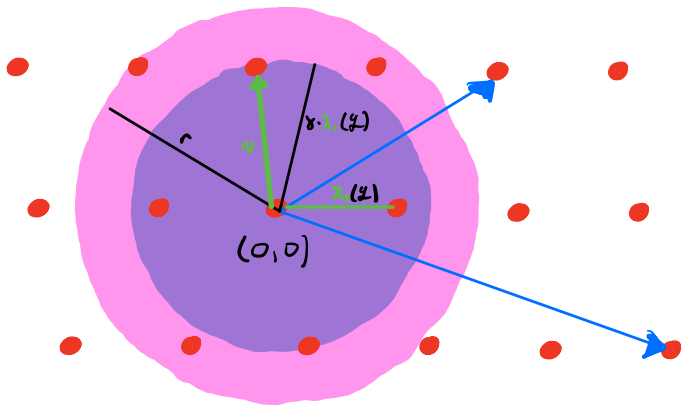
"NO"

Claim: An efficient (classical or quantum) algorithm $\mathcal{A}$ for γ-SVP implies an efficient algorithm for γ-GapSVP. Therefore, γ-SVP is at least as hard as γ-GapSVP.

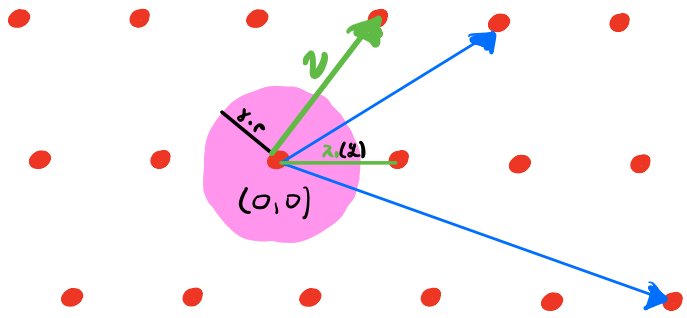
Proof: If $(\mathbf{B}, r)$ is a γ-GapSVP instance, then

$$\text{output} \begin{cases} \text{"YES" if } v = \mathcal{A}(\mathbf{B}) \text{ s.t. } \|v\| \leq \gamma r \\ \text{"NO" if } v = \mathcal{A}(\mathbf{B}) \text{ s.t. } \|v\| > \gamma r. \end{cases}$$

Note, $\|v\| \leq \gamma \cdot \lambda_1(\mathcal{L})$ by definition of $\mathcal{A}$. ■



"YES" ($\lambda_1(\mathcal{L}) \leq r$)



"NO" ($\lambda_1(\mathcal{L}) > \gamma r$)

Claim: For all $\gamma = \gamma(n)$, γ-GapSVP $\in$ NP.

Proof: Given instance (B, r),

(B, r) is "YES" instance

$\Rightarrow$

$\lambda_1(\mathcal{L}) \leq r$

$\Leftrightarrow$

$\exists v \in \mathcal{L} \setminus \{0\}$ s.t. $\|v\| \leq r$.

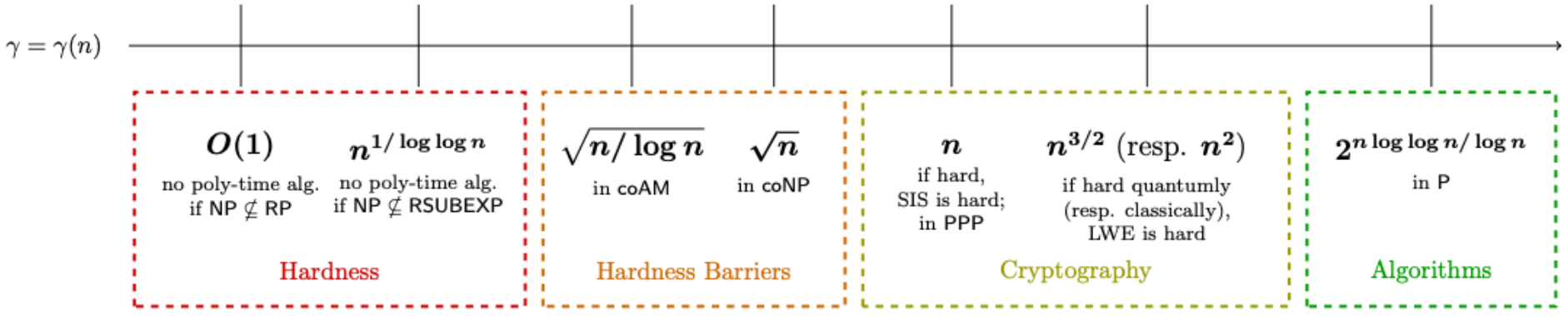
Moreover,

encoded length$(v) \leq n \cdot$ encoded length(v_i)

$\leq n \cdot \log r$.

Therefore, v is poly-size witness. ∎

The Complexity Landscape of γ-GapSVP



(ECCC TR22-170)

An Aside ...

Let

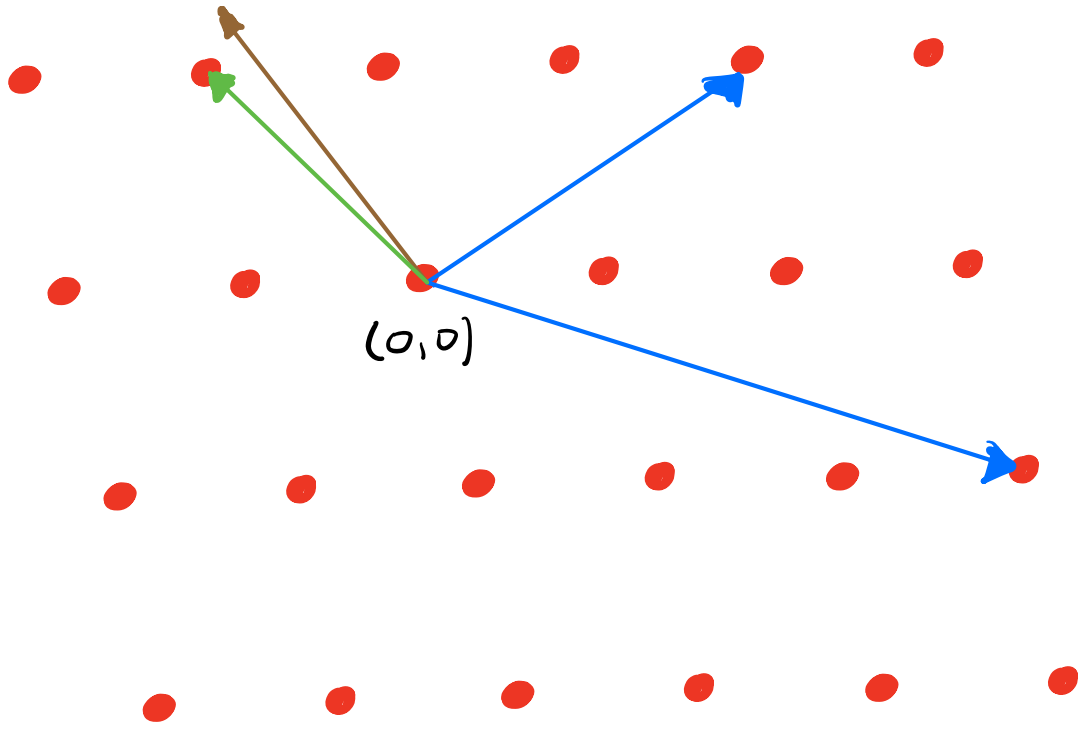
$$D_n = \{\text{symmetries of } n\text{-sided regular polygon}\}$$

E.g.

$$D_8 = \left\{ \text{  } \right\}$$

Theorem: An efficient (classical or quantum) algorithm for the non-abelian hidden subgroup problem over D_n implies an efficient algorithm for $\text{Poly}(n)$-GapSVP.

The Closest Vector Problem (CVP)

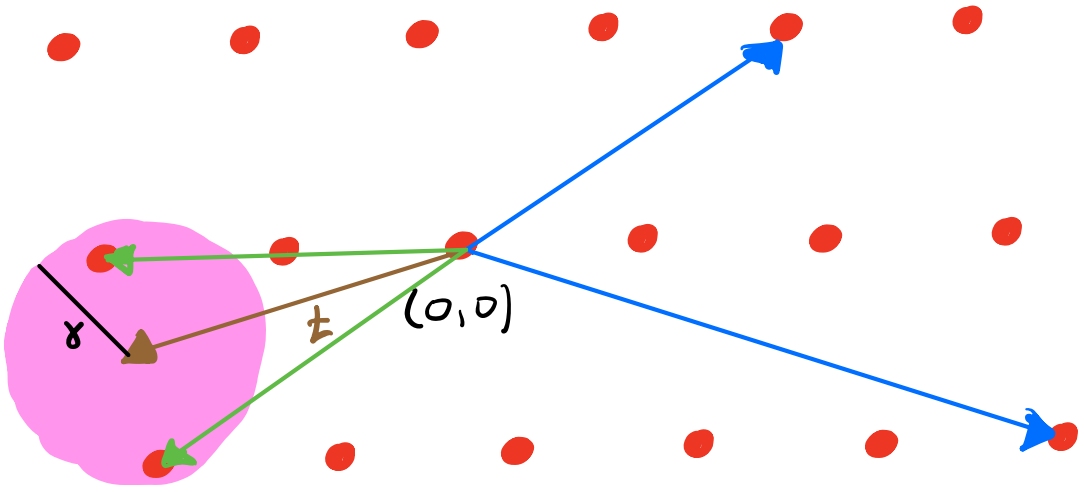


Def: For $\gamma = \gamma(n) \geq 1$, the γ-approximate closest vector problem (γ-CVP) is:

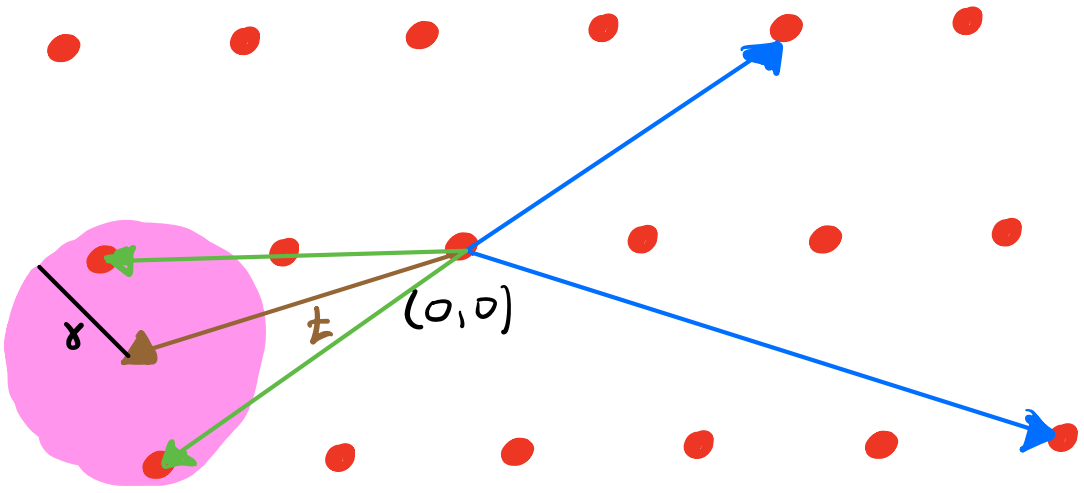
Input: (B, t), where

- $B \in \mathbb{Z}^{m \times n}$ of lattice $\mathcal{L} = \mathcal{L}(B)$
- $t \in \mathbb{R}^n$

output: $v \in \mathcal{L}$ s.t. $\|v - t\|_2 \leq \gamma$.



Fact: The larger γ, the easier γ-CVP.



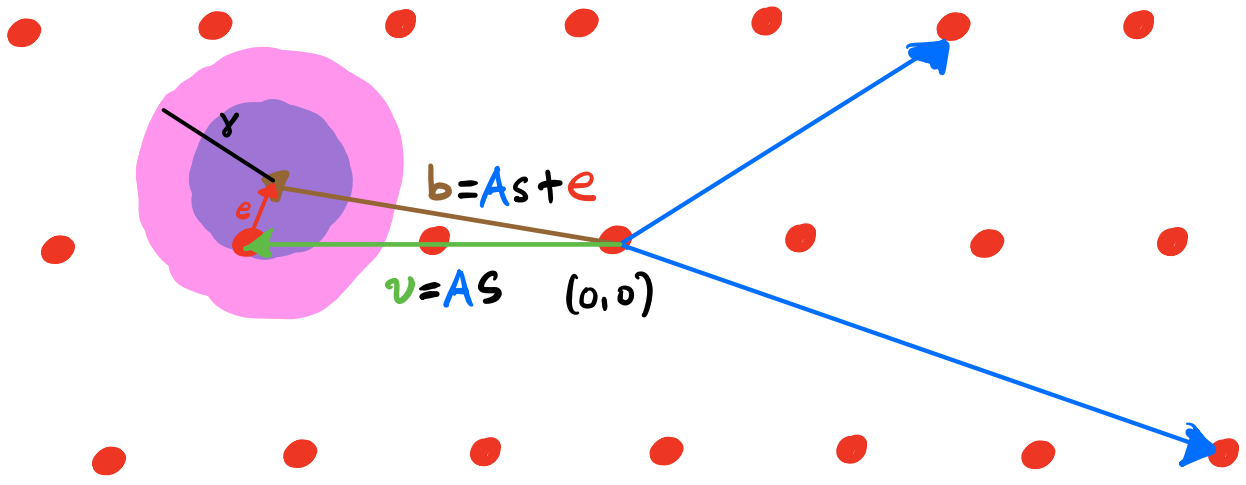
Theorem (Goldreich et al., 1999):

An efficient (classical or quantum) algorithm $\mathcal{A}$ for γ-CVP implies an efficient algorithm for γ-SVP. Therefore, γ-CVP is at least as hard as γ-GapSVP.

Non-Proof:
- Let B be an instance of γ-SVP.
- Call $\mathcal{A}(B, t = 0)$ ("closest vector to zero")
- Return $v \in \mathcal{L}$ s.t. $\|v\| \leq \gamma \leq \gamma \lambda_1(\mathcal{L})$.

Q: What's wrong with this?

The Learning With Errors Problem (LWE)



Def: For integers $m > n$, Modulus $q \geq 2$, and $\chi \sim \mathbb{Z}_q$, the (m, q, χ) - LWE Problem is:

Input: (A, b), where

- $A \sim \mathbb{Z}_q^{m \times n}$ uniformly
- $b = As + e \pmod q$ where
  - $s \sim \mathbb{Z}_q^n$ uniformly
  - $e \sim \chi^m$

Output: Find s.

"Claim": If $\Pr_{e \sim \chi^m}[\|e\| \text{ is "large"}] \ll 1$, then an efficient (classical or quantum) algorithm for (m, q, χ)-LWE implies an efficient algorithm for γ-CVP.

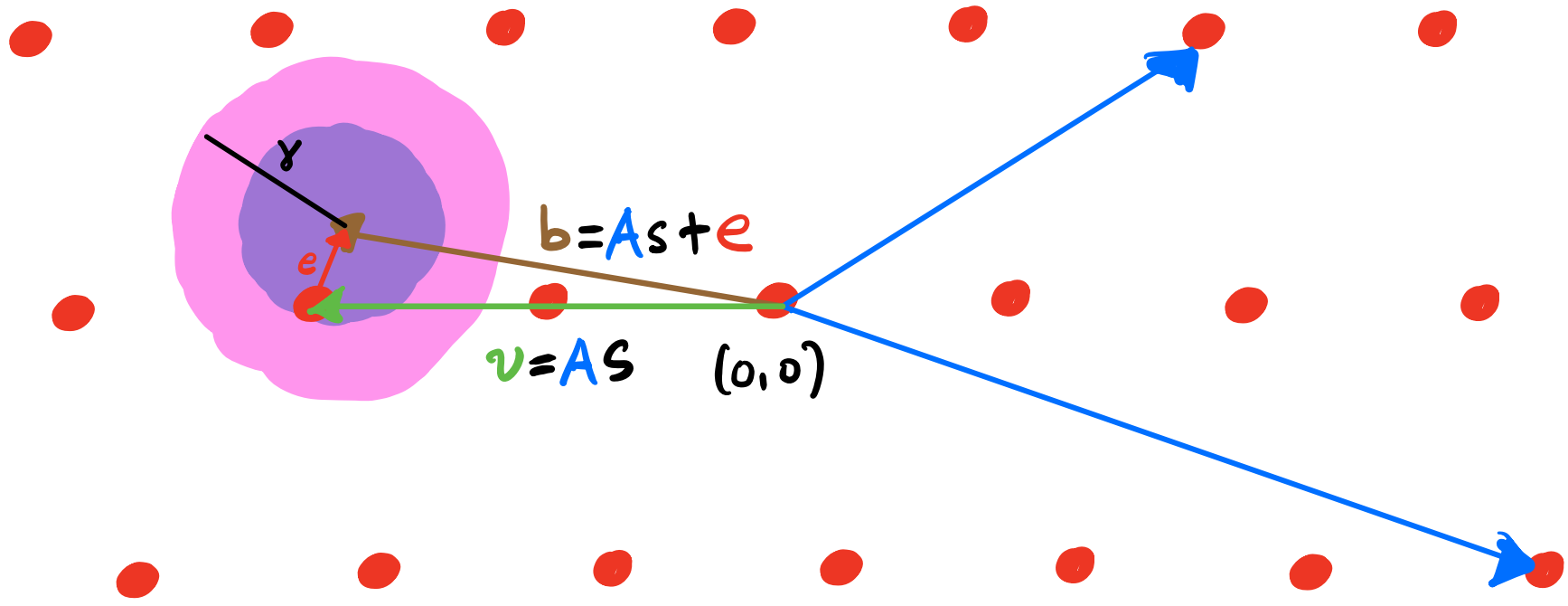
"Proof": Given LWE instance $(A, b = As + e \pmod q)$, let

$$\mathcal{L}_A = \{x \mid \exists t \in \mathbb{Z}_q^n : x = At \pmod q\} \subset \mathbb{Z}^m$$

LWE gives s. Since, by definition,

$$\gamma\text{-CVP}(A, b) = v \in \mathcal{L}_A \text{ s.t. } \|v - b\| \leq \gamma,$$

if $\|e\|$ is "small", then $v = As$ solves γ-CVP. ∎


y
b=As+e
e
v=As
(0,0)

Theorem (Regev, 2005):

For an appropriate choice of parameters, an efficient (classical or quantum) algorithm for (m, q, χ)-LWE implies an efficient algorithm for $\text{poly}(n)$-CVP. Therefore, (m, q, χ)-LWE is at least as hard as $\text{poly}(n)$-GapSVP.

"Worst-to-average case reduction."

Part III

Crypto

The Magic of Public-Key Crypto

Scott and I have never communicated before

I say Something

Everyone hears

Scott says Something

Everyone hears

I say Something

Everyone hears

only Scott understands

Def: A **Public-Key Crypto System** consists of three efficient algorithms:

- **Gen** : $1^n \longmapsto (P, S)$ (Key generation)
- **Enc** : $(P, \mu \in \{0,1\}) \longmapsto C$ (encryption/cipher)
- **Dec** : $(S, C) \longmapsto \mu$ (decryption).

We say a system is **secure** iff $\forall$ efficient adversaries A,

$$Pr[A(P, C) = \mu] \leq \frac{1}{2} + \text{negligible fnc.}$$

I.e., A cannot do better than randomly guessing μ.

Regev Encryption

- $\mathsf{Gen}(1^n) = (P, S)$, where
  - $S \sim \mathbb{Z}_q^n$
  - $P = (A, b = AS + e \pmod q)$, $A \sim \mathbb{Z}_q^{m \times n}$ and $e \sim \chi^m$

- $\mathsf{Enc}(P, \mu \in \{0,1\}) = (c, d)$, where
  - $c = (r^T A \bmod q)^T \in \mathbb{Z}_q^n$, $r \sim \{0,1\}^m$
  - $d = r^T b + \mu \lfloor q/2 \rfloor \pmod q \in \mathbb{Z}_q$

- $\mathsf{Dec}(S, (c, d)) = \begin{cases} 0 & \text{if } d - c^T s \pmod q \in [-q/4, q/4], \\ 1 & \text{otherwise.} \end{cases}$

Proof of Correctness:

Since $b - As = e \pmod q$,

$$d - c^T s = r^T b + \mu \lfloor q/2 \rfloor - r^T A s$$

$$= r^T (b - As) + \mu \lfloor q/2 \rfloor$$

$$= r^T e + \mu \lfloor q/2 \rfloor \pmod q.$$

If, as in LWE,

$$\Pr_{e \sim \chi^m} \left[\|e\| > q/10\sqrt{m} \right] \ll 1,$$

then

$$\|r^T e\| \le \|r^T\| \|e\| \le q/10 \qquad \text{(w/ high prob.)}$$

So, indeed,

$$\mu = 0 \iff d - c^T s \pmod q \in \left[-q/4, q/4 \right].$$

Theorem (Regev 2005):

For an appropriate choice of parameters, if Regev encryption is not secure, then there exists an efficient algorithm for (m, q, χ)-LWE. Therefore, breaking Regev encryption is at least as hard as $\text{poly}(n)$-GapSVP.

Some Properties of Regev Encryption

- Efficient in practice, and can be made faster by imposing more structure (RingLWE)
- No modular exponeniation like in RSA
- Fully homomorphic
- Secure against quantum adversaries (so far!)

Cryptographic Suite for Algebraic Lattices ("CRYSTALS")

Introduction

Kyber is an IND-CCA2-secure key encapsulation mechanism (KEM), whose security is based on the hardness of solving the learning-with-errors (LWE) problem over module lattices. Kyber is one of the finalists in the NIST post-quantum cryptography project. The submission lists three different parameter sets aiming at different security levels. Specifically, Kyber-512 aims at security roughly equivalent to AES-128, Kyber-768 aims at security roughly equivalent to AES-192, and Kyber-1024 aims at security roughly equivalent to AES-256.

For users who are interested in *using* Kyber, we recommend the following:

- Use Kyber in a so-called *hybrid mode* in combination with established "pre-quantum" security; for example in combination with elliptic-curve Diffie-Hellman.
- We recommend using the Kyber-768 parameter set, which—according to a very conservative analysis—achieves more than 128 bits of security against all known classical and quantum attacks.

Scientific Background

The design of Kyber has its roots in the seminal LWE-based encryption scheme of Regev. Since Regev's original work, the practical efficiency of LWE encryption schemes has been improved by observing that the secret in LWE can come from the same distribution as the noise and also noticing that "LWE-like" schemes can be built by using a square (rather than a rectangular) matrix as the public key. Another improvement was applying an idea originally used in the NTRU cryptosystem to define the Ring-LWE and Module-LWE problems that used polynomial rings rather than integers. The CCA-secure KEM Kyber is built on top of a CPA-secure cryptosystem that is based on the hardness of Module-LWE.

Users of Kyber

Kyber is already being integrated into libraries and systems by industry. For example,

- Cloudflare integrated Kyber alongside other PQ algorithms into CIRCL, the Cloudflare Interoperable, Reusable Cryptographic Library;
- Amazon now supports hybrid modes involving Kyber in their AWS Key Management Service; and
- already in 2019 IBM advertised the "World's First Quantum Computing Safe Tape Drive" using Kyber and Dilithium.

Thank You!