Computational problems on lattices: [problems parameterized by lattice dimension n] ← $n=2$ case is easy (can solve exactly using Gauss' algorithm)

- Shortest vector problem (SVP): Given a basis B of an n-dimensional lattice $\mathcal{L} = \mathcal{L}(B)$, find $v \in \mathcal{L}$ such that $\|v\| = \lambda_1(\mathcal{L})$
- Approximate SVP (SVP_γ): Given a basis B of an n-dimensional lattice $\mathcal{L} = \mathcal{L}(B)$, find $v \in \mathcal{L}$ such that $\|v\| \le \gamma \cdot \lambda_1(\mathcal{L})$
- Decisional approximate SVP (GapSVP_γ): Given a basis B of an n-dimensional lattice $\mathcal{L} = \mathcal{L}(B)$, decide if $\lambda_1(\mathcal{L}) \le 1$ or if $\lambda_1(\mathcal{L}) \ge \gamma$
  (Promise problem: one of these cases is guaranteed)
- Approximate shortest independent vectors (SIVP_γ): Given basis of full-rank n-dimensional lattice $\mathcal{L} = \mathcal{L}(B)$, output a set of linearly independent vectors $b_1', \ldots, b_n'$ where $\|b_i'\| \le \gamma \cdot \lambda_n(\mathcal{L})$ for all $i \in [n]$.

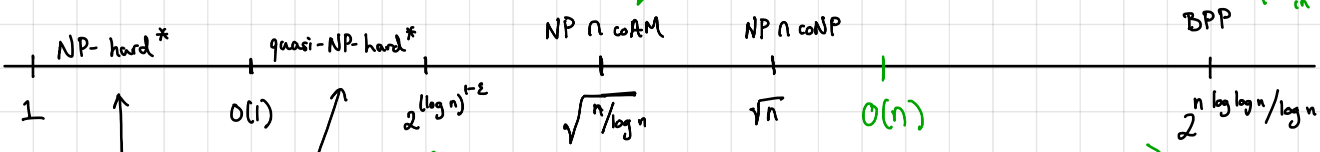
Main problems we use for cryptography are short integer solutions (SIS) and learning with errors (LWE)

↳ These reduce to GapSVP_γ and SIVP_γ

↳ Currently open: basing crypto on search-SVP (SVP or SVP_γ)

Complexity of GapSVP depends on approximation factor γ:

example language in $NP \cap coAM$ is graph isomorphism (not known to be in coNP)



under randomized reductions

[mapping NO instances to NO instances w.p. 1 and YES instances to YES instances w.p. 2/3]

"nearly polynomial"

approximation factor sufficient for cryptography (e.g., OWF / PKE exists)

unlikely to allow basing crypto on NP hardness since for approximation factors bigger than $\sqrt{n}$, $\mathrm{GapSVP}_\gamma \in NP \cap coNP$

<u>Open questions</u>: Derandomizing reductions for some gap?
(NP-hardness result known for ℓ_∞ norm up to nearly polynomial factors)
Poly-time reductions for super-constant approximation factor?

<u>Algorithms for SVP</u>: Lenstra-Lenstra-Lovsaz (LLL) algorithm (lattice reduction)

- Polynomial time algorithm for $\gamma = 2^{n \log\log n/\log n}$ approximation
- Known algorithms for $\mathrm{poly}(n)$ approx run in time $2^{\Theta(n)}$ (many need similar space as well)
- Can trade-off time for approximation factor: solve GapSVP_γ in time $2^{\Theta(n/\log\gamma)}$
- Same asymptotics with quantum algorithms

For cryptographic constructions, it is oftentimes more convenient to use average-case problems (which admit reductions from GapSVP)

- Specifically, we rely on the short integer solutions (SIS) or the learning with errors (LWE) problems, which are <u>average-case</u> problems
- Both the SIS and the LWE problems can be based on the hardness of the GapSVP problem (e.g., an adversary that solves SIS or LWE can be used to solve GapSVP in the <u>worst-case</u>)

<u>Short Integer Solutions (SIS)</u>: The SIS problem is defined with respect to lattice parameters n, m, q and a norm bound β. The $\mathrm{SIS}_{n,m,q,\beta}$ problem says that for $A \xleftarrow{R} \mathbb{Z}_q^{n\times m}$, no efficient adversary can find a non-zero vector $x \in \mathbb{Z}^m$ where

$$Ax = 0 \in \mathbb{Z}_q^n \quad \text{and} \quad \|x\| \leq \beta$$

In lattice-based cryptography, the lattice dimension n will be the primary security parameter.

<u>Notes</u>: - The norm bound β should satisfy $\beta \leq q$. Otherwise, a trivial solution is to set $x = (q, 0, 0, \ldots, 0)^T$.

- We need to choose m, β to be large enough so that a solution does exist.
  - ↳ When $m = \Omega(n \log q)$ and $\beta > \sqrt{m}$ a solution always exists. In particular, when $m \geq \lceil n \log q \rceil$, there always exists $x \in \{-1, 0, 1\}^m$ such that $Ax = 0$:
    - There are $2^m \geq 2^{n \log q} = q^n$ vectors $y \in \{0,1\}^m$
    - Since $Ay \in \mathbb{Z}_q^n$, there are at <u>most</u> q^n possible outputs of Ay

      } By a counting argument, there exist $y_1 \neq y_2 \in \{0,1\}^m$ such that $Ay_1 = Ay_2$
    - Thus, if we set $x = y_1 - y_2 \in \{-1,0,1\}^m$, then $Ax = A(y_1 - y_2) = Ay_1 - Ay_2 = 0 \in \mathbb{Z}_q^n$ and $\|y_1 - y_2\| \leq \sqrt{m}$

SIS can be viewed as an <u>average-case</u> SVP on a lattice defined by $A \in \mathbb{Z}_q^{n\times m}$:

$$\mathcal{L}^\perp(A) = \{x \in \mathbb{Z}^m : Ax = 0 \pmod q\}$$

in coding-theoretic terms, the matrix A is a "parity-check" matrix

called a "q-ary" lattice since $q\mathbb{Z}^m \subseteq \mathcal{L}^\perp(A)$

SIS problem is essentially finding short vectors in the lattice $\mathcal{L}^\perp(A)$ where $A \xleftarrow{R} \mathbb{Z}_q^{n\times m}$

<u>Theorem</u>. For any $m = \mathrm{poly}(n)$, any $\beta > 0$, and sufficiently large $q \geq \beta \cdot \mathrm{poly}(n)$, there is a probabilistic polynomial time (PPT) reduction from solving GapSVP_γ or SIVP_γ in the worst case to solving $\mathrm{SIS}_{n,m,q,\beta}$ with non-negligible probability, where $\gamma = \beta \cdot \mathrm{poly}(n)$.

<u>Implication</u>: Algorithm for $\mathrm{SIS}_{n,m,q,\beta}$ $\Rightarrow$ algorithm for $\mathrm{GapSVP}_\gamma / \mathrm{SIVP}_\gamma$ in the worst case

$\mathrm{GapSVP}_\gamma / \mathrm{SIVP}_\gamma$ hard in the worst case $\Rightarrow$ $\mathrm{SIS}_{n,m,q,\beta}$ hard on average

<u>Tightness</u>. Micciancio-Regev [MR04]: $\gamma = \beta \cdot \tilde{O}(\sqrt{n})$ — can set β so that $\gamma = \tilde{O}(n)$ with $q = \beta \cdot \tilde{O}(n\sqrt{m}) = n^{O(1)}$ ✓ for cryptographic applications

Gentry-Peikert-Vaikuntanathan [GPV08]: improved bound on q to be $q = \beta \cdot \tilde{O}(\sqrt{n})$ with $\gamma = \beta \cdot \tilde{O}(\sqrt{n})$ approximation factor

Micciancio-Peikert [MP12]: improve bound on q to $\beta \cdot n^\varepsilon$ for <u>any</u> $\varepsilon > 0$ (<u>nearly</u> optimal since $\beta < q$)

(approximation factor in MP12 bound depends on ℓ_∞-norm rather than just ℓ_2-norm)

Difficulty in these worst-case to average-case reductions is needing to take an <u>arbitrary</u> problem instance and embedding it in a <u>random</u> instance

[We will not cover it today, but may discuss in future lecture]